Hierarchical Optimization for Operationally-Constrained Resource Planning

Chen Shengkai
*Institute for Infocomm Research (I^2R), A*STAR*
Singapore
Chen_Shengkai@i2r.a-star.edu.sg

Wang Yu
*Institute for Infocomm Research (I^2R), A*STAR*
Singapore
Yu_Wang@i2r.a-star.edu.sg

Ramasamy Savitha
*Institute for Infocomm Research (I^2R), A*STAR*
Singapore
ramasamysa@i2r.a-star.edu.sg

Cheryl Wong Sze Yin
*Institute for Infocomm Research (I^2R), A*STAR*
Singapore
Cheryl_Wong@i2r.a-star.edu.sg

***Abstract*—Effective resource planning presents a broad problem across industries due to inner operational constraints. Existing methods are hard for generalization because they entail either unique models or customized specific solutions. To address this challenge, we pose the Operationally-Constrained Resource Planning Problem (OCRPP), which abstract assets and resources using job concepts to decouple the entangled constraints to describe the resource planning problem in a standard way. Meanwhile, we propose the Hierarchical Allocation Optimizer (HAO), a meta-heuristic solving framework containing 3 phases, to speed up feasible solution search with lower bound estimation. Experiments show HAO's superiority in terms of time and objectives compare to the alternatives, which adopts rule-based heuristic or general meta-heuristic. HAO's rapid response and flexibility are demonstrated to show its capability to adapt to various business scenarios and applications such as airlines, logistics companies, and so on.**

I. Introduction

A. Background and motivation

Resource planning optimization is a widely adopted cost saving strategy in various business scenarios and applications. For instance, in the pursuit of gaining a competitive edge, companies are focused optimizing human resource acquisition to foster growth and excellence [1]. In terminal operations, berth and crane allocation underscore the significance of resource management [2]. Similarly, industries like aviation and railway transportation necessitate meticulous coordination of aircraft, engines, crews, and train units for the sake of safety and efficient operations [3]–[6].

During the process of resource planning, it is necessary to consider additional operational constraints from asset management to ensure safety, flexibility and adaptability [7], [8]. These constraints, often multi-dimensional and interdependent, present great challenges in achieving optimal resource allocation. The existing studies on resource planning with particular side constraints tend to focus on a limited range of specific resource categories and designed for short-term planning scenarios [9]–[11], thus they may fail to fulfill real-world demands.

To address these challenges, we generalize the modelling for Operationally-Constrained Resource Planning Problem (OCRPP) that can be adapted to customized contexts. The proposed OCRPP framework decouples operational constraints associated with resources by introducing auxiliary "active" resource variables with the help of Constraint Programming (CP) techniques [1], [12].

The scale of the real-world long term horizon OCRPP can become very large due to the combination of resources, assets and time horizon. To tackle the optimization problems with large searching space, innovative strategies have surfaced, including multi-step solution grouping [13], machine learning-based prediction [14], fixed-point models employing efficient search techniques [15], and so forth. Although these approaches hold intuitive promise, they often cater to specific problems and cannot be adapted to different contexts.

To fill the research gap, we propose an meta-heuristic solving framework called Hierarchical Allocation Optimizer (HAO) as depicted in Fig. 1. The HAO employs a multi-phase solving strategy, enabling efficient exploration of feasible solutions for OCRPP.

B. Contributions

- We use CP techniques to formulate the OCRPP model, which **decouples the inter-dependencies among different dimension of operational constraints** to well abstracted key components for common resource planning problems.
- We propose a **novel meta-heuristic HAO framework that contains 2 warm up phases and 1 fine-tuning phase** as depicted in Fig. 1. The warm up phases accelerate the searching of high quality feasible solutions, then they will be fed in the fine-tuning phase to obtain the near optimal solutions.

This work is supported by the Agency for Science, Technology and Research (A*STAR) under the following Industry Alignment Fund - Industry Collaboration Projects (IAF-ICP): AI for Airline Operations (I2001E0076) and Automatic Workflow Tracing and Optimization for Prescriptive MRO (I2001E0073).

