IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 3 March 2024, accepted 17 March 2024, date of publication 25 March 2024, date of current version 29 March 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3380160

RESEARCH ARTICLE

An Improved Saturation Degree-Based Constructive Heuristic for Master Surgery Scheduling Problem: Case Study

MOHAMAD KHAIRULAMIRIN MD. RAZALI[1], ABDUL HADI ABD RAHMAN[2], MASRI AYOB[2], RAZMAN JARMIN[3], LIU CHIAN YONG[4], MUHAMMAD MAAYA[4], AZARINAH IZAHAM[4], AND RAHA ABDUL RAHMAN[4]

[1]Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia, Bangi, Selangor 43600, Malaysia
[2]Center for Artificial Intelligence Technology, Universiti Kebangsaan Malaysia, Bangi, Selangor 43600, Malaysia
[3]Faculty of Medicine, Universiti Kebangsaan Malaysia Medical Centre, Kuala Lumpur 56000, Malaysia
[4]Department of Anaesthesiology and Intensive Care, Faculty of Medicine, Universiti Kebangsaan Malaysia, Kuala Lumpur 56000, Malaysia

Corresponding author: Abdul Hadi Abd Rahman (abdulhadi@ukm.edu.my)

This work was supported by the Ministry of Higher Education (MoHE) Malaysia, through the Transdisciplinary Research Grant Scheme under Grant TRGS/1/2019/UKM/01/4/2.

ABSTRACT The Master Surgery Scheduling Problem (MSSP) can be described as a timetabling problem involving assigning surgery groups to operating theatre (OT) time slots. Previous MSSP optimization models considered throughput, waiting measures, resource utilization, costs, and schedule assignment objectives, but have overlooked consecutive days assignment preferences and surgical equipment-sharing limitations. Furthermore, previous works utilize greedy constructive heuristics to produce solutions, which increases quality but decreases feasibility. Our prior study demonstrated that the saturation degree heuristic enhances feasibility by considering assignment difficulty during event selection. However, its impact on solution quality remained unexplored. Therefore, this study proposes an improved saturation degree-based constructive heuristic that integrates objective function value for event selection to increase both quality and feasibility. The algorithm sorts surgery groups by unit scores, prioritizing higher assignment difficulty and objective value. The highest-scoring group is assigned to its feasible slot with the highest slot score. If no feasible slots exist, the repair mechanism vacates the slot with the highest swap score, which prioritizes lower assignment difficulty and objective value. A new mathematical model is also formulated, incorporating novel objectives regarding consecutive days assignment preference and surgical equipment-sharing limitations. Using real-world data from Hospital Canselor Tuanku Muhriz, the proposed algorithm is evaluated considering repair mechanism usage for feasibility and objective function value for quality. The algorithm is benchmarked against greedy, random, regret-based, and saturation degree-based constructive heuristics. Our algorithm achieved a 14.63% improvement in feasibility compared to the original variant. Its objective function value is over two times better than the closest competitor and 2.6 times superior to the original variant. Comparison with the hospital's actual plan demonstrates competitive objective function value and a more balanced waiting time distribution among surgical groups. Our study showcases that a saturation degree-based constructive heuristic considering objective function value has increased solution quality while maintaining feasibility.

INDEX TERMS Constructive heuristic, graph colouring heuristic, healthcare management, master surgery scheduling, operating theatre planning, solution feasibility.

The associate editor coordinating the review of this manuscript and approving it for publication was Gustavo Olague.

I. INTRODUCTION

The surgery scheduling problem involves selecting and sequencing surgeries and allocating resources for those

surgeries [1]. The problem can be divided into three levels: strategic, tactical, and operational [2], [3]. The capacity allocation problem (strategic) involves the decisions on the number of operating theatres (OT) to be made available and their allocation to surgery groups such as medical specialties or surgeons [4]. Meanwhile, the Master Surgery Scheduling Problem (MSSP) is a tactical problem that produces a master surgery schedule, also called a master plan, which defines the assignment of surgery groups to OT time slots [2]. Finally, the operational level determines individual surgeries' time slots and sequence. Good OT planning leads to an efficient hospital in terms of operation management and financial management [5], [6]. Hence, it is important to have an effective master plan.

One of the main issues in MSSP is the involvement of numerous stakeholders, each with different and perhaps conflicting objectives [7]. The surgery scheduling problem can be represented mathematically as a combinatorial optimization model for distributing hospital resources based on specific objectives [3]. Among the common objectives are maximizing the number of surgeries [8], [9], [10], minimizing OT idleness [11], [12], [13] and the number of patients in downstream resources [11], [14], [15]. However, no previous works have considered the objectives for the case study hospital, Hospital Canselor Tuanku Muhriz (HCTM), which includes assignment preference and resource-sharing limitations.

Assignment preference refers to the specific preferences of surgery groups for certain OT slots or having repeating patterns in the schedule for high predictability. The model proposed by Penn et al. [16] minimizes surgeons in non-preferred slots and maximizes surgeons in the same slot each week or consecutive slots. Other works, such as Oliveira et al. [17] and Britt et al. [11], address similar objectives. However, these models do not consider preferences for scheduling a surgery group on two consecutive days. In addition, while existing models have included objectives related to various resources such as OTs and upstream and downstream resources [18], limitations in resource sharing that constrain the daily assignment of surgery groups based on equipment availability have not been incorporated as objectives in any MSSP model.

Solution robustness has been linked to a decline in solution quality [19], prompting an exploration into whether increased solution feasibility has a similar adverse impact. Most prior constructive heuristics that address the MSSP employ the greedy approach [5], [20], [21], [22], which can maximize solution quality at the cost of other variables [23]. Although alternative approaches such as stochastic methods [24] and regret-based strategies [9] have mitigated this trade-off, they still overlook solution feasibility during generation. To overcome this issue, Razali et al. [25] introduced a saturation degree-based constructive heuristic that enhances feasibility but neglects solution quality. This trend emphasizes the tendency of prior heuristics to prioritize either feasibility or quality, calling for methods to improve both aspects.

The main motivation of this paper is to examine the relationship between solution feasibility and solution quality within the framework of a constructive heuristic to generate high-quality, feasible initial solutions for the MSSP. This study focuses on using constructive heuristics to generate initial solutions for the MSSP. Constructive heuristics can produce high-quality initial solutions, which may reduce the computational effort required to find the optimal solution [26]. The objectives of the study can be summarized as follows:

- To formulate a new mathematical model tailored for the case study hospital by incorporating variables not previously considered.
- To develop a novel constructive heuristic based on saturation degree, considering both solution feasibility and quality, and investigate the interaction between these factors.

This study formulated a new variation of the MSSP mathematical model with a novel objective function that includes assignment preference and resource-sharing limitations. Besides, an improved version of the saturation degree-based constructive heuristic from Razali et al. [25] is proposed to consider both solution feasibility and quality. A constructive heuristic for the MSSP involves an iterative process of selecting a surgery group and an OT slot for the group. The algorithm proposed by Razali et al. [25] selects the surgery group with the fewest feasible OT slots and assigns it to the OT slot with the fewest possible assignments to enhance solution feasibility. In contrast, our proposed algorithm considers a composite value, incorporating the number of feasible slots or surgery groups and potential objective function value for selection, aiming for higher quality solutions while maintaining feasibility.

We hypothesize that if consideration of objective function value in event sorting can affect the solution quality, then a constructive heuristic incorporating saturation degree heuristic based on the objective function value can increase both solution feasibility and quality. The hypothesis is tested by comparing the feasibility and quality of master plans generated using the proposed approach to greedy, random, regret-based, and the original saturation degree-based constructive heuristics. Solution feasibility and quality are measured by repair mechanism usage and objective function value, respectively. Furthermore, a simulation is carried out to assess the operational performance of each master plan.

The remaining sections of this work are structured as follows: Section II provides knowledge about the related works in the problem domain. Next, the mathematical model for the combined capacity allocation and MSSP based on the case study hospital, HCTM, is presented in Section III. Section IV describes our proposed improved saturation degree-based constructive heuristic. Sections V, VI, and VII provide an in-depth discussion of the evaluation results and statistical analysis, focusing on comparing constructive heuristics, improvement algorithms, and performance as

initial solutions. Finally, the conclusion and future works are described in Section VIII.

II. RELATED WORKS

When constructing solutions for the MSSP, the key component of optimization is the objectives. The objectives can be classified into eight categories: throughput, waiting measure, patient score, OT utilization, emergency capacity, costs, schedule assignment, and upstream and downstream resources [18]. Throughput objectives include maximizing the number of surgeries performed [8], [9], [10], [27], minimizing surgical cancellations [8], [28], and minimizing surgical postponements [29]. The waiting measure objectives include minimizing waiting time [29], [30] and associated costs [22]. Patient score objectives involve maximizing scheduled patient scores [14], [31], [32]. Each surgery or patient is assigned a score, which varies between models. In [31], the score for each surgery is determined based on its proximity to the due date, where a higher score indicates greater urgency. Aringhieri et al. [14] calculate scores by multiplying the surgery priority level by the time between diagnosis and surgery. Objectives related to OT utilization include maximizing utilization [28], [33], minimizing overtime [22], [24], and minimizing idleness [11], [12], [13], [22], [24], [34]. Emergency capacity objectives, a less common category, involve minimizing OT time reserved for emergency surgeries [9]. Cost-related objectives include minimizing hospitalization [35], machine [11], and overtime costs [35], [36], as well as minimizing losses [29] and maximizing profit [36]. Schedule assignment objectives focus on the master plan, aiming to maximize staff preferences [16], [37] or minimize assignment variation [11]. The upstream and downstream resources category includes objectives related to the utilization of wards and ICU beds. Previous models aimed at maximizing patients in beds [11], [13], [38], minimizing bed shortages [39], and minimizing variation in bed usage [13], [33], [38].

The MSSP is a combinatorial optimization problem that can be solved using exact or approximate methods. Exact methods guarantee optimal solutions within a finite time [40], but they have scaling limitations, making them unsuitable for more complex problems [41]. Approximate methods, including heuristic and approximation algorithms, can find good solutions to more complex problems within a reasonable timeframe [42]. Heuristic algorithms are categorized into low-level heuristics (commonly referred to as heuristics), metaheuristics, and hyper-heuristics [43]. Low-level heuristics modify the solution directly, whereas metaheuristics utilize one or more low-level heuristics to guide the search of the solution space [41]. A higher-level technique known as hyper-heuristics integrates multiple low-level heuristics into a framework that searches the heuristic space [44]. These heuristics can be divided into constructive and local search (or improvement) heuristics [45]. Constructive heuristics build a complete solution incrementally, whereas local search heuristics start with a complete solution and make small changes to find improvement.

Within the MSSP domain, constructive heuristics have been employed to obtain initial solutions for the MSSP before being improved by more sophisticated methods. Spratt and Kozan [9] introduced a regret-based constructive heuristic to generate an initial solution for the Simulated Annealing (SA) and Tabu Search hyper-heuristic algorithms. The regret-based approach is applied to select an OT for a chosen surgeon-specialty set in their algorithm. For each feasible OT, the regret value is calculated as the difference between the highest and the second-highest number of patients that could be scheduled. The OT with the largest regret is then chosen. Their approach can produce a master plan with the most scheduled patients. However, solution feasibility cannot be guaranteed, particularly during the OT selection for the next surgeon-specialty set, given that the greedy-based set selection is employed. Mashkani et al. [5] developed two dynamic programming-based heuristic algorithms that prioritize patients and specialties with the highest total profit. Their approach is characterized by its greedy nature, and solution feasibility is not considered during solution construction. Dellaert and Jeunet [24] utilize a constructive heuristic to generate an initial solution for their VNS algorithm. In their constructive heuristic, a day is chosen randomly with a higher probability of selecting days with high under-utilization of OT resources. Then, a patient category with a lower OT resource target deviation on the selected day has a higher chance of being chosen. Their approach introduces stochasticity, departing from the reliance on a purely greedy method. However, like the previous methods, it does not guarantee solution feasibility. Razali et al. [25] implemented a saturation degree-based constructive heuristic that prioritizes surgical units and OTs based on constraint violation to enhance solution feasibility. However, the construction of the solution did not consider solution quality, leading to a suboptimal outcome in terms of solution quality. Similar trends in the popularity of adopting a greedy approach and overlooking solution feasibility have been observed in other works that schedule individual patients through constructive heuristics [20], [21], [22]. This trend highlights that prior constructive heuristics typically focus on either solution quality or feasibility.

Various metaheuristic algorithms were employed to solve the MSSP. Schneider et al. [33] utilized SA and conducted a performance comparison with an exact method, Mixed Integer Linear Programming (MILP). Their findings indicated that MILP surpassed SA, and incorporating SA after MILP led to further improvements. Lu et al. [36] addressed a multi-objective problem using an improved Non-dominated Sorting Genetic Algorithm. Their algorithm introduced a novel population initialization process, where constraints are imposed to ensure chromosome feasibility. When solving their three-objective problem, the proposed algorithm outperformed a goal programming approach in two objectives. Marchesi & Pacheco [46] applied the Genetic Algorithm

(GA), whereas Britt et al. [11] employed a hybrid VNS-GA to solve the MSSP. Almaneea and Hosny [21] proposed a hybrid Bees Algorithm and SA that demonstrated superiority over the algorithm introduced by Aringhieri et al. [20]. Dellaert and Jeunet [24] tested the applicability of the VNS algorithm in solving the MSSP, comparing random VNS (R-VNS) and first descent VNS (FD-VNS) to an exact solver, IBM ILOG CPLEX. Results indicated that R-VNS performed comparably to the exact solver across instances with varying difficulty, whereas FD-VNS performed worse. R-VNS outperformed CPLEX, especially when using random initial solutions for hard-to-solve problems. Spratt and Kozan [10] hybridized SA and VNS to obtain better solutions for the combined MSSP and SCAP, surpassing a decomposition approach in a case study for an Australian public hospital. Zhu et al. [35] proposed a hybrid GWO-VNS to address all three decision levels of the surgical scheduling problem. The proposed algorithm outperformed non-hybrid GWO, VNS, and another swarm intelligence approach, Particle Swarm Optimization (PSO). On the other hand, only one study has employed hyper-heuristics for solving the MSSP, as demonstrated by Spratt and Kozan [9], who employed a hybridized SA and Tabu Search hyper-heuristic algorithm.

The MSSP shares similarities with the timetabling problem [25], which involves allocating events and resources in space and time [47]. Algorithms designed for addressing timetabling or scheduling problems are considered applicable to solving the MSSP, and a review of such algorithms follows. In addressing the educational timetabling problem, a widely studied timetabling scenario, researchers have extensively employed graph colouring heuristics. These heuristics are utilized to order the events, facilitating the construction of a valid solution [48], [49]. These heuristics include largest enrolment, largest degree [50], largest weighted degree [51], largest colour degree, and saturation degree [52], [53], [54], [55]. Various studies have also explored combinations of these heuristics [56], [57], [58], [59], [60]. Additionally, these heuristics have been integrated into various metaheuristic algorithms [61], [62], [63], [64] and hyper-heuristic frameworks [65], [66], [67]. Constructive heuristics are also widely applied to solve diverse optimization problems, such as the Travelling Salesman Problem [68], flowshop scheduling [69], personnel scheduling [70], staff scheduling [71], and machine assembly scheduling [72].

Additionally, metaheuristics play a crucial role in solving timetabling and scheduling problems. Li et al. [73] hybridized the Artificial Bee Colony (ABC) algorithm with Q-learning to address the permutation flow-shop problem (PFSP). The Q-learning component rewards effective neighbourhood structures and selects the appropriate action for the current state. Compared to state-of-the-art ABC, a Whale Optimization Algorithm, and ABC with random neighbourhood selection, the hybrid approach demonstrated superior performance across all benchmark instances, with a notable advantage in large-scale ones. Zheng and Wang [74] proposed a hybrid Bat Algorithm and 5NS to solve a variant of the PFSP. The algorithm ensures population diversity by classifying it into two types: search-type population and captive population. Each population exhibits different points of the search space and utilizes a different update method. Comparative evaluations against other metaheuristics, including memetic algorithms, GA, and PSO, demonstrated the superior performance of the proposed method. Other metaheuristic algorithms utilized in this domain include GA [75], Harmony Search [76], Anarchic Society Optimization [77], and Intelligent Water Drops algorithm [78].

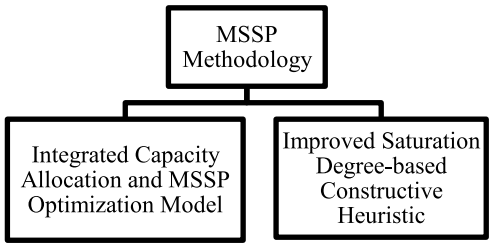

FIGURE 1. Overview of the methodology.