979-8-3503-5409-6/24/$31.00 ©2024 IEEE
DOI 10.1109/CAI59869.2024.00132

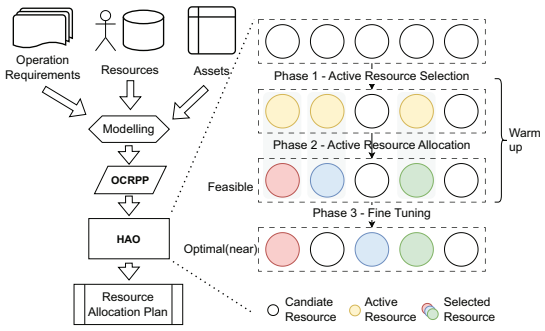


Fig. 1. Optimization Architecture for Resource Planning

Overall, these contributions provide a general framework for optimizing resource planning with operational constraints, and a practical approach for managing assets and resources in various business scenarios.

II. Modelling

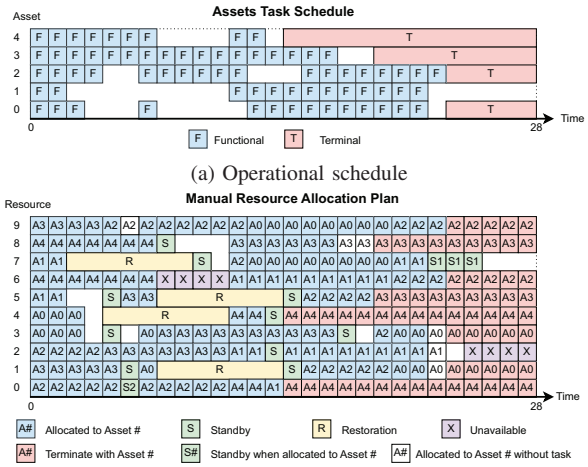
A. Problem definition

The key components of the generalised Operationally-Constrained Resource Planning Problem (OCRPP) are described as follows:

- Asset ($j \in \mathcal{J}$): The infrastructure equipment like machinery or production line, which needs allocate certain amount (C_j) of resources to operate with.
- Task: Pre-planned periodically operational schedule for assets to perform, mainly includes:
  - Functional: For any asset j in operation, it consumes resources' capacity volume by $V_j^{(f)}$.
  - Termination: For any asset j that is for reorganization or transfer. A minimal capacity volume $V_j^{(t)}$ is required for resources allocated to assets for termination at R_j.
- Resource ($i \in \mathcal{I}$): The identical purpose material, commodity or labor that needs to be allocated to asset. Its maximum capacity volume ($V_i^{(max)}$) can be restored later.
- Job: Virtual variable to help decouple entangled operational constraints for task processing of asset. The various jobs of resource are as follows:
  - **Func**tion: Meet asset's functional task schedule.
  - **Stan**dby: Handle any demand surge.
  - **Rest**oration: It costs D_i length of time to restore capacity volume.
  - **Term**inate: Meet asset's termination task schedule.
  - **Unav**ailable: Like annual leave for labor or regular check for machines.

B. Illustrated Example

We consider a case of dual-resource asset and illustrate the problem in Fig. 2. The operational task schedule of asset is given in Fig. 2(a). The asset is scheduled to be utilised according to demand (F) and will be terminated when no longer required (T). A feasible allocation plan that fulfills the task schedule is provided in Fig. 2(b). In the initial state ($t = 0$), all resources are allocated to the assets to meet the operational schedule. At $t = 2$, Resource-7 starts to undergo restoration and would only be available when $t = 9$.



(a) Operational schedule

(b) Allocation plan for dual-resource asset

Fig. 2. The generalised OCRPP

Utilizing CP techniques, we formulate the OCRPP and illustrate how to use job concepts to manage the constraints on resource capacity over the planning horizon in Fig. 3.

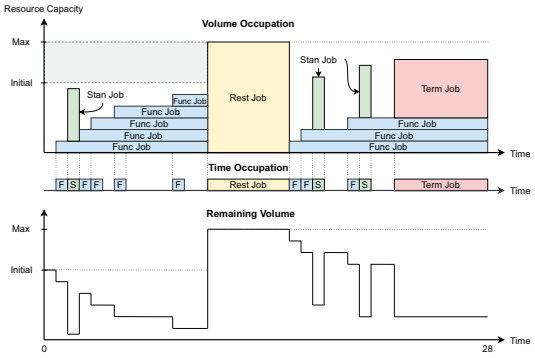


Fig. 3. Illustration of constraints on capacity and time occupation for resource

From the chart depicting the remaining volume of resources(below in Fig. 3), it becomes clear that the *Stan* and *Rest* jobs behave differently comparing to *Func* and *Term* jobs. The *Func* job consumes $V_j^{(f)}$ of resource capacity and the *Term* job will take $V^{(t)}$ instead. While the Stan job depletes renewable volume denoted as $V^{(s)}$, and the *Rest* job has the capacity to completely replenish the resource.

The CP functions are designed to enforce the restriction that each resource can only handle one task concurrently. This is

achieved by considering the time occupation shown in Fig. 3, which guarantees sufficient resource capacity for the specified jobs.

C. Decision variables

We define the following variables for the OCRPP, where i represents the resource in a set of resources $\mathcal{I}$ and j represents the asset in a set of assets $\mathcal{J}$. Here t denotes the time period of the whole planning horizon T, where $t = 0$ refers to the initial condition.

As there are various states of the resource i, the following decision variables are defined, allowing us to determine the assignment of the resource i:

- $x_{i,j,t}$: resource i is allocated to asset j (or not) at time t,
- $r_{i,m}$: restore resource i (or not) for m-th time,
- $b_{i,m}$: start time of m-th restoration period for resource i,
- $s_{i,t}$: resource i standby at time period t or not.

D. Objective

Among types of goals from asset management, we use total resource shift cost as objective value (1) for OCRPP to minimize:

$$\min Z = \sum_{i\in\mathcal{I}, j\in\mathcal{J}, 0\le t\le T} w_{i,j,t} \tag{1}$$

$$w_{i,j,t} \triangleq x_{i,j,t-1} \oplus x_{i,j,t}, \quad 0 < t \le T, \forall i, j \tag{2}$$

where $w_{i,j,t}$ is the resource shifting indicator defined in (2) to tell whether the allocation plan for resource i on asset j at time t is different from the allocation plan at $t-1$.

Some operational constraints can be exceedingly difficult to satisfy, hence, it is usually acceptable for manager to tolerate minor deviations from these constraints [16]. Hence, we introduce these "soft" constraints as penalties and incorporate them into the objective function as (3),

$$\min Z + c_r Z_r + c_s Z_s + \cdots \tag{3}$$

where Z_r, Z_s are the penalties for restoration (16) and lack of standby resources (17) respectively, and c_o, c_s are the unit cost for these objectives. It can be extended with more penalties if needed.

E. Constraints

The constraints in the model encompass the following three aspects: 1) resource occupation, 2) operational constraints, and 3) "soft" constraints as penalties.

1) Resource occupation: To ensure that the volume occupation of each resource capacity for different jobs does not exceed its maximum capacity $V_i^{(max)}$, we employ the `cumulative` function as shown in (4).

$$\texttt{cumulative}(\mathbf{b}_i, \mathbf{dv}_i, \mathbf{p}_i, \mathbf{g}_i, V_i^{(max)}), \quad \forall i \in \mathcal{I} \tag{4}$$

This function effectively limits the cumulative volume occupations of any resource i at any time, where $V_i^{(max)}$ refers to the maximum capacity volume and the meanings and values of the first 4 positional vector parameters are listed in table I . The *begin*, *duration* and *demand* describe the basic 2-dimensional resource occupation of different jobs, while *enabler* decides the assignment. In addition, $\mathbf{dt}_i$ also represents the *duration*, and it is required by (9).