In summary, our study is motivated by two research gaps. Firstly, no existing optimization model considers all objectives of the case study hospital, particularly in terms of consecutive days assignment preference and equipment-sharing limitations. Hence, this study will introduce a new mathematical model that combines existing and new objectives. Secondly, we focus on constructive low-level heuristics among the solution approaches to the MSSP optimization model. Despite the acknowledged effectiveness of metaheuristics and hyper-heuristics in tackling the problem, we focus on constructive heuristics because they generate high-quality solutions more rapidly than the other approaches [79]. We observed a lack of simultaneous consideration of solution quality and feasibility during solution construction. Our study proposes an improved saturation degree-based constructive heuristic that concurrently addresses both factors.

III. PROBLEM DESCRIPTION

The MSSP at the case study hospital, HCTM, is presented in an integrated capacity allocation and MSSP optimization model. The objective function and constraints of the problem are expressed. Master plans are generated using a constructive heuristic inspired by the saturation degree heuristic, later introduced in Section IV. Fig. 1 provides an overview of the study's methodology.

A. PROBLEM AND DEFINITIONS

This study covers the strategic and tactical levels of surgical scheduling, and the optimization is divided into two parts. The first part, called capacity allocation, distributes OT time to medical subspecialties ($j \in J$) and their surgical units ($g \in G$). Afterwards, the surgical units are assigned to suitable OTs and days, forming a master plan. The relationship between medical subspecialties and surgical units can be described as one-to-many. Correspondingly, the surgery personnel

(surgeons, anaesthetists and nurses) responsible for subspecialty j is responsible for all its units ($g \in G_j$).

OT time is divided into full-day slots identified by the indices of the week ($w \in W$), weekly working days ($p \in P$), and OT ($o \in O$). A planning horizon has a $\text{slot}_{\text{total}}$ number of slots calculated by multiplying the number of weeks, weekly working days, and OTs. Each slot is hereafter denoted as $(w, p, o) \in \text{SL}$, where SL is the set of slots in the planning horizon. Each slot is assigned to only one surgical unit, and only a limited amount of OT time is available for scheduling surgeries from the chosen unit ($s \in S_g$). All slots have equal length μ.

The number of slots assigned to each unit in the master plan is limited by the number allocated in the capacity allocation (slot_g). OT time allocation is determined based on the surgery demand and waiting time. Surgery demand is represented by the cumulative expected duration of surgeries on the waiting list, where the expected duration dur_s were given by surgeons. Additionally, the surgery waiting time WT_s can be calculated as the difference between the first planning day and the surgery booking date. The assignment of surgical units to OT time slots is subject to OT suitability and surgeon availability, referred to as feasible slots. Contrarily, slots that are not suitable are called blocked slots. The OTs can only be assigned to surgical units that are suitable for them ($g \in G_O$) according to the OT and surgical unit types. The OTs are classified into three types: *fixed*, *ultra-clean* and *general*. Fixed OTs are reserved for specific units, whereas ultra-clean OTs can be used by any unit without dirty surgeries. Surgical units with ultra-clean surgeries must be assigned to ultra-clean OTs. General OTs can be used by any unit.

Moreover, some units prefer to be assigned to specific OTs ($o \in O_g^{\text{preferred}}$). This preference could be due to the equipment required for their surgeries being stored near or at the OTs. These slots are hereafter called preferred slots. Meanwhile, a surgical unit can only be assigned on days when its surgery team is available ($p \in P_j^{\text{feasible}}$). Surgical units from the same subspecialty can be assigned on day p up to the maximum parallel slots ($\text{slot}_j^{\text{parallel}}$) due to shortages in surgery personnel.

Besides, the master plan assignments consider the availability of movable equipment ($e \in E$). Each equipment has a limited quantity available for usage on each planning day (qty_{ewp}), as it is not practical to equip all OTs due to the high cost [17]. Despite this, the number of surgical units using equipment e assigned on day p is not limited by the available quantity. The equipment can be shared between several OTs since not all surgeries scheduled in a slot use the same equipment. The movement of the equipment should be limited to avoid time loss and surgery blockings. Therefore, the model aims to assign as few surgical units that use the same equipment on the same day.

Furthermore, there is high variability in the availability of the equipment, such as equipment breakdown. Therefore, it is preferred that an extra quantity of each piece of equipment ($\text{qty}_e^{\text{extra}}$) is made available on all days to ensure uninterrupted services. Less expensive equipment may have a lot of extra quantities, whereas costly equipment may not be possible to have any extras.

Schedule stability is a desirable feature of a master plan and can reduce operational planning complexity [28]. It could be achieved by minimizing the differences in the assignments. Surgeons from a surgical unit prefer to be assigned to the same OT throughout the planning horizon. Besides, the surgery team will also benefit by reducing the complexity of their schedule. Surgical units should also be assigned on consecutive days so that cancelled surgeries can be moved to the following day if there is available time. Meanwhile, in a multiple-week master plan, assigning the same unit to the same day and OT each week is desirable.

Multiple subspecialties may have clashing resource requirements and should not be assigned on the same day. The clashing units for unit g are defined as $C_g \subseteq G$. Furthermore, the surgical units can be divided into two types based on their resource consumption: *heavy* and *light*. Heavy units ($g \in G_{\text{heavy}}, G_{\text{heavy}} \subset G$) are characterized as having many resource requirements, including requiring anaesthesia. Light units ($g \in G_{\text{light}}, G_{\text{light}} \subset G$) do not require anaesthesia, and hospital stays for their patients are discretional.

On each day, it is preferred that only one OT is assigned to any unit of a subspecialty. However, subspecialties with many surgeries may require more OT time to operate them timely. In this case, having one heavy and one light unit is preferable to two heavy unit slots for the same subspecialty on the same day. Coordinating operations between heavy unit slots will be more difficult.

The case study hospital handles both elective and emergency surgeries. Ultra-clean emergency surgeries had to use elective ultra-clean OTs since the emergency OTs are not suitable for this surgery type. Therefore, one ultra-clean OT should be assigned to a unit with light resource requirements or left unassigned daily to minimize the interruption to the planned elective surgeries. Interruptions to light surgeries are more tolerable as the patients do not require anaesthesia. Table 1 presents the parameters, whereas Table 2 shows the decision variables available in the mathematical model.

B. PROPOSED OPTIMIZATION MODEL FOR CAPACITY ALLOCATION AND MASTER SURGERY SCHEDULING PROBLEM

Based on the problem at the case study hospital, a mathematical model including a novel objective function and general constraints is formulated. The objective function minimizes violations of soft constraints (SC). It comprises eight assignment preference objectives (SC1, 4 – 10) and two resource-sharing limitation objectives (SC2 – 3). The constraints are extracted from various mathematical models, with minor modifications to suit the problem.

TABLE 1. Parameters for the mathematical model.

Parameters	Description
$\theta_{\text{preferred}}$	Weight of preferred slot objective
θ_e	Weight of movable/shared equipment objective
θ_{extra}	Weight of extra equipment quantity objective
$\theta_{\text{consecutive}}$	Weight of consecutive days objective
θ_{same}	Weight of same OT objective
θ_{weekly}	Weight of weekly stability objective
θ_{clashing}	Weight of clashing subspecialty objective
θ_{parallel}	Weight of parallel slots objective
θ_{heavy}	Weight of parallel heavy slots objective
θ_{reserve}	Weight of reserved ultra-clean OT objective
dsp_{gp}	1, if the surgery team of unit g is available on day p; 0, otherwise
y_{go}	1, if operating theatre o is preferred by unit g
z_{gp}	1, if weekly working day p is preferred by unit g
ζ_{ge}	1, if unit g requires equipment e; 0, otherwise
c_{gwpo}	1, if unit g is assigned to OT o on day p of week w and OT o on day $(p-1)$ of week w; 0, otherwise
b_{gwpo}	1, if unit g is assigned to OT o on day p of week w and day $(p-1)$ of week w; 0, otherwise
α_{gwpo}	1, if unit g is assigned to OT o on day p of week w and week $w-1$; 0, otherwise
β_{giwp}	1, if clashing unit i is assigned to day p of week w; 0 otherwise
q_{jwp}	1, if subspecialty j is assigned to more than one operating theatre on day p of week w
h_{jwp}	1, if heavy units from subspecialty j are assigned to more than one operating theatre on day p of week w
ι_g	1, if unit g has a light resource requirement; 0, otherwise
π_o	1, if the operating theatre o is an ultra-clean OT; 0, otherwise
δ_{wpo}	1, if slot (w,p,o) is not assigned to any unit; 0, otherwise

TABLE 2. Decision variables for the mathematical model.

Decision variables	Description
slot_g	Number of slots allocated to unit g in the planning horizon
slot_j	Number of slots allocated to subspecialty j in the planning horizon
x_{gwpo}	1, if unit g is assigned to operating theatre o on day p of week w; 0 otherwise

1) OBJECTIVE FUNCTION

The objective function (1) minimizes violations of ten soft constraints. The soft constraints can be divided into two types: *penalties* and *rewards*. Penalties are given to undesirable violations that do not affect solution feasibility, whereas rewards are awarded when preferred constraints are satisfied. Since the objective function minimizes, penalties increase the objective value, whereas rewards reduce it.

SC1, described in (2), rewards a unit each time it is assigned to its preferred slots (preferred OT or preferred days). Meanwhile, SC2, expressed in (3), penalizes the objective value for each unit assigned to a day exceeding the available quantity of movable equipment [14], [80]. SC3 presented in (4) rewards the objective value for not using all quantities of equipment on each day [14], [80].

Equations (5)-(7) promote schedule stability by rewarding each time a unit is assigned to the same OT in two consecutive days (SC4), or to any OT in two consecutive days (SC5), or to the same slot in two consecutive weeks (SC6) [16]. However, $\text{SC4}-6$ do not incentivize assignments to consecutive days or weeks of more than two. Furthermore, SC4 and SC5 do not consider the blocked days of a surgical unit.

SC7, described in (8), gives out a penalty if a unit is assigned to a day when its clashing unit has been assigned. SC8 and SC9 are concerned with the parallel slots of a unit. Equation (9) sums the penalties handed out each time a unit is assigned to more than one slot a day. Meanwhile, (10) penalizes each time heavy units from the same subspecialty are assigned to multiple slots on a day.

Finally, SC10, expressed in (11), gives a reward for each day an ultra-clean OT is assigned to a unit with a light resource requirement or not assigned [80].

$$\begin{aligned}\text{Min} &- \theta_{\text{preferred}}\,\text{SC1} + \theta_e\,\text{SC2} - \theta_{\text{extra}}\,\text{SC3} - \theta_{\text{same}}\,\text{SC4} \\ &- \theta_{\text{consecutive}}\,\text{SC5} - \theta_{\text{weekly}}\,\text{SC6} + \theta_{\text{clashing}}\,\text{SC7} \\ &+ \theta_{\text{parallel}}\text{SC8} + \theta_{\text{heavy}}\text{SC9} - \theta_{\text{reserve}}\text{SC10} \end{aligned} \tag{1}$$

$$\text{SC1}: \sum_{g\in G}\sum_{(w,p.o)\in \text{SL}} x_{gwpo} \times \min(y_{go} + z_{gp}; 1) \tag{2}$$

$$\text{SC2}: \sum_{w\in W}\sum_{p\in P}\sum_{e\in E} \min((\text{qty}_{ewp} - \sum_{o\in O}\sum_{g\in G} x_{gwpo}\zeta_{ge}); 0) \tag{3}$$

$$\begin{aligned}\text{SC3}: &\sum_{w\in W}\sum_{p\in P}\sum_{e\in E} \max(0; (\text{qty}_{ewp} - \text{qty}_e^{\text{extra}} \\ &- \sum_{o\in O}\sum_{g\in G} x_{gwpo}\zeta_{ge} + 1)) \end{aligned} \tag{4}$$

$$\text{SC4}: \sum_{g\in G}\sum_{(w,p,o)\in \text{SL}} c_{gwpo} \tag{5}$$

$$\text{SC5}: \sum_{g\in G}\sum_{(w,p,o)\in \text{SL}} b_{gwpo} \tag{6}$$

$$\text{SC6}: \sum_{g\in G}\sum_{(w,p,o)\in \text{SL}} \alpha_{gwpo} \tag{7}$$

$$\text{SC7}: \sum_{g\in G}\sum_{i\in C_g}\sum_{(w,p,o)\in \text{SL}} x_{gwpo}\beta_{iwp} \tag{8}$$

$$\text{SC8}: \sum_{j\in J}\sum_{w\in W}\sum_{p\in P} q_{jwp} \tag{9}$$

$$\text{SC9}: \sum_{j\in J}\sum_{w\in W}\sum_{p\in P} h_{jwp} \tag{10}$$

$$\text{SC10}: \sum_{w\in W}\sum_{p\in P} \max(\sum_{o\in O}(\sum_{g\in G}\left(x_{gwpo}\iota_g\pi_o\right) + \delta_{wpo}\pi_o); 1) \tag{11}$$

2) NORMALIZED VALUES

The objectives have different granularity and levels of importance. To address the former, we normalized the values from each objective. The normalized value is obtained using $N = \frac{y - O_{\text{minvalue}}}{O_{\text{maxvalue}} - O_{\text{minvalue}}}$, where N is the normalized value, y is the non-normalized value, O_{maxvalue} is the extreme best value if the objective is satisfied, O_{minvalue} is the extreme worst value for the objective (ignoring other objectives) [56], [81], [82]. All soft constraints have minimum values of 0. The maximum

values for each objective are given in Appendix A. Afterwards, the normalized objective is multiplied by a weight assigned based on the importance of the objective.

3) CONSTRAINTS

The constraints can be divided into capacity allocation (12)-(15) and MSSP (16)-(20). Constraint (12) states that each surgical unit must be allocated at least one slot in the planning horizon. Constraint (13) maintains that the total slots allocated to all units of a subspecialty must not exceed the number of slots allocated to the subspecialty. Constraint (14) guarantees that the total slots allocated to all subspecialties must not exceed the number of slots in the planning horizon [36]. The number of slots for subspecialty j must not exceed its total feasible slots in the planning horizon, as stated in constraint (15).

For the MSSP, constraint (16) requires each slot to be assigned to only one unit [9], [14], [17]. Constraint (17) forces unit g to be assigned to a slot with an available surgery team on day p of the week w [9], [31]. Constraint (18) ensures unit g is only assigned to suitable or predefined OTs [17]. For each day, the number of slots assigned to units of subspecialty j must not exceed its maximum parallel slots, as expressed in constraint (19) [31]. Finally, constraint (20) ensures that the number of slots assigned to unit g equals the number allocated to the unit [21], [31].

$$\text{slot}_g \geq 1, \quad \forall g \in G \tag{12}$$

$$\sum_{g \in G_j} \text{slot}_g = \text{slot}_j, \quad \forall j \in J \tag{13}$$

$$\sum_{j \in J} \text{slot}_j = \text{slot}_{\text{total}} \tag{14}$$

$$\text{slot}_j \leq |W| \times \left| P_j^{\text{feasible}} \right| \times \text{slot}_j^{\text{parallel}}, \quad \forall j \in J \tag{15}$$

$$\sum_{g \in G} x_{gwpo} \leq 1, \quad \forall w \in W, p \in P, o \in O \tag{16}$$

$$x_{gwpo} \leq \text{dsp}_{gp}, \quad \forall g \in G, w \in W, p \in P, o \in O \tag{17}$$

$$\sum_{g \in G \setminus G_o} x_{gwpo} = 0, \quad \forall w \in W, p \in P, o \in O \tag{18}$$

$$\sum_{o \in O} \sum_{g \in G_j} x_{gwpo} \leq \text{slot}_j^{\text{parallel}}, \quad \forall w \in W, p \in P, j \in J \tag{19}$$

$$\sum_{w \in W} \sum_{p \in P} \sum_{o \in O} x_{gwpo} = \text{slot}_g \quad \forall g \in G \tag{20}$$

C. NUMERICAL EXAMPLE

The solution for the MSSP is represented in a two-dimensional array ($n \times m$), where n denotes the number of planning days, and m is the number of OTs. This array is populated by surgical units, signifying that the unit assigned to a particular slot can utilize the designated OT on the specified day. Empty positions within the array indicate unassigned slots.

We present a numerical example demonstrating the assignment of 15 surgical units from 11 subspecialties to a master plan, as in Fig. 2. The list of subspecialties and their surgical units can be found in Table 3. Each surgical unit is allocated a suitable number of slots based on surgery demand and waiting time. During master plan construction, the list of all booked surgeries for each surgical unit is retrieved. Each surgery contains two crucial pieces of information for OT capacity allocation, including the booking date and the estimated surgery duration provided by the medical team. Each surgical unit's cumulative estimated surgery duration is computed, and OT slots are allocated proportionally to the total duration. Surgical units with higher cumulative durations receive more slots in the master plan. The surgery booking date is utilized to calculate the waiting time for each surgery, and surgical units with longer average waiting times receive additional slots in the master plan. The resulting slot allocations are summarized in Table 3.

Subsequently, each surgical unit is assigned to specific OTs and days within the master plan based on the designated slot quantities. The slot assignment process adheres to constraints related to slot suitability and preferability for each surgical unit. For example, consider the unit VAS, which

W1	1	2	3	4	5	6	7
Mon	VAS	ENT (HV)	CTC	HEP	HEP	OPH (HV)	OPH (HV)
Tue	VAS	ENT (LT)	CTC	ART	ART	OPH (HV)	OPH (LT)
Wed	URO (HV)	ENT (HV)	CTC	PAE (LT)	PAE (HV)	OPH (HV)	OPH (LT)
Thu	URO (HV)	ENT (LT)	COL	COL	PAE (LT)	OPH (HV)	OPH (LT)
Fri	URO (LT)	ENT (HV)	NEU	NEU	PAE (LT)	ORT	ORT
W2	1	2	3	4	5	6	7
Mon	VAS	ENT (HV)	CTC	HEP	HEP	OPH (HV)	OPH (LT)
Tue	VAS	ENT (LT)	CTC	HEP	ART	OPH (HV)	OPH (LT)
Wed	URO (HV)	ENT (HV)	COL	PAE (LT)	PAE (HV)	OPH (HV)	OPH (LT)
Thu	URO (HV)	ENT (LT)	COL	PAE (LT)	PAE (HV)	OPH (HV)	OPH (LT)
Fri	URO (LT)	ENT (HV)	NEU	NEU	ORT	OPH (HV)	OPH (LT)

FIGURE 2. Example of a complete master plan generated based on the optimization model.

TABLE 3. Example of capacity allocation.

Subspecialty	Surgical unit	Allocated slots
VAS	VAS	4
URO	URO (HV)	4
	URO (LT)	2
ENT	ENT (HV)	6
	ENT (LT)	4
CTC	CTC	5
COL	COL	4
NEU	NEU	4
HEP	HEP	5
ART	ART	3
PAE	PAE (HV)	3
	PAE (LT)	5
OPH	OPH (HV)	10
	OPH (LT)	8
ORT	ORT	3

can only be scheduled for surgeries on Mondays, Tuesdays, or Wednesdays due to the availability of surgical personnel. However, there are no restrictions on which OT can be assigned. Consequently, the feasible slots for unit VAS encompass all OTs from Monday to Wednesday, whereas the blocked slots include all OTs on Thursday and Friday. In the overall slot assignment process, each surgical unit is accommodated in the master plan within their feasible slots. Fig. 2 provides a visual representation of a complete master plan, showcasing the assignment of each surgical unit to its allocated slots across specific OTs and days.

We provide examples illustrating the application of soft constraints within the given master plan. The assignment of unit VAS to OT1 on Monday in Week 1 (W1, Mon, OT1) satisfies SC1, as OT1 is the preferred choice for unit VAS. This adherence to the preference results in a reward being added to the objective value. SC4 is fulfilled when unit VAS is consecutively assigned on (W1, Mon, OT1) and (W1, Tue, OT1), earning the associated reward. In the case of SC5, unit PAE (LT) is assigned to different OTs on consecutive days (W1, Wed, OT4) and (W1, Thu, OT5), meeting the conditions and receiving a reward for the objective value. Similarly, for SC6, unit VAS is assigned to the same days and OTs for both (W1, Mon, OT1) and (W2, Mon, OT1), meeting the specified criteria.

For SC7, note that both units, CTC and HEP, utilize the same nursing team. Thus, they should not be assigned on the same day. In the example presented, however, both units are assigned on Monday in Week 1, resulting in a penalty to the objective value. The assignment of unit HEP also violates SC8 on Monday in Week 1, as it prefers to have only one slot per day but is given two slots. On the other hand, OPH assignments on Monday in Week 1 violate SC9 since two heavy units are assigned on the same day, which is avoided on other days. SC10 is fulfilled for assignments on all planning days except Monday in Week 1, when no slots are assigned to a unit with light resource requirements or are left empty.

For equipment sharing constraints, we present an example involving the equipment II machine used by VAS and ART units, where the hospital has only two units available. To avoid equipment sharing, SC2 requires that these units be assigned in just two slots on the same day. The example, however, reveals that the units have been assigned to three slots on Tuesday in Week 1 at OT1, OT4, and OT5, thereby violating the soft constraint. Conversely, SC3 requires that just one slot be assigned to any unit employing the equipment daily. This soft constraint is fulfilled on Monday in Week 1 since only OT1 is assigned to VAS and none to ART, satisfying the soft constraint and rewarding the objective value.

IV. PROPOSED IMPROVED SATURATION DEGREE-BASED CONSTRUCTIVE HEURISTIC

The MSSP constructive heuristic consists of three main tasks. The first task is selecting a surgical unit to assign. Then, a slot is selected among the feasible slots for the unit. Finally, if a suitable slot is not found, a unit already assigned in the incomplete master plan is selected to be swapped out.

This section first provides an overview of the main procedure involved in the proposed improved saturation degree-based constructive heuristic. This procedure comprises two key tasks: slot allocation and slot assignment, which are subsequently detailed in separate subsections. The proposed constructive heuristic, encompassing both tasks, is presented as Algorithm 1. Additionally, the algorithm governing the allocation of OT slots to surgery groups is extracted as Algorithm 2 and discussed in its corresponding subsection. The intentional separation of Algorithm 2 from Algorithm 1 aims to enhance clarity and facilitate a better understanding of the distinct tasks involved.

A. MAIN ALGORITHM

The algorithm requires the number of planning days and OTs to define the dimension of the master plan. Multiplying the number of weeks by the number of weekly working days yields the number of planning days. Besides, the list of medical subspecialties and their surgical units with information on constraints is also required. Information on the types of movable equipment and their quantities is also required. Finally, the algorithm requires a list of surgeries with their booking date, estimated duration and associated surgical unit for slot allocation. The pseudocode for the proposed improved saturation degree-based constructive heuristic is provided in Algorithm 1. The detailed pseudocode using programming syntax is given in Appendix B.

The algorithm is divided into two tasks: determining the number of slots allocated to each surgical unit (slot allocation) and assigning them to the master plan (slot assignment). The slot allocation is determined for each medical subspecialty $j \in J$ and then their surgical units $g \in G_j, \forall j \in J$ (line 1). The pseudocode for the slot allocation process is presented in Algorithm 2 and discussed in Section IV-B.

Afterwards, the surgical units are assigned to slots in the master plan. During slot assignment, conflicts may arise when a unit cannot be assigned due to its feasible slots having been taken by other units. The saturation degree heuristic is employed in sorting the surgical units to minimize conflicts (line 2). In addition to the number of feasible slots, two other factors were considered in the basic saturation degree heuristic. They are the number of preferred slots and the ratio between the slots allocated to the maximum possible slot allocation of each surgical unit. The former is added to increase the reward for assigning preferred slots. The saturation degree surgical unit sorting is explained in Section IV-C.

The algorithm then assigns surgical units with predefined OTs to their respective slots (line 3). In the slot assignment, units assigned to their allocated slots are removed from the unassigned list. Besides, the saturation degree's heuristic value, such as the number of feasible slots, changes with each assignment in the partial solution [66]. The list is re-sorted after each assignment.

Algorithm 1 Proposed Improved Saturation Degree-Based Constructive Heuristic

Input: Number of planning days and OTs, subspecialties (J) and surgical units (G) with their constraints information, movable equipment (E) type and quantity, surgeries list (S) with booking date and estimated duration.
Output: OT slot allocation and assignment of each surgical unit to a master plan.

1 Determine the suitable slot allocation for each surgical unit (slot_g) [see Algorithm 2].
2 Sort the surgical units using the saturation degree heuristic considering assignment difficulty and potential objective value [see Section IV-C.1].
3 Assign surgical units with predefined OTs. Re-sort the unassigned list.
4 Assign surgical units with preferred slots, where possible. Re-sort the unassigned list.
5 Iterate the unassigned list and assign the first unit (g). Stop when all units are assigned.
6 If unit g has feasible slots in the partial master plan.
7 Assign unit g to the slot with the fewest feasible unassigned units and highest potential objective value [see Section IV-C.2]. Re-sort the unassigned list.
8 Otherwise
9 If the repair mechanism has not reached its usage limit.
10 Swap unit g with unit g' from the partial master plan selected using the saturation degree heuristic [see Section IV-C.3]. Re-sort the unassigned list.
11 Otherwise
12 Mark the solution as infeasible.
13 End (return a complete master plan)

Subsequently, the algorithm assigns surgical units to their preferred slots if available (line 4). Then, the remaining units are iterated and assigned to a feasible slot at each iteration (lines 5-12). If feasible slots are available for the unit, the slot with the fewest feasible unassigned units and the highest potential objective value is chosen (lines 6-7). The slot selection process based on saturation degree is described in Section IV-C.

If no feasible slot can be found for the unit, the algorithm will execute a repair mechanism in which a unit is unassigned from the partial solution (lines 8-12). The saturation degree repair mechanism is discussed in Section IV-C. A maximum number of repair mechanism invocations is set to prevent cycling between two surgical units. The solution will be deemed infeasible if this limit is reached during the slot assignment (lines 11-12).

The algorithm stops when all slots have been assigned to the master plan. The algorithm's output includes a complete master plan satisfying all hard constraints, provided the repair mechanism does not exceed its allowed usage. Fig. 3 summarizes the proposed constructive heuristic in a flowchart. The green-shaded boxes represent the slot allocation (Algorithm 2), whereas the blue-shaded boxes represent the repair mechanism.

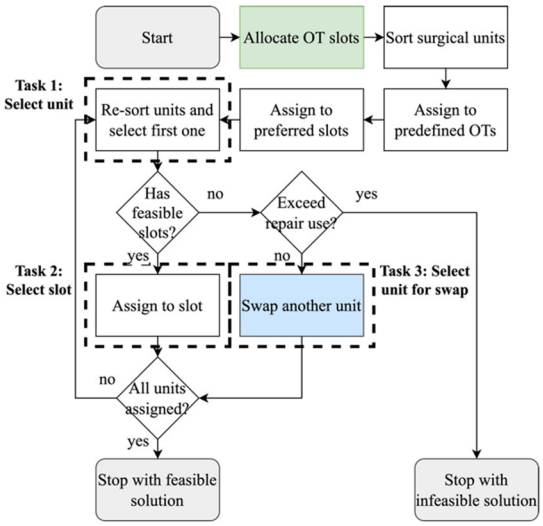

FIGURE 3. Flowchart of the improved saturation degree-based constructive heuristic.

The computational complexity of the proposed improved saturation degree-based constructive heuristic can be derived as $O(n \log n)$, where n depends on the number of surgical units. The algorithm sorts all surgical units in line 2, which takes $O(n \log n)$ computation. Assigning surgical units with predefined OTs and preferred slots involves nested loops iterating through surgical units, weeks, fixed OTs or preferred slots, and weekly days. The time complexity is influenced by the number of surgical units with fixed OTs or preferred slots. The main loop (lines 5-12) has a complexity that depends on the number of surgical units and the execution of assigning and swapping units. In both cases, score calculation involves operations dependent on the number of feasible slots and sorting that involves $O(m \log m)$, where m is the number of feasible slots. The overall time complexity of the entire algorithm is dominated by the main loop and the operations within each loop, which depend on the number of surgical units and slots. The exact complexity may vary based on specific conditions and the input data size.

B. SLOTS ALLOCATION

The slot allocation is determined based on the surgery demand represented by the estimated surgery duration. The number of surgeries was not considered to eliminate bias, such as surgical units with many short surgeries would be prioritized over units with fewer long surgeries. Slot allocation occurs in two phases, as described in Algorithm 2.

In the first phase, all slots in the planning horizon ($\text{slot}_{\text{total}}$) are distributed to the medical subspecialties (lines 1-5). The second phase involves distributing each subspecialty's allotted slots (slot_j) to their respective surgical units, following the same procedure as the first, where $\text{slot}_{\text{total}}$ is substituted to slot_j, dur_j to dur_g and $\text{dur}_{\text{total}}$ to dur_j (line 6). Firstly, the suggested slots are calculated from the ratio of the total estimated surgical times of the subspecialty/unit over the total

Algorithm 2 Algorithm to Allocate the Number of Slots for the Master Plan

Input: Subspecialties (J), Surgical units (G), Surgeries list with estimated duration and waiting time for each unit (S_j), number of slots in the planning horizon ($\text{slot}_{\text{total}}$)
Output: Slots allocation to each surgical unit (slot_g)

1 For each subspecialty j, calculate the suggested number of slots (slot_j) based on the total estimated surgery duration using $\text{slot}_j = \lfloor \text{slot}_{\text{total}} * \frac{\text{dur}_j}{\text{dur}_{\text{total}}} \rfloor$, where slot_j is rounded to the lower integer value, $\text{dur}_j = \sum_{g \in G_j} \sum_{s \in S_g} \text{dur}_s$ and $\text{dur}_{\text{total}} = \sum_{j \in J} \text{dur}_j$.
2 Repair slot_j for each subspecialty to avoid invalid values.
3 Calculate total residual slots ($\text{slot}_{\text{residual}}$) where $\text{slot}_{\text{residual}} = \text{slot}_{\text{total}} - \sum_{j \in J} \text{slot}_j$.
4 Sort the subspecialties by the average waiting time in decreasing order.
5 Iterate the subspecialties list and redistribute $\text{slot}_{\text{residual}}$ where $\text{slot}_j = \text{slot}_j + 1$ at each valid iteration until $\text{slot}_{\text{residual}}$ is exhausted
6 Allocate slot_j to the surgical units of subspecialty j using the same method as in lines 1-5, ensuring $\text{slot}_j = \sum_{g \in G_j} \text{slot}_g$
7 End (return number of slots allocated to all surgical units)

estimated surgical times of all subspecialties/units (line 1). The calculation will produce a decimal value. As the OT slots are non-shareable, the decimal number is converted to discrete values by rounding the values to the lowest integer. Then, the algorithm fixes invalid slot allocation (line 2). Next, the number of residual slots is then calculated (line 3).

A redistribution mechanism is introduced to allocate the residual slots based on waiting time (lines 4-5). Subspecialties/surgical units are sorted by average waiting time (line 4). The algorithm iterates through the sorted list, adding one slot to each subspecialty/unit subject to the hard constraints until there are no more residual slots (line 5). At the end of the algorithm, each surgical unit has been allocated a fair number of slots in the master plan based on surgery demand and waiting time.

C. SLOTS ASSIGNMENT

This section describes the saturation degree-based sorting for the surgical unit selection, slot selection and repair mechanism. The MSSP can be described as a timetabling problem involving allocating events and resources to space and time [47]. In the MSSP, surgery groups (events) and their resources, such as surgical equipment, are assigned to a set of OTs (space) and days (time). The saturation degree heuristic prioritizes events with fewer remaining feasible space and time, and the order dynamically changes after every assignment [66].

1) SURGICAL UNIT SORTING USING SATURATION DEGREE

Drawing inspiration from the heuristic hybridization from Sabar et al. [65], the surgical units are sorted based on multiple variables, and their ranks from each variable are combined to produce the *unit score*. Three variables are considered: the number of feasible slots, preferred slots, and the ratio slot allocation, hereafter called ratio max. The ratio max (rm) is calculated using $\text{rm} = \frac{\text{slot}_g}{|W| * \left|P_g^{\text{feasible}}\right| * \text{slot}_g^{\text{parallel}}}$, where slot_g is the slots allocated for unit g and $|W| * \left|P_g^{\text{feasible}}\right| * \text{slot}_g^{\text{parallel}}$ is unit g's maximum slots allocation.

Furthermore, the surgical units are sorted separately before being recombined from three smaller exclusive lists: *constrained units* (units with blocked OTs), *unconstrained units* (units without blocked OTs), and *fixed units* (units with predefined OTs). The surgical units are segregated as described because each list has different priorities.

Constrained units are sorted by all three variables. Firstly, the list is sorted by the number of feasible slots in increasing order, with ties broken by the number of preferred slots and random selection. Surgical units with fewer feasible slots are prioritized to ensure solution feasibility. Secondly, the units are sorted according to the number of preferred slots in increasing order, where units without any preferred slots are placed at the end of the list. Ties are broken by the number of feasible slots and random selection. Surgical units with fewer preferred slots have precedence in order to maximize the preferred slots' reward. Finally, the list is sorted by the decreasing ratio max, where ties are broken by random selection to increase solution feasibility.