TABLE I
PARAMETERS FOR OCCUPATION CONSTRAINTS OF RESOURCE i

Job Name[①]	Begin $\mathbf{b}_i$	Duration $\mathbf{dv}_i$[②]	Duration $\mathbf{dt}_i$[③]	Demand $\mathbf{g}_i$	Enabler $\mathbf{p}_i$
Func	t	$d_{i,j,t,m}$	1	$V_j^{(f)}$	$p_{i,j,t,m}$
Stan	t	1	1	$V^{(s)}$	$s_{i,t}$
Rest	$b_{i,m}$	D_i	D_i	$V^{(max)}$	$r_{i,m}$
Term	R_j	$T - R_j$	$T - R_j$	$V^{(t)}$	x_{i,j,R_j}

[①] *Unav* jobs are predetermined in the schedule as parameters.
[②] For `cumulative` only
[③] For `no_overlap` only

The additional undefined elements $d_{i,j,t,m}$, $p_{i,j,t,m}$ are introduced to handle *Func* jobs that appear between $(m-1)$-th and m-th restoration of resource i.

The *Func* job consumes resource volume without recovery, and this occupation effect will end at the subsequent Rest job, which can refill the resource with volume $V_i^{(max)}$. Therefore, the duration of *Func* job can be described as (5).

$$d_{i,j,t,m} = \begin{cases} b_{i,m} - t & \text{if enabled} \\ max\{T, R_j \cdot x_{i,j,t}\} & \text{otherwise.} \end{cases} \tag{5}$$

The enabler $p_{i,j,t,m}$ is to indicate *Func* job's presence. It has strong relation to decision variables $x_{i,j,t}$ and $r_{i,m-1}$:

$$p_{i,j,t,m} = x_{i,j,t} \cdot r_{i,m-1}, \tag{6}$$

which can be linearized as (7).

$$\begin{cases} p_{i,j,t,m} \le r_{i,m-1}, \quad m > 0 \\ \sum_{m=0}^{T/D_i} p_{i,j,t,m} = x_{i,j,t} \end{cases} \quad \forall i \in \mathcal{I}, j \in \mathcal{J} \tag{7}$$

Meantime, restoration enabler should adhere to (8) to reduce systematic symmetry, and set $r_{i,0} = 1$.

$$r_{i,m} \le r_{i,m-1}, \quad 1 \le m \le T/D_i, \forall i \in \mathcal{I} \tag{8}$$

On the other hand, to guarantee that only one job can be assigned to each resource i at any time, we utilize the `no_overlap` function as illustrated in (9).

$$\texttt{no\_overlap}(\mathbf{b}_i, \mathbf{dt}_i, \mathbf{p}_i), \quad \forall i \in \mathcal{I} \tag{9}$$

This function ensures that there are no overlapping of time occupations for different jobs assigned to the same resource. It is important to note that the duration $\mathbf{dt}$ used in the `no_overlap` function differs slightly from $\mathbf{dv}$ that in the `cumulative` function as listed in table I.

2) Operational constraints: Operational constraints are imposed to manage the relationship between resource and asset, which are listed in eqs. (10)–(15).

$$\begin{cases} \sum_{j\in\mathcal{J}} x_{i,j,t} \le 1 & \forall i \in \mathcal{I} \quad (10) \\ \sum_{i\in\mathcal{I}} x_{i,j,t} \le C_j & \forall j \in \mathcal{J} \quad (11) \\ \sum_{i\in\mathcal{I}} x_{i,j,t} \ge C_j \cdot y_{j,t} & \forall j,t \quad (12) \\ \sum_{i\in\mathcal{I}} x_{i,j,t} = C_j & t \ge R_j, \forall j \quad (13) \\ w_{i,j,t} = 0 & t > R_j, \forall i,j \quad (14) \\ x_{i,j,t} = 0 & t \in \mathcal{X}_i \cup \mathcal{R}_i, \forall i,j \quad (15) \end{cases}$$

Eq. (10) indicates each resource can be allocated to at most 1 asset at any time, while (11) means the number of assigned resources for a certain asset should not exceed its configuration capacity. Assets are required to be fully allocated with resources if performing any functional tasks as (12), where $y_{j,t}$ indicates if asset j has functional task. Eq. (13) is for similar Termination tasks. Eq. (14) prohibits any further allocation change after asset termination, while (15) constraints no resource allocation during the unavailable period, including restoration and unavailable, where $\mathcal{X}_i$ and $\mathcal{R}_i$ represent when resource i is unavailable and in-restoration respectively.