In each sorted list, sorting scores are given inverse to the sorted placement, with the unit at the first position receiving the maximum score equivalent to the list size. In preferred slots sorting, units without preferred slots do not receive scores while not affecting the scores for other units. Afterwards, the scores from each sorted list are aggregated to obtain the *unit score*. Each score is multiplied by its respective weight coefficient: $\theta_{\text{preferred}} = 1$, $\theta_{\text{ratio}} = 2$, $\theta_{\text{feasible}} = 3$. The units are sorted by the *unit score* in decreasing order, where the number of feasible slots, preferred slots, and random selection breaks ties.

The *unconstrained* and *fixed units* are sorted considering only the number of preferred slots and the ratio max, following steps similar to those of the *constrained units*. The three smaller exclusive lists are then recombined, where the *constrained units* are placed before the *unconstrained units* to avoid infeasible solutions. *Fixed units* are added to the end of the final sorted list as they will not affect other assignments. Fig. 4 represents an example of the surgical unit sorting method, where yellow-shaded boxes denote constrained units, green-shaded boxes denote unconstrained units, and blue-shaded boxes denote fixed units. The sorting scores are indicated in brackets.

2) SLOT SELECTION USING SATURATION DEGREE

The algorithm iterates through unassigned units, assigning the unit with the highest priority to a slot selected based on the saturation degree strategy. All feasible slots for the unit are given *slot scores* calculated based on two factors, including:

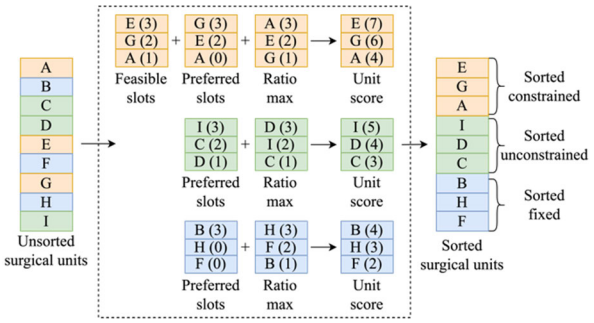

FIGURE 4. Illustration of the surgical unit sorting method.

1. The number of feasible unassigned units for the slot.
2. The potential objective function value if the chosen unit is assigned to the slot.

The potential objective function value is derived from the equations presented in Section III-B, limited to the chosen unit and slot. The slot with fewer feasible unassigned units and lower potential objective function value, as the problem is minimization, is given a higher score. The score values are normalized, where the number of feasible unassigned units is divided by the total number of unassigned units, and the potential objective function values are divided by their respective upper boundaries, given in Appendix C. The chosen unit is assigned to the slot with the highest *slot score* calculated by summing the weighted normalized score values.

3) REPAIR MECHANISM USING SATURATION DEGREE

A repair mechanism is called when all feasible slots of the chosen unit have already been taken. The repair mechanism vacates one of the feasible slots chosen based on its *swap score*. The *swap score* for a slot is calculated by adding the normalized values of the *repair unit score* of its original unit and its *repair slot score*.

The *repair unit score* is calculated following the unit sorting method based on the saturation degree discussed in Section IV-C, where surgical units are sorted considering three variables. The difference is that the number of feasible slots is measured when the master plan is empty (initial state) and at the partial master plan (current state). The following weights are used to obtain the *unit score*: $\theta_{\text{preferred}} = 1$, $\theta_{\text{ratio}} = 2$, $\theta_{\text{feas.init}} = 3$, $\theta_{\text{feas.cur}} = 4$.

Contrary to unit sorting in Section IV-C, a higher *repair unit score* is given to units with a higher feasible and preferred slots count and a lower ratio max. This method is employed because the repair mechanism aims to unassign a unit with the lowest assignment difficulty. The *repair unit score* is also normalized by dividing it by the number of unassigned units.

Meanwhile, the *repair slot score* is calculated similarly to the *slot score* in Section IV-C. The potential objective function value is extended to consider the potential gains if the chosen unit is assigned to the slot and the potential loss if the original unit is unassigned. A slot with fewer feasible unassigned units and higher objective value gain and lower objective value loss is preferred.

The slot with the highest *swap score* is selected. The original unit from the slot is re-inserted into the list of unassigned surgical units. Since the unit with the lowest assignment difficulty is selected, finding a replacement slot in the following iterations should be easier. The swap is performed by considering the objective function value so that the solution quality is unaffected.

TABLE 4. Details of the dataset.

Information	Description
Source	Hospital Canselor Tuanku Muhriz
Type	Booked surgeries in a year
Duration	1 January 2023 to 31 December 2023
Number of subspecialties	25
Number of surgical units	29
Number of surgeries-based	6006

V. COMPUTATIONAL EXPERIMENT

A. DATASET

The experiment utilizes the post-operative dataset from the case study hospital, HCTM, which includes surgeries booked between 1 January 2023 and 31 December 2023. Each surgery includes details on its booking date, estimated surgery duration, associated subspecialty, surgical unit, and type of resource requirement. The booking date is crucial for calculating waiting time, whereas the estimated surgery duration is used for slot allocation. These variables will be utilized in a simulation to assess the operational performance of the generated master plan. Information on subspecialty, surgical unit, and type of resource requirement determine the surgery's group affiliation to guide the assignment to the appropriate slot. Table 4 briefly summarizes the dataset employed in the experiment.

HCTM has 13 OTs and utilizes a two-week cyclic master plan with five weekly working days. 25 medical subspecialties comprised of 29 surgical units, where several subspecialties have two units, were considered. Surgeons from a subspecialty can only operate surgeries from its subspecialty. Meanwhile, anaesthetists and nurses are shared among the subspecialties and are assumed to be always available.

B. EXPERIMENTAL SETUP

Master plans for the case study hospital are generated by executing the proposed constructive heuristic written in Java. An additional Java program is developed to simulate surgery assignments to the master plans produced. The simulation gives an operation date for each surgery in the dataset from the first simulated planning day (1 January 2024) until the

TABLE 5. Sensitivity analysis of the objective function weightings.

S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	Objective Value	Average Waiting Time
1	1	1	1	1	1	1	1	1	1	-2.2819	303.1997
2	1	1	1	1	1	1	1	1	1	-2.6237	303.0138
1	2	1	1	1	1	1	1	1	1	-1.9987	303.0057
1	1	2	1	1	1	1	1	1	1	-2.8960	303.0454
1	1	1	2	1	1	1	1	1	1	-2.7200	303.0125
1	1	1	1	2	1	1	1	1	1	-2.9395	303.0197
1	1	1	1	1	2	1	1	1	1	-2.7934	303.0140
1	1	1	1	1	1	2	1	1	1	-2.1721	303.0482
1	1	1	1	1	1	1	2	1	1	-1.7793	303.0615
1	1	1	1	1	1	1	1	2	1	-1.9765	303.0215
1	1	1	1	1	1	1	1	1	2	-2.1822	303.0081
1	1	1	1	1	1	1	1	1	3	-2.1915	303.0138

dataset is exhausted, following the cyclic master plan. No holidays on workdays throughout the period are assumed.

The algorithm's performance is assessed using three metrics: solution feasibility, quality, and operational-level performance. Solution feasibility is measured by the number of times a feasible solution is found after 100 executions and the average repair mechanism calls. Solution quality is determined by the objective function value. The operational-level performance is measured by the average waiting time, waiting time standard deviation or variability among surgical units, and the number of days required to schedule all surgeries from the dataset. Each algorithmic run starts with an empty solution and stops when a complete solution is produced or the limit for repair mechanism usage is reached, resulting in an infeasible run. The experiment was executed on a computer with an Apple M1 chip with an 8-core CPU and 8GB of RAM.

C. SENSITIVITY ANALYSIS

The objective function for the optimization model consists of ten objectives aggregated using the weighted sum method. Each objective is assigned a weight value based on its importance. Ideally, the master plan generated should differ as the weightings are modified. The modifiable weightings will enable hospital planners to evaluate the effect of each objective on the solution quality and simulated operational performance of the master plan. Adapting the method by Penn et al. [16], the analysis is designed to explore the effect of small changes on each objective weighting.

Table 5 shows that the objective values of the master plans produced vary based on the different weightings used. On the other hand, the operational performance, assessed through the average waiting time of surgeries scheduled in a simulation, is only minimally influenced by the different master plans.

Additionally, the proposed constructive heuristic incorporates weight coefficients in calculating the *unit score* and *repair unit score*. These weight coefficients are lexicographical, leading to the examination of different permutations.

TABLE 6. Sensitivity analysis of the weight coefficient for the unit score.

$\theta_{\text{preferred}}$	θ_{ratio}	θ_{feasible}	Objective Value
1	2	3	**-2.2819**
1	3	2	-2.1799
2	1	3	-2.1910
2	3	1	-2.1806
3	1	2	-2.1837
3	2	1	-2.1893

TABLE 7. Sensitivity analysis of the weight coefficient for the repair unit score.

$\theta_{\text{preferred}}$	θ_{ratio}	$\theta_{\text{feas.init}}$	$\theta_{\text{feas.cur}}$	Objective Value
1	2	3	4	**-2.2819**
1	2	4	3	-2.1825
1	3	2	4	-2.1822
1	3	4	2	-2.1808
1	4	2	3	-2.1764
1	4	3	2	-2.1873
2	1	3	4	-2.1821
2	1	4	3	-2.1892
2	3	1	4	-2.1778
2	3	4	1	-2.1895
2	4	1	3	-2.1921
2	4	3	1	-2.1788
3	1	2	4	-2.1887
3	1	4	2	-2.1801
3	2	1	4	-2.1809
3	2	4	1	-2.1801
3	4	1	2	-2.1818
3	4	2	1	-2.1873
4	1	2	3	-2.1842
4	1	3	2	-2.1778
4	2	1	3	-2.1895
4	2	3	1	-2.1840
4	3	1	2	-2.1851
4	3	2	1	-2.1797

Table 6 illustrates that the best objective function value is obtained using $\theta_{\text{preferred}} = 1$, $\theta_{\text{ratio}} = 2$, and $\theta_{\text{feasible}} = 3$ as the weight coefficients for *unit scores*. Table 7 indicates that using $\theta_{\text{preferred}} = 1$, $\theta_{\text{ratio}} = 2$, $\theta_{\text{feas.init}} = 3$, and $\theta_{\text{feas.cur}} = 4$ weight coefficients for *repair unit scores* lead to the best objective function value. Consequently, these values are utilized in our experiment.

D. BENCHMARK CONSTRUCTIVE ALGORITHMS

The proposed improved saturation degree-based constructive heuristic is compared to two basic approaches (greedy and random constructive heuristics) and two algorithms from the literature (regret-based and saturation degree-based constructive heuristics). Benchmarking is conducted against only

constructive heuristics and not to improvement heuristics. The slot allocation is obtained from Algorithm 2 and remains constant for all benchmark algorithms. Algorithmic runs for each benchmark algorithm utilize the same execution environment as those employed in the proposed algorithm to ensure fairness. In addition, the generated master plans are compared to a manual master plan from the case study hospital.

A greedy approach is very popular due to its simplicity and efficacy as an initial solution for perturbative algorithms [83]. The greedy constructive heuristic in the experiment minimizes waiting time by assigning the highest average waiting time unit to the earliest feasible slot available. Ties are broken by selecting the unit with the higher total estimated surgery duration. Further ties are broken with random selection. The repair mechanism vacates the slot with the lowest average waiting time unit. However, this strategy may not return a feasible solution. Therefore, it is later replaced with random swaps.

The random constructive heuristic randomly selects a surgical unit at each iteration. Then, the unit is assigned to a random slot that does not violate hard constraints. If a feasible slot is unavailable, the heuristic vacates a random feasible slot in the partially constructed master plan.

The regret-based constructive heuristic [9] is adapted to prioritize surgical units with a higher waiting time. Ties are broken by selecting the unit with the higher total surgical times and random selection. Then, the *regret* value for each feasible slot is calculated by finding the difference between the highest and second-highest average waiting time of its feasible units. The slot with the smallest *regret* value is chosen, while ties are broken by considering the third highest. Infeasibility is repaired by swapping out the unit with the lowest average waiting time. However, the repair mechanism may not be able to guarantee feasibility and later switch to random swap.

The saturation degree-based constructive heuristic from Razali et al. [25] only considers the solution feasibility in surgical units and OT sorting. At each iteration, the surgical unit with the fewest feasible slots is assigned to a slot with the least feasible units. The repair mechanism removes the unit with the most feasible slots in the partial master plan.

The master plan utilized in the case study hospital, HCTM, for 2023 and 2024 is taken to compare the solution quality and operational-level performance. The comparison will demonstrate the potential improvements of using intelligent systems to generate the hospital's timetable.

E. RESULTS

1) SOLUTION FEASIBILITY

Due to the poor results on solution feasibility when using the greedy repair mechanism, additional runs using random swaps are executed. The solution feasibility for the hospital's master plan cannot be measured as it was produced manually by the hospital planners.

TABLE 8. Number of feasible runs by the benchmarked constructive algorithms (over 100 runs).

Algorithm (Repair Mechanism)	Number of Feasible Runs
Greedy (Greedy)	0
Greedy (Random)	100
Random (Random)	100
Regret-based (Greedy)	0
Regret-based (Random)	100
Saturation degree-based (Saturation degree-based)	100
Improved saturation degree-based (Improved saturation degree-based)	100

Table 8 shows the results of each algorithm to produce feasible solutions over 100 runs. It indicated that algorithms employing a greedy repair mechanism failed in producing a feasible solution. Infeasibility could be due to *cycling*, which occurs when the same pair of surgical units are exchanged with each other. Greedy approaches are more prone to suffer from this issue since they cannot select a different surgical unit when cycling occurs. On the other hand, random repair and saturation degree-based repair mechanisms produced feasible solutions in every run. Afterwards, we observe the number of repair mechanism usage required to obtain the feasible solutions among these algorithms.

Fig. 5 shows the box-plot graph of the repair calls, where a lower value is preferred. It shows that both saturation degree-based constructive heuristics outperform other algorithms in minimizing the repair mechanism usage. The improved variant performs best at an average of 2.80 calls over 100 runs. Furthermore, both saturation degree-based constructive heuristics have a smaller deviation and lower maximum value compared to the other algorithms using random swaps. Fewer requirements for the repair mechanism demonstrate the efficacy of the saturation degree heuristic for choosing a surgical unit and slot. Next, the difference in execution time between the basic approaches (greedy and random) and the designed approaches (saturation degree-based and Spratt and Kozan [9]'s regret-based) is investigated.

Fig. 6 represents the box-plot graph of the execution times, in seconds, for the benchmark configurations. It demonstrates that the improved saturation degree-based constructive heuristic suffers a longer average execution time that is arguably unnoticeable. Notably, extreme outliers exist for the proposed algorithm, with runs between 0.06 and 0.10 seconds. Nevertheless, the difference is negligible as all values are small (less than one second).

2) SOLUTION QUALITY

Runs using the greedy repair mechanism are excluded from hereafter. Table 9 shows the normalized rewarding objectives values. The improved saturation degree-based constructive heuristic recorded the most rewards for the preferred

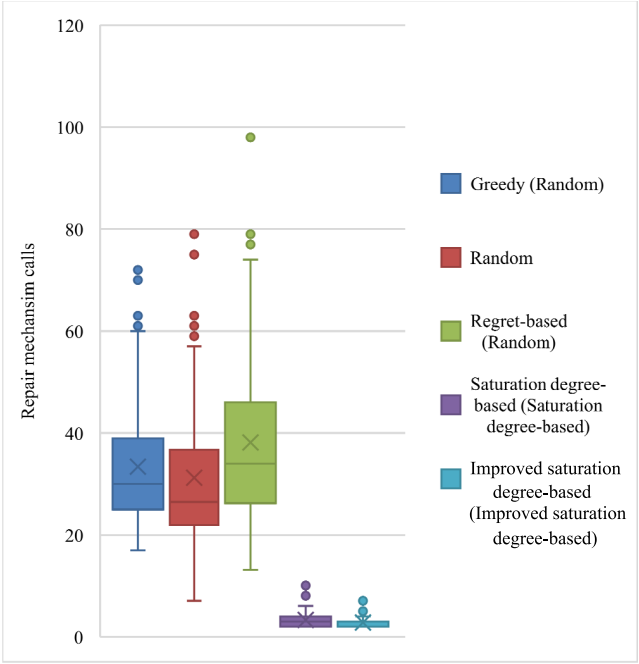

FIGURE 5. Box-plot graph for the repair mechanism calls of the benchmarked constructive algorithms (lower is better).

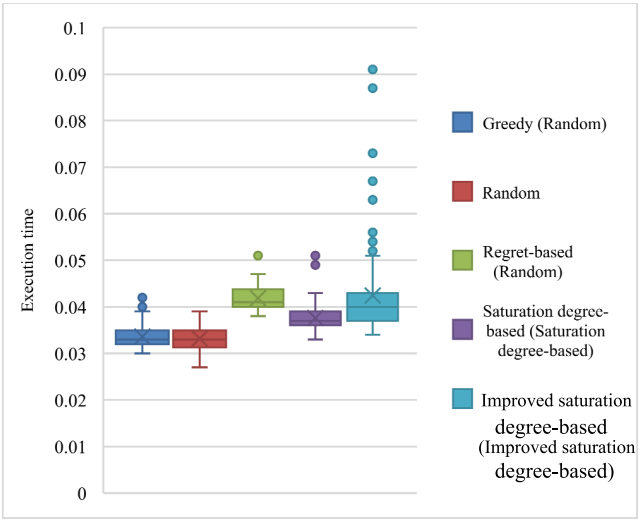

FIGURE 6. Box-plot graph for the execution time of the benchmarked constructive algorithms (lower is better).

slot (SC1), same OT (SC4), and consecutive days (SC5) objectives. Meanwhile, the original saturation degree-based constructive heuristic outputs the best results for the extra equipment (SC3) objective. The hospital's master plan achieved the highest same weekly slot (SC6), indicating the master plans from the first and second weeks have high schedule stability. Besides, the plan was able to reserve an ultra-clean OT (SC10) on each planning day, which is not achieved by any of the intelligent approaches.

Table 10 shows the values for penalizing objectives. Based on the results shown, the movable equipment sharing (SC2), clashing subspecialty (SC7), and parallel heavy resource requirements slots (SC9) were minimized best by the proposed improved saturation degree-based constructive heuristic. This observation highlights the effectiveness of the proposed algorithm in reducing resource-sharing among scheduled surgeries, contributing to a lower chance of scheduling disruption. The best value for exceeding the soft limit on the parallel slots (SC8) is obtained by the hospital's master plan. This could happen because some slots are not utilized in the hospital's master plan, potentially affecting the surgery scheduling performance since less OT time is available.

The weighted sum of the objectives is demonstrated in Fig. 7. It indicates the hospital's master plan as the best performer (at -2.7411), followed closely by the proposed improved saturation degree-based constructive heuristic (with -2.2819). The hospital's plan excelled in achieving the ultra-clean OT objective (SC10), significantly increasing its weighted sum of objectives. In contrast, the improved saturation degree-based constructive heuristic did not fulfil this soft constraint on any planning days. Despite this, we contend that the proposed constructive heuristic produces high-quality solutions, as evidenced by its superior performance in six of the ten objectives. Other benchmarked algorithms performed significantly worse than the two. We theorize that there could be a trade-off between several objectives and operational

TABLE 9. Rewarding objective value results for the benchmarked constructive algorithms (bold represents best).

Algorithm	SC1	SC3	SC4	SC5	SC6	SC10
Greedy	0.0608	0.7221	0.3281	0.7438	0.2166	0.004
Random	0.0705	0.7178	0.2676	0.6679	0.2195	0.029
Regret-based	0.0721	0.7178	0.2632	0.6642	0.2252	0.016
Hospital's plan	0.1692	0.6323	0.3162	0.4444	**0.8923**	**1.000**
Saturation degree-based	0.0876	**0.7230**	0.2626	0.5674	0.3277	0.001
Improved saturation degree-based	**0.4378**	0.7159	**0.5412**	**0.7582**	0.5928	0.000

TABLE 10. Penalizing objective value results for the benchmarked constructive algorithms (bold represents best).

Algorithm	SC2	SC7	SC8	SC9
Greedy	0.1784	0.0118	0.5412	0.3567
Random	0.1708	0.0109	0.4543	0.2655
Regret-based	0.1707	0.0108	0.4567	0.2677
Hospital's plan	0.1839	0.0128	**0.3167**	0.2000
Saturation degree-based	0.1800	0.0123	0.5535	0.3567
Improved saturation degree-based	**0.1674**	**0.0103**	0.3932	**0.1932**

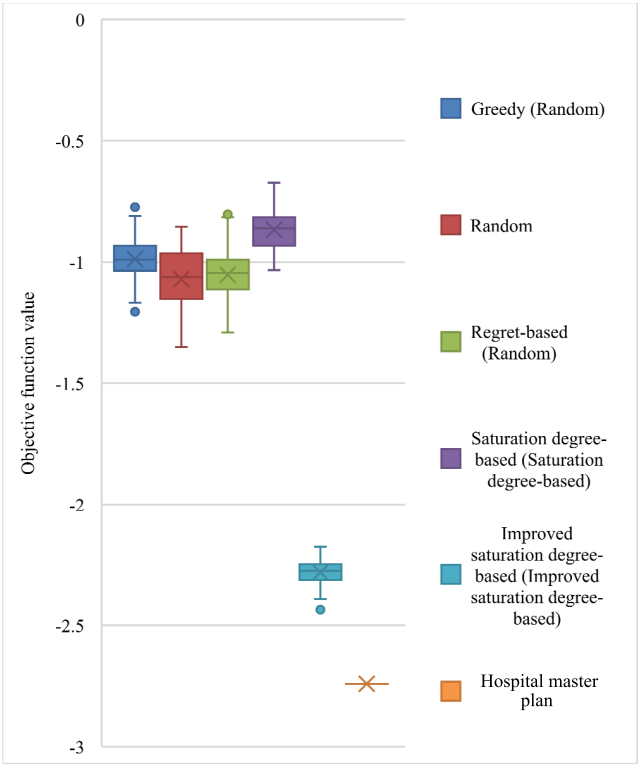

FIGURE 7. Box-plot graph for the objective function value of the benchmarked constructive algorithms (lower is better).

scheduling performance. The objective values for SC2, SC3, SC7, SC8 and SC9 could be increased if fewer surgical units are assigned on the same day or fewer OTs are in use.

3) SIMULATED OPERATIONAL LEVEL PERFORMANCE

The operational-level performance is evaluated in terms of surgeries' waiting time, the standard deviation or variability for waiting times among surgical units, and the number of days required to operate all surgeries. The results of the surgery scheduling simulation are presented in Table 11. Based on the results, the hospital's plan obtained the best average waiting time (with an average of 300.53 days). Results from the constructive heuristics are nearly identical, nearly three days longer than the hospital's master plan (ranging from 302.97 to 303.26 days). This increase may be attributed to the slot allocation from the constructive heuristic, which failed to allocate sufficient slots to surgical units with many surgeries. Consequently, the days to complete the surgery list are lower for the manually generated master plan (309 days) than for the intelligent approaches (370 days). Despite this, the standard deviation for waiting times among surgical units is smaller for the master plans produced by constructive heuristics compared to the hospital's plan, at 67.772 compared to 70.720. This suggests a more balanced distribution of waiting time values among the surgical units.

F. STATISTICAL ANALYSIS

To support the conclusions drawn from the results, we conducted statistical analysis on all results acquired, including

TABLE 11. Simulation results for the benchmarked constructive algorithms (bold represents best).

Algorithm	Average Waiting Time	Variability in Waiting Time Across Surgical Units	Days Required
Greedy	303.26	**67.080**	370
Random	303.05	67.759	370
Regret-based	302.97	67.763	370
Hospital's plan	**300.53**	70.720	**309**
Saturation degree-based	303.05	67.772	370
Improved saturation degree-based	303.20	67.323	370

normality tests and non-parametric tests with pairwise comparison. The hospital's master plan is excluded from the analysis as only one master plan exists, and each run of the simulation will produce the same result.

Normality Tests (Shapiro-Wilk Royston Test): We tested the normality of the three performance measures (repair calls, average waiting time and objective value) from the five benchmarked algorithms using the Shapiro-Wilk Royston test [84]. According to the results, at least one sample is not normally distributed for every performance measure. Therefore, non-parametric tests are carried out next.

Non-Parametric Tests (Kruskal-Wallis Test): We performed the Kruskal-Wallis test ($\alpha = 0.05$) to compare five independent samples for each performance measure. The results showed a significant difference between the algorithms in all performance measures (p-value = 7.63057E-78 for repair calls; p-value = 1.09357E-35 for average waiting time and p-value = 1.78344E-73 for objective value).

Post-hoc Tests (Dunn's Test): Post-hoc pairwise comparisons are carried out on ten algorithm pairings for each performance measure. The test has proven the significance of the differences in repair calls for all pairs except *Greedy–Random* (p-value = 0.247), *Greedy–Regret-based* (p-value = 0.257), and *Saturation degree-based–Improved saturation degree-based* (p-value = 0.264). For average waiting time, only *Random–Saturation degree-based* (p-value = 0.895) indicates non-significance. For objective value, the null hypothesis was rejected only for *Random–Regret-based* (p-value = 0.898), indicating no difference between the samples. The claim that the proposed algorithm outperformed the benchmark algorithms in repair calls has been statistically proven, except for the original saturation degree-based constructive heuristic, where no statistical difference was observed between the improved and original variants. The regret-based constructive heuristic exhibits a statistically significant advantage over other algorithms regarding average waiting time but shows no significant difference in objective value compared to the random approach. Importantly, the

improved saturation degree-based algorithm demonstrates a statistically significant superiority in objective value compared to other algorithms.

VI. PERFORMANCE AGAINST IMPROVEMENT HEURISTICS

A. BENCHMARK IMPROVEMENT ALGORITHMS

Comparison of our proposed constructive heuristic to improvement heuristics is unfair as the former is only run for one iteration. In contrast, the latter executes multiple iterations to find a higher-quality solution. Despite this, we are interested in observing the performance of our proposed algorithm against state-of-the-art improvement heuristics for the MSSP, including SA metaheuristics and two SA-based hyper-heuristics. Our proposed algorithm does not necessarily have to beat the improvement heuristics to be presented as a good solution method.

The improved saturation degree-based constructive heuristic is compared against SA, Hyper SA, and Hyper SA-TS, which were introduced by Spratt and Kozan [9]. The algorithms utilize five types of moves involving assignment swaps and modifications to the schedule structure, particularly regarding half-day and full-day slots. When adapting their approach to our problem, we incorporated the assignment swap moves while omitting the schedule structure modifications. This is because our case study does not utilize half-day slots, rendering such modifications not possible. We adapted the assignment swap moves to:

1. Swap two subspecialties.
2. Swap two surgical units.
3. Shuffle three subspecialties.
4. Shuffle three surgical units.
5. Swap surgical units until a feasible solution is obtained, up to a maximum number of swaps.

SA employs a randomly selected move to obtain a candidate solution. The acceptance of this candidate solution as the current working solution is determined by a probability based on the change in the objective function value and the solution temperature. The solution temperature decreases upon accepting a solution, calculated using the formula $t^* = \frac{t}{1+\varepsilon t}$, where t^* is the new value, t is the previous value, and ε is the temperature decrease factor.

In Hyper SA, a move is selected based on heuristic rankings, guided by a Tabu search strategy. The algorithm chooses the move with the highest heuristic rank, which is not in the Tabu list. It applies the same move for multiple iterations, referred to as a block, before updating the heuristic ranks. A heuristic's rank increases when the solution from a block improves the previous solution; otherwise, the rank is reduced, and the heuristic is added to the Tabu list. The SA approach is employed for move acceptance, where the temperature decreases when the candidate solution is accepted and increases otherwise. The solution temperature increases at a rate of $t^* = \frac{t}{1-\gamma t}$, where t^* is the new value, t is the previous value, and γ is the temperature increase factor.

TABLE 12. Parameter settings for the benchmarked improvement algorithms.

Parameter	SA	Hyper SA	Hyper SA-TS
Initial temperature	10	10	1
Temperature decrease factor	0.5	0.95	0.95
Temperature increase factor	NA	0.76	0.76
Block length	50	100	50
Tabu length for neighbourhood moves	NA	3	4
Tabu length for assignments	NA	NA	20

Hyper SA-TS extends Hyper SA by incorporating a Tabu mechanism to restrict neighbourhood moves on forbidden assignments (combinations of OT time block and subspecialties or surgical units). Specifically, the algorithm is prohibited from performing swaps on assignments that have previously led to worsening solutions. The algorithm maintains two Tabu lists: one for the neighbourhood moves and the other for the assignments. The Tabu length for the latter is dynamic and decreases at iteration intervals. Other mechanisms, including the move acceptance strategy, remain similar to the implementation of Hyper SA.

The improvement algorithms undergo 31 runs to ensure robust and representative results across multiple iterations. The first 31 runs of the improved saturation degree-based constructive heuristic, as used in the constructive heuristics comparison, are employed for the comparison of improvement algorithms. Based on the configuration provided by Spratt and Kozan [9], each algorithm undergoes 16,000 iterations from an initial solution produced by the improved saturation degree-based constructive heuristic. The parameter settings align with the specifications outlined in the original publication, as illustrated in Table 12. Readers are encouraged to consult the source publication for a comprehensive understanding of each algorithm.

B. RESULTS

1) SOLUTION QUALITY

The normalized rewarding objective values in Table 13 reveal that the improved saturation degree-based constructive heuristic excelled in SC1, SC4, and SC5, surpassing the improvement algorithms. Assignment swap moves were ineffective in enhancing these objectives. However, they successfully improved SC6 and SC10, where SA increased the rewards. Hyper SA exhibited the best performance for SC3, albeit with only a slight difference compared to the proposed algorithm.

Regarding penalizing objective values, as shown in Table 14, the proposed algorithm exhibits the best value only for SC2. Assignment swap moves were effective in improving SC7, SC8, and SC9. SA leads to improvements in SC8 and

TABLE 13. Rewarding objective value results for the benchmarked improvement algorithms (bold represents best).

Algorithm	SC1	SC3	SC4	SC5	SC6	SC10
SA	0.3290	0.7181	0.4935	0.7414	**0.6650**	**0.2871**
Hyper SA	0.4270	**0.7183**	0.5313	0.7579	0.5658	0.0000
Hyper SA-TS	0.4268	0.7180	0.5305	0.7499	0.5757	0.0032
Improved saturation degree-based	**0.4382**	0.7159	**0.5434**	**0.7607**	0.5906	0.0000

TABLE 14. Penalizing objective value results for the benchmarked improvement algorithms (bold represents best).

Algorithm	SC2	SC7	SC8	SC9
SA	0.1894	0.0102	**0.2672**	**0.0817**
Hyper SA	0.1899	0.0101	0.4016	0.2022
Hyper SA-TS	0.1892	**0.0100**	0.3995	0.1995
Improved saturation degree-based	**0.1675**	0.0101	0.3925	0.1925

SC9, whereas Hyper SA-TS enhances SC7 compared to the proposed constructive heuristic.

Fig. 8 presents a box-plot graph depicting the weighted sum of the objectives, comparing the proposed improved saturation degree-based constructive heuristic to the benchmark improvement algorithms. The graph demonstrates that the improvement algorithm, particularly SA, has improved solution quality by 17.50% (-2.686 compared to -2.286). This improvement is achieved through the application of assignment swap moves to a complete initial solution. SA results suggest that the objectives can be increased at the cost of other objectives. However, the other improvement algorithms, Hyper SA and Hyper SA-TS, fall short of outperforming the proposed constructive heuristic. We observed that they failed to improve their initial solutions, possibly due to ineffective pairing of the swap moves and algorithmic strategies.

2) SIMULATED OPERATIONAL LEVEL PERFORMANCE

Table 15 indicates that average waiting times are improved when simulating scheduling surgeries into master plans produced by the improvement algorithms. Specifically, Hyper SA generates master plans with the lowest average waiting time. However, the variability for waiting time among surgical units worsens compared to the master plans from the proposed improved saturation degree-based constructive heuristic. The days required to schedule all surgeries remain the same for all algorithms because they utilize the same slot allocation.

C. STATISTICAL ANALYSIS

Normality Tests (Shapiro-Wilk Test): The normality of results for three performance metrics (objective value, average waiting time, and waiting time standard deviation are assessed for the improvement algorithms and the proposed constructive heuristic. The Shapiro-Wilk test is employed, considering a sample size of 31. The test results indicated that at least one sample in each performance metric is non-normally distributed. Consequently, non-parametric tests are utilized for hypothesis testing.