3) "Soft" constraints as penalties: "Soft" constraints refer to those that are challenging to satisfy but can be treated as penalties in objective (3).

1. Penalty for exceeding the maximum number M_i of restoration for each resource as (16).

$$Z_r = \sum_{i\in\mathcal{I}} max\left\{0, \sum_{m=1}^{T/D_i} r_{i,m} - M_i\right\} \quad (16)$$

2. Penalty for insufficient number S_t of standby resource in each time period as (17).

$$Z_s = \sum_{t=1}^{T} max\left\{0, S_t - \sum_{i\in\mathcal{I}} s_{i,t}\right\} \quad (17)$$

III. Methodology

Directly solving the OCRPP model can be time-consuming and yield low-quality solutions due to the significant number of systematic symmetries with regard to the inter-changeable matching for resources and assets. We propose a meta-heuristic solving framework Hierarchical Allocation Optimizer (HAO) as shown in fig. 1 to address this challenge.

A. Phase 1: Active resource selection

The current model exhibits numerous systematic symmetries caused by the specific resource allocation plan, where many resources are interchangeable for functional tasks of assets without affecting the objective value. To address this, we introduce the concept of "active" resource, as defined in (18). These resources are selected to be candidates for certain operation jobs. However, their allocation to each specific asset is not explicitly determined. By selecting only the necessary number of "active" resources, the number of decision variables can be dramatically reduced because j is not considered in this phase.

$$\hat{x}_{i,t} \triangleq \sum_{j\in\mathcal{J}} x_{i,j,t}, \quad \forall i,t, \quad (18)$$

Given the asset task schedule illustrated in Fig. 2 (a) for example, the required number of "active" resources per time period is determined in advance. This enables us to calculate the required number of resources to support the asset tasks at every time period.

However, the new challenge is that the shift count cannot be determined directly. Hence, we estimate the shift count for any asset j at a given time t using:

$$\hat{w}_{j,t} = q_{j,t-1} + q_{j,t} - 2 \cdot a_{j,t}, \quad t > 0, \forall j, \quad (19)$$

where $\hat{w}_{j,t} \triangleq \sum_{i\in\mathcal{I}} w_{i,j,t}$ and $q_{j,t}$ represent the required total number of resources for asset j to be allocated at time t, as determined by (20), while $a_{j,t}$ denotes the number of resources attached to asset j in both time t and $t-1$.

$$q_{j,t} = \begin{cases} C_j & y_{j,t} = 1 \text{ or } t \ge R_j \\ \sum_{i\in\mathcal{I}} x_{i,j,t} & \text{otherwise} \end{cases} \quad \forall j \in \mathcal{J}. \quad (20)$$

Then, we can determine the upper bound of estimation indicator $a_{j,t}$ as in eqs. (21)–(23):

$$\begin{cases} \sum_{\substack{\forall j\in\mathcal{J} \\ y_{j,t-1}=0 \wedge y_{j,t}=1}} a_{j,t} \le \sum_{j\in\mathcal{J}} max\left\{C_j, \sum_{i\in\mathcal{I}} x_{i,j,t-1}\right\}, & (21) \\ \sum_{\substack{\forall j\in\mathcal{J} \\ y_{j,t-1}=1 \wedge y_{j,t}=0}} a_{j,t} \le \sum_{j\in\mathcal{J}} max\left\{C_j, \sum_{i\in\mathcal{I}} x_{i,j,t}\right\}, & (22) \\ \sum_{\substack{\forall j\in\mathcal{J} \\ y_{j,t-1}=1 \wedge y_{j,t}=1}} a_{j,t} \le \sum_{j\in\mathcal{J}} C_j. & (23) \end{cases}$$

By utilizing eqs. (19) and (21), we can calculate the lower bound of the shift count, as expressed in (24).

$$\underline{Z} = \sum_{\forall j,t} \hat{w}_{j,t} \quad (24)$$

This lower bound can then be used as a substitution for the variable Z in the objective function (3) to facilitate the active resource selection process.