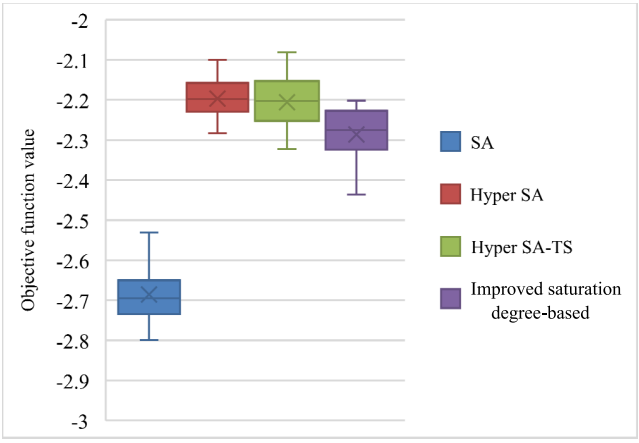

FIGURE 8. Box-plot graph for the objective function value comparing the proposed constructive heuristic to improvement algorithms (lower is better).

TABLE 15. Simulation results for the benchmarked improvement algorithms (bold represents best).

Algorithm	Average Waiting Time	Variability in Waiting Time Across Surgical Units	Days Required
SA	303.06	68.534	**370**
Hyper SA	**302.98**	68.095	**370**
Hyper SA-TS	303.00	68.111	**370**
Improved saturation degree-based	303.19	**67.329**	**370**

Non-Parametric Tests (Kruskal-Wallis Test): Kruskal-Wallis tests at a 0.05 significance level indicate statistically significant differences among the results from the four algorithms across all three performance metrics: objective value (p-value = 9.14725E-19), average waiting time (p-value = 2.39033E-05), and waiting time standard deviation (p-value = 1.68388E-18). Subsequent post-hoc tests are conducted to identify specific pairs of algorithms with significant differences.

Post-hoc Tests (Dunn's Test): Post-hoc Dunn's tests revealed significant differences in the objective value between SA and other algorithms, including the proposed constructive heuristic. Only the pair of Hyper SA and Hyper SA-TS shows an insignificant difference in objective value (p-value = 0.591). Regarding average waiting time, Hyper SA exhibited a significant difference from the proposed

algorithm (p-value = 8.70127E-06), whereas there was no significant difference with Hyper SA-TS (p-value = 0.631). For the waiting time standard deviation, all pairings demonstrated significant differences except for *Hyper SA–Hyper SA-TS* (p-value = 0.896), indicating the statistically proven superior performance of the proposed constructive heuristic in this metric.

VII. PERFORMANCE AS INITIAL SOLUTIONS

We aim to investigate the impact of employing solutions generated from various constructive heuristics on the overall effectiveness of improvement heuristics. In this experiment, we utilize solutions generated by greedy, random, regret-based, saturation degree-based, and improved saturation degree-based constructive heuristics. The improvement heuristic SA, identified as the most effective in Section VI, is applied to enhance solutions produced by each constructive heuristic. We generate 100 initial solutions and improve them using SA to obtain the final solution, employing the same experimental setup as in previous sections. The outcomes are presented below.

A. RESULTS

1) SOLUTION QUALITY

The SA algorithm produced the best solutions for the preferred slot (SC1), same OT (SC4), and same weekly slot (SC6) objectives when utilizing initial solutions from the proposed improved saturation degree-based constructive heuristic, as indicated in Table 16. The large difference in the objective value of SC1 compared to other initial solutions is attributed to the proposed heuristic's strategy of assigning surgery groups to their preferred slots when the master plan is empty. In contrast, other solutions were improved through iterative swap moves on a complete solution. The changes are more restricted, leading to lower improvements in the objective value. The objectives related to extra equipment (SC3) and consecutive days (SC5) are optimized most effectively when utilizing regret-based constructed initial solutions, albeit with small differences compared to other initial solutions. The original saturation degree-based constructive heuristic can reach solutions with a higher average for the ultra-clean OT (SC10) objective, where the proposed algorithm performed less favourably.

Table 17 highlights that random initial solutions exhibit optimal minimization of penalties for parallel slots (SC8) and parallel heavy resource requirements slots (SC9). The lowest penalty for clashing subspecialty (SC7) is observed when utilizing initial solutions generated through the regret-based constructive heuristic. The combination of the proposed constructive heuristic with the SA improvement heuristic achieved the most favourable outcome for the movable equipment sharing objective (SC2). It is crucial to emphasize that all observed differences are minimal, suggesting that the improvement heuristic could potentially be the bottleneck in terms of performance.

TABLE 16. Rewarding objective value results comparing different initial solutions (bold represents best).

Algorithm	SC1	SC3	SC4	SC5	SC6	SC10
Greedy + SA	0.2195	0.7182	0.4059	0.7368	0.6062	0.289
Random + SA	0.2125	0.7183	0.4009	0.7302	0.6125	0.296
Regret-based + SA	0.2204	**0.7185**	0.4067	**0.7390**	0.6003	0.297
Saturation degree-based + SA	0.2109	0.7183	0.4007	0.7360	0.6086	**0.299**
Improved saturation degree-based + SA	**0.3369**	0.7179	**0.4837**	0.7312	**0.6786**	0.284

TABLE 17. Penalizing objective value results comparing different initial solutions (bold represents best).

Algorithm	SC2	SC7	SC8	SC9
Greedy + SA	0.1897	0.0102	0.2690	0.0860
Random + SA	0.1899	0.0102	**0.2683**	**0.0802**
Regret-based + SA	0.1902	**0.0101**	0.2710	0.0848
Saturation degree-based + SA	0.1898	0.0103	0.2710	0.0860
Improved saturation degree-based + SA	**0.1890**	0.0103	0.2723	0.0813

The box-plot graph in Fig. 9 demonstrates that employing initial solutions generated by the improved saturation degree-based constructive heuristic yields the overall best solution (with an average of -2.6793). This aligns with the notion that the initial solution quality impacts the final solution, with high-quality initial solutions leading to better final solutions [85]. Wang et al. [86] suggested that high-quality initial solutions can enhance computational efficiency. Our results support this idea, as achieving a solution of comparable quality is possible with lower computational time when starting from initial solutions already close to a specific point, compared to less optimal initial solutions.

Apart from the improved saturation degree-based constructive heuristic, other configurations achieved nearly identical final solution quality despite differing initial fitness, as shown in Table 18. This similarity in outcomes suggests the possibility that these configurations have reached a local optimum. This implies that the effectiveness of an improvement algorithm could be constrained by the starting point [87], especially those without effective diversification strategies.

2) SIMULATED OPERATIONAL LEVEL PERFORMANCE

Table 19 shows that the regret-based constructive heuristic combined with the SA improvement algorithm obtained the best operational-level performance. This is evidenced by achieving the lowest waiting time average and variability across surgical units.

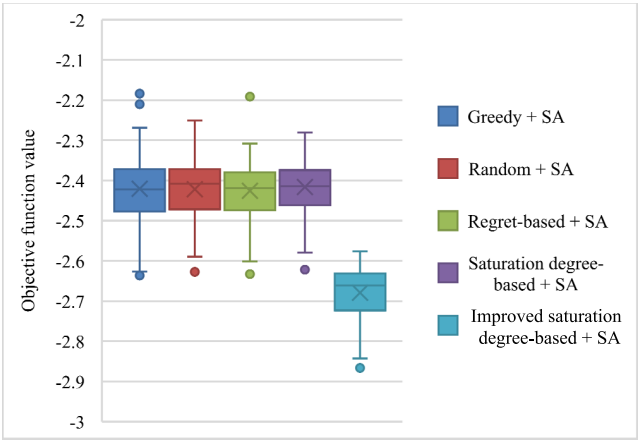

FIGURE 9. Box-plot graph for the objective function value comparing the performance using different initial solutions (lower is better).

TABLE 18. Rate of improvements from using different initial solutions.

Algorithm	Average Initial Quality	Average Final Quality	Average Delta	Relative Improvement
Greedy + SA	-0.9326	-2.4206	1.4880	161.72%
Random + SA	-1.0367	-2.4218	1.3851	136.64%
Regret-based + SA	-1.0320	-2.4257	1.3936	137.25%
Saturation degree-based + SA	-0.8542	-2.4164	1.5622	185.24%
Improved saturation degree-based + SA	-2.1753	-2.6793	0.5041	23.24%

TABLE 19. Simulation results comparing different initial solutions (bold represents best).

Algorithm	Average Waiting Time	Variability in Waiting Time Across Surgical Units	Days Required
Greedy + SA	**303.01**	68.543	**370**
Random + SA	303.09	68.533	**370**
Regret-based + SA	**303.01**	**68.490**	**370**
Saturation degree-based + SA	303.07	68.553	**370**
Improved saturation degree-based + SA	303.06	68.543	**370**

B. STATISTICAL ANALYSIS

Normality Tests (Shapiro-Wilk Royston Test): We assessed the normality of distributions for objective value, average waiting time, and waiting time variability samples using the Shapiro-Wilk Royston test. The results revealed non-normal distribution only for the objective value of the improved saturation degree-based constructive heuristic (p-value = 1.65462E-4) and the waiting time variability of the random (p-value = 2.01878E-2) and regret-based approaches (p-value = 1.81908E-2). Consequently, non-parametric tests are applied to identify significant differences in objective value and waiting time variability. As all samples for average waiting time exhibit normal distribution, parametric tests are employed.

Hypothesis Testing (Kruskal-Wallis Test & One-Way ANOVA): Kruskal-Wallis tests were performed on non-normally distributed samples for objective value and waiting time variability, revealing a significant difference in objective value (p-value = 4.9351E-50). However, no statistical significance was observed for waiting time variability (p-value = 0.77705). The parametric one-way analysis of variance (ANOVA) [88] is applied to the normally distributed samples of average waiting time, indicating a significant difference with a p-value of 0.02065. Subsequent pairwise comparisons are conducted using suitable post-hoc tests.

Post-hoc Tests (Dunn's Test & Tukey's Test): Post-hoc tests were applied to the significantly different samples of objective value and average waiting time, with ten pairwise comparisons across five algorithms. Dunn's test, employed for non-normally distributed objective value samples, revealed significant differences between the improved saturation degree-based constructive heuristic and all other heuristics (p-value = 3.12961E-11 to greedy, p-value = 6.70597E-12 to random, p-value = 5.91296E-11 to regret-based, and p-value = 2.77667E-12 to saturation degree-based). No significant differences were observed among the remaining four heuristics. Conversely, Tukey's test, conducted on normally distributed average waiting time samples, displayed no significant differences between any heuristic pairs, contrary to the parametric one-way ANOVA's result. This discrepancy may arise from differing sensitivities of one-way ANOVA and Tukey's test, along with the latter's multiple comparison adjustment, rendering it less conservative and more sensitive [89]. Additionally, two heuristic pairings approached significance (p = 0.05969 for *Greedy–Random* and 0.05381 for *Random–Regret-based*), suggesting potential sensitivity to sample variations [90].

VIII. CONCLUSION

This study has addressed the issues in MSSP by proposing a mathematical model variant with a novel objective function and an improved saturation degree-based constructive heuristic that enhances both solution quality and feasibility. Compared to the original saturation degree-based variant, the improved heuristic reduces repair mechanism requirements by 14.63% and increases objective function value by 2.6 times, supported by statistical tests that demonstrated significant differences with small p-values. The results indicate that solution quality and feasibility can be increased concurrently. Moreover, a simulation study suggested that the proposed heuristic could yield a more balanced waiting time distribution among surgical units compared to the hospital's plan. Initial solutions generated by the proposed algorithm

can also result in superior solutions discovered by improvement algorithms.

We attribute the increased solution quality to the proposed heuristic's ability to maximize potential objective value while ensuring feasibility at each step of solution construction. In contrast, the original variant prioritized slot feasibility, compromising quality, whereas the greedy heuristic focused on objective value, potentially reducing both feasibility and quality. The regret-based approach failed to enhance solution feasibility or quality consistently, whereas the random approach lacks intelligent rules during solution construction. Given the superior performance of the improved saturation degree-based constructive heuristic, it holds potential for application in other timetabling problems, such as conference and event scheduling, sports scheduling, and employee shift scheduling.

Despite its performance improvement, the proposed method has shortcomings, including subjectivity in solution quality and the need for a repair mechanism. Implementation may require adjustments to align with specific objectives in different contexts. Besides, the model formulated in this study has several limitations, such as uncertainty in the surgery duration and demand not being considered. Future works can extend the model and constructive heuristics to include stochastic and dynamic aspects.

APPENDIX

A. UPPER BOUNDARY FOR THE NORMALIZATION OF MASTER PLAN'S OBJECTIVE VALUES

The maximum values for each soft constraint are detailed in Table 20.

TABLE 20. Best- or worst-case scenario for each assignment objective.

Soft Constraints	Best- or Worst-Case Scenario	Maximum Value (O_{maxvalue})
SC1	All slots in the master plan have been assigned to a unit that preferred the slot.	$\lvert W\rvert * \lvert P\rvert * \lvert O\rvert$
SC2	All equipment used by a unit is shared in all OTs for all planning days.	$\left(\sum_{e\in E} u_e(\text{qty}_{ewp} - \lvert O\rvert)\right)\times\lvert W\rvert\times\lvert P\rvert$ u_e: 1, if equipment e is used by any unit
SC3	No unit used any equipment for all planning days.	$\sum_{e\in E} u_e \text{qty}_{ewp}\times\lvert W\rvert\times\lvert P\rvert$ u_e: 1, if equipment e is used by any unit
SC4	All slots have been assigned when the same unit is assigned to the same OT on the previous day for all planning days except the first.	$(\lvert W\rvert\times\lvert P\rvert - 1)\times\lvert O\rvert$
SC5	All slots have been assigned when the same unit is assigned in any OT of the previous day for all planning days except the first.	$(\lvert W\rvert\times\lvert P\rvert - 1)\times\lvert O\rvert$
SC6	All slots have been assigned when the same unit is assigned in the same OT and day in the previous week for all planning days except the working days in the first week.	$(\lvert W\rvert - 1)\times\lvert P\rvert\times\lvert O\rvert$
SC7	For all slots, each time a unit is assigned, all other OTs on the same day have been assigned to their clashing units.	$\min(\sum_{g\in G} \varsigma_g \text{slot}_g^{\text{parallel}}; \lvert O\rvert)\times(\lvert O\rvert - 1)\times\lvert W\rvert\times\lvert P\rvert$ ς_g: 1, if unit g has any clashing subspecialty/unit $slot_g^{parallel}$: Maximum number of parallel slots for unit g
SC8	For all planning days, if there are fewer subspecialties than OTs, each subspecialty is assigned to more than one OT. Otherwise, the subspecialties are assigned to two slots, except the last one, if there is an odd number of OT.	$\lvert W\rvert\times\lvert P\rvert\times\min(\lvert J\rvert; \lfloor\frac{\lvert O\rvert}{2}\rfloor)$
SC9	For all planning days, assign heavy units from each subspecialty to more than one OT or up to the maximum allowed in the day.	$\lvert W\rvert\times\lvert P\rvert\times\min\left(\sum_{j\in J} v_j; \left\lfloor\frac{\lvert O\rvert}{2}\right\rfloor\right)$ v_j: 1, if subspecialty j has at least one heavy unit
SC10	For each planning day, an ultra-clean OT has been reserved.	$\lvert W\rvert\times\lvert P\rvert$

B. PROGRAMMING SYNTAX PSEUDOCODE OF THE IMPROVED SATURATION DEGREE-BASED CONSTRUCTIVE HEURISTIC

The proposed algorithm is written in Java. Procedures within the algorithm are explained in Algorithm 3 – 8.

Algorithm 3 Improved Saturation Degree-Based Constructive Heuristic Main Method

```
schedule ← initialize empty master plan
sgtaList ← retrieve surgical units
for i = 1 → | sgtaList | do
    Calculate unit score for sgtaList.get(i)
end for
Sort sgtaList by the unit score in descending order
assignToFixedOtUnits(schedule, sgtaList)
assignToPreferredSlots(schedule, sgtaList)
repairCount ← 0
while sgtaList is not empty do
    g ← sgtaList.get(0)
    wpo ← selectSlot(schedule, g)
    hasAssigned ← assign(schedule, g, wpo)
    if NOT hasAssigned then
        if repairCount < maxRepair then
            swap(schedule, g)
            repairCount ← repairCount + 1
        else
            break
        end if
    end if
end while
```

Algorithm 4 assignFixedOtUnits(schedule, sgtaList)

```
for i = 1 → | sgtaLists | do
    g ← sgtaList.get(i)
    if g has fixed Ots then
        for j = 1 → | g.noOfSlotsAllocated | do
            for k = 1 → | weeks | do
                for l = 1 → | g.fixedOTs | do
                    for m = 1 → | weekly days | do
                        wpo.ot ← g.fixedOTs.get(l)
                        wpo.week ← k
                        wpo.day ← m
                        assign(schedule, g, wpo)
                    end for
                end for
            end for
        end for
    end if
end for
```

Algorithm 5 assignToPreferredSlots(schedule, sgtaList)

```
for i = 1 → | sgtaList | do
    g ← sgtaList.get(i)
    if g has preferred slots then
        for j = 1 → g.noOfSlotsAllocated do
            for k = 1 → | weeks | do
                for l = 1 → | g.preferredSlots | do
                    wpo ← g.preferredSlots.get(l)
                    assign(schedule, g, wpo)
                end for
            end for
        end for
    end if
end for
```

Algorithm 6 selectSlot(schedule, g)

```
unitFeasibleSlots ← get all feasible slots for g
if unitFeasibleSlots is empty then
    return null
end if
for i = 1 → | unitFeasibleSlots | do
    Calculate slot score for unitFeasibleSlots.get(i)
end for
Sort unitFeasibleSlots by slot score in descending order
return unitFeasibleSlots.get(0)
```

Algorithm 7 assign(schedule, g, wpo)

```
if wpo is null then
    return false
end if
planDaydx ← wpo.week ∗ | weekly days | + wpo.day
otIdx ← wpo.ot
schedule[planDayIdx][otIdx] ← g
for i = 1 → | sgtaList | do
    Calculate unit score for sgtaList.get(i)
end for
Sort sgtaList by the unit score in descending order
return true
```

Algorithm 8 swap(schedule, g)

```
candidateSwapSlots ← get all possible swaps for g
for i = 1 → | candidateSwapSlots | do
    Calculate the swap score for candidateSwapSlots.get(i)
end for
Sort candidateSwapSlots by swap score in descending order
wpo ← candidateSwapSlots.get(0)
sgtaList.add(wpo.unit)
assign(schedule, g, wpo)
```

C. UPPER BOUNDARY FOR THE NORMALIZATION OF A SLOT'S OBJECTIVE VALUE

Since $SC1$, $SC4$, $SC5$, $SC6$, and $SC10$ are Boolean objectives (whether they meet the soft constraint or not), the maximum value for the normalization is set to one. The maximum objective value for each slot is listed in Table 21.

TABLE 21. Upper boundary for the normalization of the objective values for each slot.

Soft Constraint	Maximum Value (Per Slot)
SC1	1
SC2	Number of equipment required by the unit
SC3	Number of equipment required by the unit
SC4	1
SC5	1
SC6	1
SC7	Number of slots per day
SC8	Number of slots per day
SC9	Number of slots per day
SC10	1

REFERENCES

[1] J. H. May, W. E. Spangler, D. P. Strum, and L. G. Vargas, "The surgical scheduling problem: Current research and future opportunities," *Prod. Oper. Manage.*, vol. 20, no. 3, pp. 392–405, May 2011, doi: 10.1111/j.1937-5956.2011.01221.x.

[2] I. Rahimi and A. H. Gandomi, "A comprehensive review and analysis of operating room and surgery scheduling," *Arch. Comput. Methods Eng.*, vol. 28, no. 3, pp. 1667–1688, May 2021, doi: 10.1007/s11831-020-09432-2.

[3] S. Zhu, W. Fan, S. Yang, J. Pei, and P. M. Pardalos, "Operating room planning and surgical case scheduling: A review of literature," *J. Combinat. Optim.*, vol. 37, no. 3, pp. 757–805, Apr. 2019, doi: 10.1007/s10878-018-0322-6.

[4] D. Gupta, "Surgical suites' operations management," *Prod. Oper. Manage.*, vol. 16, no. 6, pp. 689–700, 2007, doi: 10.3401/poms.

[5] M. Mashkani, H. Gu, D. Thiruvady, and A. T. Ernst, "Minimizing total clinical deterioration in operating theatres," in *Proc. IEEE Symp. Ser. Comput. Intell. (SSCI)*, Canberra, ACT, Australia, Dec. 2020, pp. 2373–2380, doi: 10.1109/SSCI47803.2020.9308264.

[6] N. S. M. Yusoff, S. Alias, and N. M. Mohamad, "Prediction of patient admissions and bed requirement in inpatient department by using system dynamic simulation," *J. Quality Meas. Anal.*, vol. 18, no. 2, pp. 1–12, 2022.

[7] F. Guerriero and R. Guido, "Operational research in the management of the operating theatre: A survey," *Health Care Manage. Sci.*, vol. 14, no. 1, pp. 89–114, Mar. 2011, doi: 10.1007/s10729-010-9143-6.

[8] T. R. Bovim, M. Christiansen, A. N. Gullhav, T. M. Range, and L. Hellemo, "Stochastic master surgery scheduling," *Eur. J. Oper. Res.*, vol. 285, no. 2, pp. 695–711, Sep. 2020, doi: 10.1016/j.ejor.2020.02.001.

[9] B. Spratt and E. Kozan, "An integrated rolling horizon approach to increase operating theatre efficiency," *J. Scheduling*, vol. 24, no. 1, pp. 3–25, Feb. 2021, doi: 10.1007/s10951-020-00655-6.

[10] B. Spratt and E. Kozan, "Waiting list management through master surgical schedules: A case study," *Operations Res. Health Care*, vol. 10, pp. 49–64, Sep. 2016, doi: 10.1016/j.orhc.2016.07.002.

[11] J. Britt, M. F. Baki, A. Azab, A. Chaouch, and X. Li, "A stochastic hierarchical approach for the master surgical scheduling problem," *Comput. Ind. Eng.*, vol. 158, Aug. 2021, Art. no. 107385, doi: 10.1016/j.cie.2021.107385.

[12] R. Shafaei and A. Mozdgir, "Master surgical scheduling problem with multiple criteria and robust estimation," *Scientia Iranica*, vol. 26, no. 1E, pp. 486–502, 2019, doi: 10.24200/sci.2018.20416.

[13] X. Li, N. Rafaliya, M. F. Baki, and B. A. Chaouch, "Scheduling elective surgeries: The tradeoff among bed capacity, waiting patients and operating room utilization using goal programming," *Health Care Manage. Sci.*, vol. 20, no. 1, pp. 33–54, Mar. 2017, doi: 10.1007/s10729-015-9334-2.

[14] R. Aringhieri, D. Duma, P. Landa, and S. Mancini, "Combining workload balance and patient priority maximisation in operating room planning through hierarchical multi-objective optimisation," *Eur. J. Oper. Res.*, vol. 298, no. 2, pp. 627–643, Apr. 2022, doi: 10.1016/j.ejor.2021.07.033.

[15] A. Abedini, W. Li, and H. Ye, "An optimization model for operating room scheduling to reduce blocking across the perioperative process," *Proc. Manuf.*, vol. 10, pp. 60–70, Jan. 2017, doi: 10.1016/j.promfg.2017.07.022.

[16] M. L. Penn, C. N. Potts, and P. R. Harper, "Multiple criteria mixed-integer programming for incorporating multiple factors into the development of master operating theatre timetables," *Eur. J. Oper. Res.*, vol. 262, no. 1, pp. 194–206, Oct. 2017, doi: 10.1016/j.ejor.2017.03.065.

[17] M. Oliveira, F. Visintin, D. Santos, and I. Marques, "Flexible master surgery scheduling: Combining optimization and simulation in a rolling horizon approach," *Flexible Services Manuf. J.*, vol. 34, no. 4, pp. 824–858, Dec. 2022, doi: 10.1007/s10696-021-09422-x.

[18] M. K. M. Razali, A. H. A. Rahman, M. Ayob, R. Jarmin, F. Qamar, and G. Kendall, "Research trends in the optimization of the master surgery scheduling problem," *IEEE Access*, vol. 10, pp. 91466–91480, 2022, doi: 10.1109/ACCESS.2022.3202546.

[19] A. Jebali and A. Diabat, "A chance-constrained operating room planning with elective and emergency cases under downstream capacity constraints," *Comput. Ind. Eng.*, vol. 114, pp. 329–344, Dec. 2017, doi: 10.1016/j.cie.2017.07.015.

[20] R. Aringhieri, P. Landa, P. Soriano, E. Tànfani, and A. Testi, "A two level metaheuristic for the operating room scheduling and assignment problem," *Comput. Operations Res.*, vol. 54, pp. 21–34, Feb. 2015, doi: 10.1016/j.cor.2014.08.014.

[21] L. I. Almaneea and M. I. Hosny, "A two level hybrid bees algorithm for operating room scheduling problem," in *Proc. Sci. Inf. Conf.*, vol. 1, 2019, pp. 272–290, doi: 10.1007/978-3-030-01174-1_21.

[22] A. Moosavi and S. Ebrahimnejad, "Robust operating room planning considering upstream and downstream units: A new two-stage heuristic algorithm," *Comput. Ind. Eng.*, vol. 143, May 2020, Art. no. 106387, doi: 10.1016/j.cie.2020.106387.

[23] S. Wang, W. Rao, and Y. Hong, "A distance matrix based algorithm for solving the traveling salesman problem," *Oper. Res.*, vol. 20, no. 3, pp. 1505–1542, Sep. 2020, doi: 10.1007/s12351-018-0386-1.

[24] N. Dellaert and J. Jeunet, "A variable neighborhood search algorithm for the surgery tactical planning problem," *Comput. Oper. Res.*, vol. 84, pp. 216–225, Aug. 2017, doi: 10.1016/j.cor.2016.05.013.

[25] M. K. M. Razali, A. H. A. Rahman, M. Ayob, R. Jarmin, L. C. Yong, M. Maaya, A. Izaham, and R. A. Rahman, "Saturation degree-based constructive heuristic for master surgery scheduling problem," in *Proc. IEEE Int. Conf. Artif. Intell. Eng. Technol. (IICAIET)*, Kota Kinabalu, Malaysia, Sep. 2023, pp. 12–14.

[26] B. Amaliah, C. Fatichah, and E. Suryani, "A new heuristic method of finding the initial basic feasible solution to solve the transportation problem," *J. King Saud Univ. Comput. Inf. Sci.*, vol. 34, no. 5, pp. 2298–2307, May 2022, doi: 10.1016/j.jksuci.2020.07.007.

[27] A. O. Sigurpalsson, T. P. Runarsson, and R. J. Saemundsson, "Stochastic master surgical scheduling under ward uncertainty," in *Health Care Systems Engineering*. Cham, Switzerland: Springer, 2020, pp. 163–176, doi: 10.1007/978-3-030-39694-7_13.

[28] A. Kumar, A. M. Costa, M. Fackrell, and P. G. Taylor, "A sequential stochastic mixed integer programming model for tactical master surgery scheduling," *Eur. J. Oper. Res.*, vol. 270, no. 2, pp. 734–746, Oct. 2018, doi: 10.1016/j.ejor.2018.04.007.

[29] M. Lalmazloumian, "Robust optimization framework to operating room planning and scheduling in stochastic environment," Ph.D. dissertation, Dept. Mech., Automot., Mater. Eng., Univ. Windsor (Canada), Windsor, ON, Canada, 2017. [Online]. Available: https://www.proquest.com/dissertations-theses/robust-optimization-framework-operating-room/docview/1914884849/se-2?accountid=27931

[30] A. Kheiri, R. Lewis, J. Thompson, and P. Harper, "Constructing operating theatre schedules using partitioned graph colouring techniques," *Health Syst.*, vol. 10, no. 4, pp. 286–297, Oct. 2021, doi: 10.1080/20476965.2020.1796530.

[31] S. Makboul, S. Kharraja, A. Abbassi, and A. E. H. Alaoui, "A two-stage robust optimization approach for the master surgical schedule problem under uncertainty considering downstream resources," *Health Care Manage. Sci.*, vol. 25, no. 1, pp. 63–88, Mar. 2022, doi: 10.1007/s10729-021-09572-2.

[32] S. Makboul, S. Kharraja, A. Abbassi, and A. E. Hilali, "A multi-objective approach for the combined master surgical schedule and surgical case assignment problems," in *Proc. 13th Conf. Internationale De Modelisation, Optimisation Et Simulation (MOSIM)*, Agadir, Maroc, Nov. 2020, pp. 12–14.

[33] A. J. T. Schneider, J. T. van Essen, M. Carlier, and E. W. Hans, "Scheduling surgery groups considering multiple downstream resources," *Eur. J. Oper. Res.*, vol. 282, no. 2, pp. 741–752, Apr. 2020, doi: 10.1016/j.ejor.2019.09.029.

[34] A. Ghasemkhani, R. Tavakkoli-Moghaddam, M. Hamid, and M. Mahmoodjanloo, "An improvement in master surgical scheduling using artificial neural network and fuzzy programming approach," in *IFIP Advances in Information and Communication Technology*. Cham, Switzerland: Springer, 2020, pp. 254–262, doi: 10.1007/978-3-030-57997-5_30.

[35] S. Zhu, W. Fan, T. Liu, S. Yang, and P. M. Pardalos, "Dynamic three-stage operating room scheduling considering patient waiting time and surgical overtime costs," *J. Combinat. Optim.*, vol. 39, no. 1, pp. 185–215, Jan. 2020, doi: 10.1007/s10878-019-00463-5.

[36] Q. Lu, X. Zhu, D. Wei, K. Bai, J. Gao, and R. Zhang, "Multi-phase and integrated multi-objective cyclic operating room scheduling based on an improved NSGA-II approach," *Symmetry*, vol. 11, no. 5, p. 599, Apr. 2019, doi: 10.3390/sym11050599.

[37] M. Oliveira, L. Lubomirska, and I. Marques, "Reallocating operating room time: A Portuguese case," in *Springer Proceedings in Mathematics & Statistics*. Cham, Switzerland: Springer, 2020, pp. 133–145, doi: 10.1007/978-3-030-39694-7_11.

[38] J. Britt, "Stochastic goal programming and a metaheuristic for scheduling of operating rooms," M.S. thesis, Dept. Ind. Manuf. Syst. Eng., Univ. Windsor (Canada), Windsor, ON, Canada, 2016. [Online]. Available: https://www.proquest.com/dissertations-theses/stochastic-goal-programming-metaheuristic/docview/1768024956/se-2?accountid=27931

[39] F. Hooshmand, S. A. MirHassani, and A. Akhavein, "A scenario-based approach for master surgery scheduling under uncertainty," *Int. J. Healthcare Technol. Manage.*, vol. 16, nos. 3–4, p. 177, 2017, doi: 10.1504/ijhtm.2017.088849.

[40] I. V. Sergienko, L. F. Hulianytskyi, and S. I. Sirenko, "Classification of applied methods of combinatorial optimization," *Cybern. Syst. Anal.*, vol. 45, no. 5, pp. 732–741, Sep. 2009, doi: 10.1007/s10559-009-9134-0.

[41] M. Sánchez, J. M. Cruz-Duarte, J. c. Ortíz-Bayliss, H. Ceballos, H. Terashima-Marin, and I. Amaya, "A systematic review of hyper-heuristics on combinatorial optimization problems," *IEEE Access*, vol. 8, pp. 128068–128095, 2020, doi: 10.1109/ACCESS.2020.3009318.

[42] E. G. Talbi, *Metaheuristics: From Design to Implementation*. Hoboken, NJ, USA: Wiley, 2009.

[43] N. Pillay and R. Qu, *Hyper-Heuristics: Theory and Applications*, 1st ed. Cham, Switzerland: Springer, 2018, p. 130.

[44] J. H. Drake, A. Kheiri, E. Özcan, and E. K. Burke, "Recent advances in selection hyper-heuristics," *Eur. J. Oper. Res.*, vol. 285, no. 2, pp. 405–428, Sep. 2020, doi: 10.1016/j.ejor.2019.07.073.

[45] S. Muthuraman and V. P. Venkatesan, "A comprehensive study on hybrid meta-heuristic approaches used for solving combinatorial optimization problems," in *Proc. World Congr. Comput. Commun. Technol. (WCCCT)*, Feb. 2017, pp. 185–190, doi: 10.1109/WCCCT.2016.53.

[46] J. F. Marchesi and M. A. Cavalcanti Pacheco, "A genetic algorithm approach for the master surgical schedule problem," in *Proc. IEEE Conf. Evolving Adapt. Intell. Syst. (EAIS)*, May 2016, pp. 17–21, doi: 10.1109/EAIS.2016.7502366.

[47] D. Zhang, Y. Liu, R. M'Hallah, and S. C. H. Leung, "A simulated annealing with a new neighborhood structure based algorithm for high school timetabling problems," *Eur. J. Oper. Res.*, vol. 203, no. 3, pp. 550–558, Jun. 2010, doi: 10.1016/j.ejor.2009.09.014.