B. Phase 2: Active resource allocation

After the active resource selection phase, we can proceed to specify the resource allocation while keeping the values of other decision variables fixed. This can be done using (25),

$$\begin{cases} x_{i,j,t} = x^*_{i,j,t} & \text{if } t \notin \mathcal{F}_j, \forall j,t \\ b_{i,m} = b^*_{i,m} & \\ s_{i,t} = s^*_{i,t} & \forall t \end{cases}, \quad \forall i \quad (25)$$

where $x^*_{i,j,t}$, $b^*_{i,m}$ and $s^*_{i,t}$ represent the solution values obtained from active resource selection phase.

Next, the resource allocation phase involves solving a smaller Mixed-Integer Programming (MIP) problem that includes the new constraint in eq. (26). It ensures that active resources are allocated to specific asset for operational tasks,

$$\sum_{j \in \mathcal{J}} x_{i,j,t} = 1, \quad i \in \hat{\mathcal{I}}_t \tag{26}$$

where $\hat{\mathcal{I}}_t$ is the active resource set at t.

The objective of the resource allocation phase remains the same as stated in (3), with the constraints of eqs. (10), (25) and (26).

C. Phase 3: Fine tuning

After warming up with the above two phases, we can obtain feasible solution(s), taking one of them as $\mathbf{s}^*$ without loss of generalization. This solution may not be globally optimized since the shift count estimation serves only as a guideline, and most of the decision variable values are fixed during the active resource allocation phase.

To further refine the solution, we will use $\mathbf{s}^*$ as the starting point, where lower bound is hinted, to start for a new global search for the original OCRPP.

IV. Experiments and Applications

A. Experimental setup

The proposed HAO follows a hierarchical framework, allowing us to utilize different solvers for each solving phase. In our experiments, we have adopted Google OR-Tools as the solver wrapper, which can employ different backend solvers. The experiments were conducted on a workstation with a 16-core CPU (11th Gen Intel i9-11900) using Python 3.9 and OR-Tools 9.0.

B. Results and analysis

For the given simple scenario depicted in Fig. 2, its problem scale is $|\mathcal{J}| \times |\mathcal{I}| \times T = 5 \times 10 \times 28$, and the task schedule is depicted in Fig. 2 (a). The manual solution for this 10-resource scenario is presented in Fig. 2 (b), while the solution obtained from the HAO is illustrated in Fig. 4, which is one of the optimal solutions.

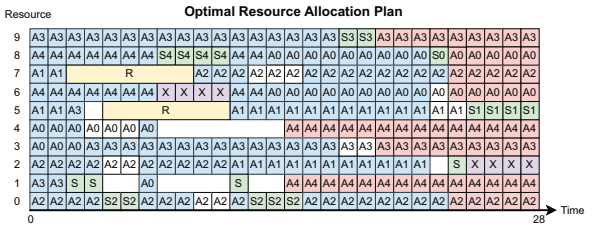


Fig. 4. Resource allocation plan for 10-Resource scenario using HAO

In addition to the manual solution, we compared HAO with a rule-based heuristic, and customized Genetic Algorithm (GA). The rule-based method based on the experience from experts' manual work. The customized GA uses specific encoding and decoding procedures, along with customized crossover and mutation operations. The individual consists of resource and asset parts, encoded with values representing resource states and asset. The decoding process calculates fitness values by assigning resources and assets based on the encodes. Customized operations include crossover between parents and shuffling asset indices for mutation.

Using the same data, the best results from multiple rounds for the rule-based, GA, and HAO methods are listed in table II, where the objectives selected for evaluation, namely Z, Z_r, and Z_s, refer to resource shift counts, exceeding restorations and lack of standby resource respectively from (3).

TABLE II
Experiment results comparison

Method	Z	Z_r	Z_s	std(obj)[①]	Running time
Manual	*56*	4	18	-	2 days
Heuristic[②]	222	*3*	9	152193.58	**0.06 s**
Meta-heuristic[③]	116	5	**4**	167183.61	850.41 s
HAO	**25**	**2**	*8*	**0**	59.42 s

[①] standard deviation of objective (3)
[②] rule-based strategy + random sample
[③] customized GA with 10000 of population for 1000 iterations

It is evident that the rule-based method yields immediate results but with a high number of resource shift counts. The manual solution considerably reduces this count, yet it requires significant human effort to accommodate all constraints. Unfortunately, the resulting solution leads to high penalty values for Stan and Rest jobs, potentially due to human errors. On the other hand, the GA produces a lower count of shifts compared to the rule-based. However, its performance is less stable compared to the rule-based, and not comparable with the manual solution or HAO.