[48] E. K. Burke and S. Petrovic, "Recent research directions in automated timetabling," *Eur. J. Oper. Res.*, vol. 140, no. 2, pp. 266–280, Jul. 2002, doi: 10.1016/s0377-2217(02)00069-3.

[49] M. W. Carter, G. Laporte, and S. Y. Lee, "Examination timetabling: Algorithmic strategies and applications," *J. Oper. Res. Soc.*, vol. 47, no. 3, pp. 373–383, Mar. 1996, doi: 10.1057/jors.1996.37.

[50] D. Kusumawardani, A. Muklason, and V. A. Supoyo, "Examination timetabling automation and optimization using greedy-simulated annealing hyper-heuristics algorithm," in *Proc. 12th Int. Conf. Inf. Commun. Technol. Syst. (ICTS)*, Surabaya, Indonesia, Jul. 2019, pp. 1–6, doi: 10.1109/ICTS.2019.8850932.

[51] A. Soghier and R. Qu, "Adaptive selection of heuristics for assigning time slots and rooms in exam timetables," *Appl. Intell.*, vol. 39, no. 2, pp. 438–450, Sep. 2013, doi: 10.1007/s10489-013-0422-z.

[52] H. Wu, A. Lin, D. Zhang, and C. P. Mukamakuza, "A new time-dependent algorithm for post enrolment-based course timetabling problem," in *Proc. 8th Int. Conf. Adv. Comput. Intell. (ICACI)*, Chiang Mai, Thailand, Feb. 2016, pp. 425–431, doi: 10.1109/ICACI.2016.7449863.

[53] A. L. Bolaji, A. T. Khader, M. A. Al-Betar, and M. A. Awadallah, "University course timetabling using hybridized artificial bee colony with Hill climbing optimizer," *J. Comput. Sci.*, vol. 5, no. 5, pp. 809–818, Sep. 2014, doi: 10.1016/j.jocs.2014.04.002.

[54] N. Leite, F. Melício, and A. C. Rosa, "A fast simulated annealing algorithm for the examination timetabling problem," *Expert Syst. Appl.*, vol. 122, pp. 137–151, May 2019, doi: 10.1016/j.eswa.2018.12.048.

[55] A. K. Mandal, M. N. M. Kahar, and G. Kendall, "Addressing examination timetabling problem using a partial exams approach in constructive and improvement," *Computation*, vol. 8, no. 2, p. 46, May 2020.

[56] T. L. June, J. H. Obit, Y.-B. Leau, J. Bolongkikit, and R. Alfred, "Sequential constructive algorithm incorporate with fuzzy logic for solving real world course timetabling problem," in *Computational Science and Technology*. Kota Kinabalu, Malaysia: Springer, Aug. 2020, pp. 257–267.

[57] S. A. Rahman, A. Bargiela, E. K. Burke, E. Özcan, B. McCollum, and P. Mcmullan, "Adaptive linear combination of heuristic orderings in constructing examination timetables," *Eur. J. Oper. Res.*, vol. 232, no. 2, pp. 287–297, Jan. 2014, doi: 10.1016/j.ejor.2013.06.052.

[58] S. Abdul-Rahman, E. K. Burke, A. Bargiela, B. McCollum, and E. Özcan, "A constructive approach to examination timetabling based on adaptive decomposition and ordering," *Ann. Oper. Res.*, vol. 218, no. 1, pp. 3–21, Jul. 2014, doi: 10.1007/s10479-011-0999-8.

[59] A. Azlan and N. Mohd. Hussin, "Implementing graph coloring heuristic in construction phase of curriculum-based course timetabling problem," in *Proc. IEEE Symp. Comput. Informat. (ISCI)*, Langkawi, Malaysia, Apr. 2013, pp. 25–29, doi: 10.1109/ISCI.2013.6612369.

[60] E. K. Burke, N. Pham, R. Qu, and J. Yellen, "Linear combinations of heuristics for examination timetabling," *Ann. Operations Res.*, vol. 194, no. 1, pp. 89–109, Apr. 2012, doi: 10.1007/s10479-011-0854-y.

[61] M. A. Al-Betar, "A β-hill climbing optimizer for examination timetabling problem," *J. Ambient Intell. Humanized Comput.*, vol. 12, no. 1, pp. 653–666, Jan. 2021, doi: 10.1007/s12652-020-02047-2.

[62] B. A. Aldeeb, N. M. Norwawi, M. A. Al-Betar, and M. Z. B. Jali, "Solving university examination timetabling problem using intelligent water drops algorithm," in *Proc. Int. Conf. Swarm, Evol., Memetic Comput.*, Cham, Switzerland: Springer, 2014, pp. 187–200.

[63] M. A. Ahandani and M. T. V. Baghmisheh, "Memetic algorithms for solving university course timetabling problem," in *Proc. 1st Int. eConf. Comput. Knowl. Eng. (ICCKE)*, Mashhad, Iran, Oct. 2011, pp. 40–44, doi: 10.1109/ICCKE.2011.6413321.

[64] M. A. Ahandani, M. T. V. Baghmisheh, M. A. B. Zadeh, and S. Ghaemi, "Hybrid particle swarm optimization transplanted into a hyper-heuristic structure for solving examination timetabling problem," *Swarm Evol. Comput.*, vol. 7, pp. 21–34, Dec. 2012, doi: 10.1016/j.swevo.2012.06.004.

[65] N. R. Sabar, M. Ayob, R. Qu, and G. Kendall, "A graph coloring constructive hyper-heuristic for examination timetabling problems," *Int. J. Speech Technol.*, vol. 37, no. 1, pp. 1–11, Jul. 2012, doi: 10.1007/s10489-011-0309-9.

[66] N. Pillay and E. Özcan, "Automated generation of constructive ordering heuristics for educational timetabling," *Ann. Operations Res.*, vol. 275, no. 1, pp. 181–208, Apr. 2019, doi: 10.1007/s10479-017-2625-x.

[67] E. K. Burke, R. Qu, and A. Soghier, "Adaptive selection of heuristics for improving exam timetables," *Ann. Oper. Res.*, vol. 218, no. 1, pp. 129–145, Jul. 2014, doi: 10.1007/s10479-012-1140-3.

[68] U. J. Mele, L. M. Gambardella, and R. Montemanni, "A new constructive heuristic driven by machine learning for the traveling salesman problem," *Algorithms*, vol. 14, no. 9, p. 267, Sep. 2021.

[69] H. Abedinnia, C. H. Glock, and A. Brill, "New simple constructive heuristic algorithms for minimizing total flow-time in the permutation flowshop scheduling problem," *Comput. Oper. Res.*, vol. 74, pp. 165–174, Oct. 2016, doi: 10.1016/j.cor.2016.04.007.

[70] P. Smet, A. T. Ernst, and G. V. Berghe, "Heuristic decomposition approaches for an integrated task scheduling and personnel rostering problem," *Comput. Oper. Res.*, vol. 76, pp. 60–72, Dec. 2016, doi: 10.1016/j.cor.2016.05.016.

[71] M. Rocha, J. F. Oliveira, and M. A. Carravilla, "A constructive heuristic for staff scheduling in the glass industry," *Ann. Oper. Res.*, vol. 217, no. 1, pp. 463–478, Jun. 2014, doi: 10.1007/s10479-013-1525-y.

[72] C. Talens, V. Fernandez-Viagas, P. Perez-Gonzalez, and J. M. Framinan, "New efficient constructive heuristics for the two-stage multi-machine assembly scheduling problem," *Comput. Ind. Eng.*, vol. 140, Feb. 2020, Art. no. 106223, doi: 10.1016/j.cie.2019.106223.

[73] H. Li, K. Gao, P.-Y. Duan, J.-Q. Li, and L. Zhang, "An improved artificial bee colony algorithm with Q-learning for solving permutation flow-shop scheduling problems," *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 53, no. 5, pp. 2684–2693, May 2023, doi: 10.1109/TSMC.2022.3219380.

[74] J. Zheng and Y. Wang, "A hybrid bat algorithm for solving the three-stage distributed assembly permutation flowshop scheduling problem," *Appl. Sci.*, vol. 11, no. 21, p. 10102, Oct. 2021.

[75] M. S. Umam, M. Mustafid, and S. Suryono, "A hybrid genetic algorithm and Tabu search for minimizing makespan in flow shop scheduling problem," *J. King Saud Univ. Comput. Inf. Sci.*, vol. 34, no. 9, pp. 7459–7467, Oct. 2022, doi: 10.1016/j.jksuci.2021.08.025.

[76] M. A. Al-Betar, A. T. Khader, and M. Zaman, "University course timetabling using a hybrid harmony search metaheuristic algorithm," *IEEE Trans. Syst. Man, Cybern. C, Appl. Rev.*, vol. 42, no. 5, pp. 664–681, Sep. 2012, doi: 10.1109/TSMCC.2011.2174356.

[77] A. Ahmadi-Javid and P. Hooshangi-Tabrizi, "Integrating employee timetabling with scheduling of machines and transporters in a job-shop environment: A mathematical formulation and an anarchic society optimization algorithm," *Comput. Oper. Res.*, vol. 84, pp. 73–91, Aug. 2017, doi: 10.1016/j.cor.2016.11.017.

[78] B. A. Aldeeb, M. A. Al-Betar, N. M. Norwawi, K. A. Alissa, M. K. Alsmadi, A. A. Hazaymeh, and M. Alzaqebah, "Hybrid intelligent water drops algorithm for examination timetabling problem," *J. King Saud Univ. Comput. Inf. Sci.*, vol. 34, no. 8, pp. 4847–4859, Sep. 2022, doi: 10.1016/j.jksuci.2021.06.016.

[79] G. Zobolas, C. D. Tarantilis, and G. Ioannou, "Exact, heuristic and meta-heuristic algorithms for solving shop scheduling problems," in *Metaheuristics for Scheduling in Industrial and Manufacturing Applications*. Berlin, Germany: Springer, 2008, pp. 1–40.

[80] A. van den Broek d'Obrenan, A. Ridder, D. Roubos, and L. Stougie, "Minimizing bed occupancy variance by scheduling patients under uncertainty," *Eur. J. Oper. Res.*, vol. 286, no. 1, pp. 336–349, Oct. 2020.

[81] B. Cardoen, E. Demeulemeester, and J. Beliën, "Optimizing a multiple objective surgical case sequencing problem," *Int. J. Prod. Econ.*, vol. 119, no. 2, pp. 354–366, Jun. 2009, doi: 10.1016/j.ijpe.2009.03.009.

[82] S. Opricovic and G.-H. Tzeng, "Compromise solution by MCDM methods: A comparative analysis of VIKOR and TOPSIS," *Eur. J. Oper. Res.*, vol. 156, no. 2, pp. 445–455, Jul. 2004, doi: 10.1016/s0377-2217(03)00020-1.

[83] E. T. Yassen, A. Arram, M. Ayob, and M. Z. A. Nazri, "A constructive heuristic for police patrol routing problems," *Pertanika J. Sci. Technol*, vol. 25, pp. 87–96, Jun. 2017.

[84] P. Royston, "Approximating the Shapiro–Wilk W-test for non-normality," *Statist. Comput.*, vol. 2, no. 3, pp. 117–119, Sep. 1992, doi: 10.1007/bf01891203.

[85] M. N. M. Kahar and G. Kendall, "A great deluge algorithm for a real-world examination timetabling problem," *J. Oper. Res. Soc.*, vol. 66, no. 1, pp. 116–133, Jan. 2015, doi: 10.1057/jors.2012.169.

[86] Y. Wang, Y. W. Wu, and N. Xu, "Discrete symbiotic organism search with excellence coefficients and self-escape for traveling salesman problem," *Comput. Ind. Eng.*, vol. 131, pp. 269–281, May 2019, doi: 10.1016/j.cie.2019.04.008.

[87] A. Elhag and E. Özcan, "A grouping hyper-heuristic framework: Application on graph colouring," *Expert Syst. Appl.*, vol. 42, no. 13, pp. 5491–5507, Aug. 2015, doi: 10.1016/j.eswa.2015.01.038.

[88] T. K. Kim, "Understanding one-way ANOVA using conceptual figures," *Korean J. Anesthesiol.*, vol. 70, no. 1, p. 22, 2017, doi: 10.4097/kjae.2017.70.1.22.

[89] S. Lee and D. K. Lee, "What is the proper way to apply the multiple comparison test?" *Korean J. Anesthesiol.*, vol. 71, no. 5, pp. 353–360, Oct. 2018, doi: 10.4097/kja.d.18.00242.

[90] S. A. Rusticus and C. Y. Lovato, "Impact of sample size and variability on the power and type I error rates of equivalence tests: A simulation study," *Practical Assessment, Res., Eval.*, vol. 19, no. 1, p. 11, 2019.

MOHAMAD KHAIRULAMIRIN MD. RAZALI received the Bachelor of Computer Science degree (Hons.) from Universiti Kebangsaan Malaysia (UKM), in 2021, where he is currently pursuing the Ph.D. degree in computer science. His main research interests include combinatorial optimization problems, healthcare scheduling, artificial intelligence, and optimization algorithms.

ABDUL HADI ABD RAHMAN received the Bachelor of Electrical Engineering degree in computer from Universiti Teknologi Malaysia (UTM), in 2006, the Master of Computer Science degree from Universiti Putra Malaysia (UPM), in 2009, and the Ph.D. degree from UTM, in 2016. He had an internship with Tokai University during the Ph.D. study. He is currently with the Center of Artificial Intelligence Technology, Universiti Kebangsaan Malaysia. His research interests include robotics, artificial intelligence, computer vision, and mobile apps.

MASRI AYOB received the Ph.D. degree in computer science from the University of Nottingham, in 2005. She is currently a Professor and a Principal Researcher with the Data Mining and Optimisation Research Group (DMO), Centre for Artificial Intelligence (CAIT), Universiti Kebangsaan Malaysia. She has published more than 150 papers in international journals and at peer-reviewed international conferences. Her main research interests include meta-heuristics, hyper-heuristics, scheduling, and timetabling, especially educational timetabling, healthcare scheduling, and routing problems.

RAZMAN JARMIN received the Medical Doctor (MD) degree from UKM, in 1994. He completed his postgraduate training in general surgery. He obtained a fellowship in hepatobiliary and transplantation surgery from The University of Melbourne, in 2006. He is currently a Professor in faculty of medicine with Universiti Kebangsaan Malaysia Medical Centre, and the Director of the Hospital Canselor Tuanku Mukhriz. He is also a Consultant Surgeon specializing in hepatobiliary and pancreatic surgery. He has published more than 100 articles in journals and proceedings at national and international levels.

LIU CHIAN YONG received the medical and master's degrees in anesthesiology and critical care degree from Universiti Kebangsaan Malaysia (UKM), in 2002 and 2010, respectively, and the master's degree in clinical acupuncture from International Medical University (IMU). He is currently a Senior Lecturer with the Faculty of Medicine, UKM. He is also a Clinical Consultant with UKM Medical Centre. He has an interest in airway surgeries, total intravenous anesthesia, and integrating new technologies in healthcare services. He frequently performs anesthesia for major and complex surgeries with the Hospital Canselor Tuanku Muhriz, which is the teaching hospital of UKM.

MUHAMMAD MAAYA received the B.Sc. degree in medical science from the University of St. Andrews, in 1993, and the medical degree from The University of Manchester, in 1996. He is currently an Associate Professor with the Faculty of Medicine, Universiti Kebangsaan Malaysia, and a Consultant Anesthesiologist with the Hospital Canselor Tuanku Muhriz, Kuala Lumpur, Malaysia. His research interests include difficult airway management and total intravenous anesthesia. He was a FRCA, in 2004. He is also an American Heart Association-Certified Advanced Cardiac Life Support Trainer and a Training Faculty Member, overseeing and monitoring new ACLS providers and instructors. He is one of the section editors of *Malaysian Journal of Anaesthesiology*.

AZARINAH IZAHAM received the medical and master's degrees in anaesthesiology from Universiti Kebangsaan Malaysia (UKM), Kuala Lumpur, Malaysia, in 1999 and 2007, respectively. She is currently an Associate Professor and a Consultant Anesthesiologist with the Faculty of Medicine, UKM. She is also the Head of the Department of Anaesthesiology and Intensive Care and the Chair of the Board of Examination of Anaesthesiology and Intensive Care in Malaysia. She has also authored various publications and proceedings in local and international journals. Her research interests include perioperative liver surgery and transplant and endocrine surgery.

RAHA ABDUL RAHMAN is currently an Associate Professor and a Consultant Anesthesiologist with the Faculty of Medicine, Universiti Kebangsaan Malaysia (UKM), Kuala Lumpur, Malaysia. Her expertise is in anaesthesiology and intensive care. She has authored various publications and proceedings in local and international journals.