In contrast, the proposed HAO method outperforms all other approaches, particularly in minimizing shift counts while maintaining an acceptable runtime. Additionally, it always produces optimal solutions thus the performance is highly stable in this example.

To validate the effectiveness of the proposed HAO, we conduct a bunch of ablation studies with the following variants in table III.

TABLE III
HAO variants for ablation study

Name	Phase 1	Phase 2	Phase 3
HAO-001	✗	✗	✓[a]
HAO-110	✓	✓	✗
HAO-111	✓	✓	✓

[a] no warm up

Through the generated instances at different problem scales, experiments were carried out for multiple rounds under each setting group. The average results in terms of $\bar{Z}$, $\bar{Z}_r$, and $\bar{Z}_s$

are listed in table IV. Each experiment runs within 1 hour for the entire solving process.

TABLE IV
RESULT FOR ABLATION STUDY

Exp.	Scale ($\lvert\mathcal{J}\rvert \times \lvert\mathcal{I}\rvert \times T$)	Method	$\bar{Z}$	$\bar{Z}_r$	$\bar{Z}_s$
1	$16 \times 48 \times 24$	HAO-001	**96.00**	**0.00**	**0.00**
		HAO-110	249.33	0.33	**0.00**
		HAO-111	101.33	0.33	**0.00**
2	$16 \times 96 \times 24$	HAO-001	628.33	5.00	0.33
		HAO-110	**614.67**	5.00	**0.00**
		HAO-111	635.33	**4.33**	**0.00**
3	$16 \times 48 \times 48$	HAO-001	164.00	**0.00**	**0.00**
		HAO-110	268.67	**0.00**	**0.00**
		HAO-111	**160.33**	**0.00**	**0.00**
4	$16 \times 96 \times 48$	HAO-001	589.00	6.00	**0.00**
		HAO-110	610.00	3.67	**0.00**
		HAO-111	**534.33**	**3.33**	**0.00**

The results reveal that the full-featured HAO-111 outperforms the ablated variants, particularly under relatively larger problem scales. This underscores the value of investing time in a warm-up procedure before engaging in global searching. However, warm-up phases only leads to inconsistent performance due to the theoretical nature of the estimation in phase 1 and the discrete distribution of the search space. This indicates that lower bound regions do not always guarantee better solutions. Consequently, the fine-tuning phase should be adopted in conjunction with the warm-up phase. Furthermore, for smaller cases (Exp. 1), HAO-001 exhibits better performance as it starts the global searching at the beginning, but HAO-111 has relative less time for global searching.

In practical real-world applications, when the problem scale remains relatively small, direct employment of global searching (HAO-001) is recommended. The comprehensive HAO-111 also demonstrates strong performance. As real-world problems tend to be large, opting for the full-featured HAO-111 is a judicious decision.

V. CONCLUSION

This paper addresses resource planning problem prevailing in diverse industries, such as engine optimization in airlines, vehicle fleet management in logistic companies, labor rostering, and so forth. We abstract common operational constraints and introduce the job concept to structure resource and asset coordination, proposed the OCRPP model. To solve this, we present the HAO meta-heuristic framework, encompassing 3 distinct phases. Through comparative experiments involving manual, heuristic and general meta-heuristic. HAO demonstrates remarkable performance with acceptable runtime. The ablation studies highlight the role of warm-up and fine-tuning steps within HAO.

In the future, we aim to enhance the HAO by integrating Deep Reinforcement Learning to enhance the decision making for active resource selection. Furthermore, we plan to explore real-time dynamic approaches to generate robust plan that can adapt to changing requirements.

REFERENCES

[1] Y. Naveh, Y. Richter, Y. Altshuler, D. L. Gresh, and D. P. Connors, "Workforce optimization: Identification and assignment of professional workers using constraint programming," *IBM Journal of Research and Development*, vol. 51, no. 3.4, pp. 263–279, May 2007.

[2] A. Imai, H. C. Chen, E. Nishimura, and S. Papadimitriou, "The simultaneous berth and quay crane allocation problem," *Transportation Research Part E: Logistics and Transportation Review*, vol. 44, no. 5, pp. 900–920, Sep. 2008.

[3] K. Ertogral and F. S. Öztürk, "An integrated production scheduling and workforce capacity planning model for the maintenance and repair operations in airline industry," *Computers & Industrial Engineering*, vol. 127, pp. 832–840, Jan. 2019.

[4] L. Zhou, Z. Liang, C.-A. Chou, and W. A. Chaovalitwongse, "Airline planning and scheduling: Models and solution methodologies," *Frontiers of Engineering Management*, vol. 7, no. 1, pp. 1–26, Mar. 2020.

[5] A. Chun and T. Suen, "Engineering Works Scheduling for Hong Kong's Rail Network," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 28, no. 2, pp. 2890–2897, Jul. 2014.

[6] Z. Feng, C. Cao, A. Mostafizi, H. Wang, and X. Chang, "Uncertain demand based integrated optimisation for train timetabling and coupling on the high-speed rail network," *International Journal of Production Research*, vol. 61, no. 5, pp. 1532–1555, Mar. 2023.

[7] A. Karageorgos, N. Mehandjiev, G. Weichhart, and A. Hämmerle, "Agent-based optimisation of logistics and production planning," *Engineering Applications of Artificial Intelligence*, vol. 16, no. 4, pp. 335–348, Jun. 2003.

[8] L. W. Krakow, L. Rabiet, Y. Zou, G. Iooss, E. K. P. Chong, and S. Rajopadhye, "Optimizing Dynamic Resource Allocation," *Procedia Computer Science*, vol. 29, pp. 1277–1288, Jan. 2014.

[9] L. B. Toktay and R. Uzsoy, "A capacity allocation problem with integer side constraints," *European Journal of Operational Research*, vol. 109, no. 1, pp. 170–182, Aug. 1998.

[10] F. T. S. Chan, T. C. Wong, and L. Y. Chan, "Flexible job-shop scheduling problem under resource constraints," *International Journal of Production Research*, vol. 44, no. 11, pp. 2071–2089, Jun. 2006.

[11] X. Jiang, Z. Tian, W. Liu, Y. Suo, K. Chen, X. Xu, and Z. Li, "Energy-efficient scheduling of flexible job shops with complex processes: A case study for the aerospace industry complex components in China," *Journal of Industrial Information Integration*, vol. 27, p. 100293, May 2022.

[12] J. N. Hooker, "Logic, Optimization, and Constraint Programming," *INFORMS Journal on Computing*, vol. 14, no. 4, pp. 295–321, Nov. 2002.

[13] J. Van den Bergh, P. De Bruecker, J. Beliën, L. De Boeck, and E. Demeulemeester, "A three-stage approach for aircraft line maintenance personnel rostering using MIP, discrete event simulation and DEA," *Expert Systems with Applications*, vol. 40, no. 7, pp. 2659–2668, Jun. 2013.

[14] B. Abbasi, T. Babaei, Z. Hosseinifard, K. Smith-Miles, and M. Dehghani, "Predicting solutions of large-scale optimization problems via machine learning: A case study in blood supply chain management," *Computers & Operations Research*, vol. 119, p. 104941, Jul. 2020.

[15] Y. Fan, J. Ding, H. Liu, Y. Wang, and J. Long, "Large-scale multimodal transportation network models and algorithms-Part I: The combined mode split and traffic assignment problem," *Transportation Research Part E: Logistics and Transportation Review*, vol. 164, p. 102832, Aug. 2022.

[16] J. Wang, T. Yu, and W. Wang, "Integrating Analytic Hierarchy Process and Genetic Algorithm for Aircraft Engine Maintenance Scheduling Problem," in *Proceedings of the 6th CIRP-Sponsored International Conference on Digital Enterprise Technology*, ser. Advances in Intelligent and Soft Computing, G. Q. Huang, K. L. Mak, and P. G. Maropoulos, Eds. Berlin, Heidelberg: Springer, 2010, pp. 897–915.