Departamento de
Ingeniería Industrial
Universidad de Concepción

GENERACIÓN AUTOMÁTICA DE UNA METAHEURÍSTICA PARA EL PROBLEMA DE ASIGNACIÓN DE CIRUGÍAS ELECTIVAS CONSIDERANDO AFINIDAD Y PREFERENCIAS EN EL EQUIPO QUIRÚRGICO

Por: Francisco Alejandro Ríos Fierro

Tesis presentada a la Facultad de Ingeniería de la Universidad de Concepción
para optar al grado académico de Magíster en Ingeniería Industrial

Junio 2025
Concepción, Chile

Profesor Guía: Carlos Emilio Contreras Bolton

A mis amigos, familia, y por supuesto, mi perro Coco

ACKNOWLEDGMENTS

This thesis was partially supported by the supercomputing infrastructure of the NLHPC, Chile (CCSS210001). Additional funding was received from ANID through FONDECYT INICIACIÓN 11241132.

Esta tesis fue parcialmente apoyada por la infrastructura de supercómputo del NLHPC (CCSS210001). Se recibió financiamiento adicional de ANID mediante el FONDECYT INICIACIÓN 11241132.

Resumen

La programación de cirugías electivas implica asignar pacientes a quirófanos en días y bloques horarios junto con un equipo quirúrgico. Aunque este problema ha sido ampliamente estudiado, la afinidad humana y las preferencias de recursos de los cirujanos, factores que mejoran el bienestar, acortan la duración de los procedimientos y fomentan un mejor ambiente laboral, rara vez han sido incluidos. En enfoques tradicionales, dichos factores se tratan como restricciones duras en los modelos, exigiendo un nivel mínimo de afinidad en cada equipo. Sin embargo, esta rigidez puede reducir el número de cirugías agendadas, una decisión que podría afectar la salud de un paciente. Para mitigarlo, surgieron modelos con sistemas de penalización por puntajes: cada cirujano dispone de un presupuesto diario que le permite integrarse en equipos mejor afinados y acceder a recursos preferidos, manteniendo el equilibrio en las asignaciones. Estos sistemas son incorporados a los modelos matemáticos. Sin embargo, suelen ser muy pesados computacionalmente, requiriendo largos tiempos de ejecución, realidades que pocos hospitales pueden afrontar sin afectar la práctica clínica. Así, esta tesis propone un framework de generación de un algoritmo metaheurístico, capaz de resolver la asignación de cirugías electivas en plazos cortos y de ofrecer soluciones competitivas. La generación automática explora todo el espacio de parámetros sin necesidad de calibración manual, garantizando una calidad difícil de alcanzar por ajustes convencionales. Para la calibración, se diseñan doce operadores de diversificación, diez de intensificación, dos de destrucción, tres constructores de soluciones iniciales y tres criterios de aceptación. Este conjunto de operadores equilibra búsqueda global y local, aumentando la robustez de la metaheurística en diversos escenarios. Se implementó un framework para llevar a cabo la calibración automática, seleccionando iterativamente las mejores combinaciones de operadores y parámetros. La metaheurística resultante se comparó con modelos matemáticos de la literatura y con metaheurísticas clásicas en tres tiempos limites definidos. Los resultados muestran que el algoritmo generado automáticamente produce mejores soluciones que los modelos matemáticos en el mismo tiempo y se sitúa en primer lugar frente a las metaheurísticas clásicas. Su implementación en un hospital ofrece menores tiempos de cómputo y mayor eficiencia operativa ante la llegada constante de pacientes, además de formar equipos quirúrgicos con mejor afinidad, repercutiendo positivamente en el bienestar del personal y de los pacientes.

Keywords – Surgery scheduling, Metaheuristics, Automathic Algorithm Generation

Abstract

Elective surgery scheduling involves assigning patients to operating rooms on specific days and time blocks, together with a surgical team. Although this problem has been widely studied, human affinity and surgeons' resource preferences—factors that improve well-being, shorten procedure durations and foster a better working environment—are rarely included. In traditional approaches, these factors are treated as hard constraints in the models, requiring a minimum level of affinity in each team. However, this rigidity can reduce the number of scheduled surgeries, a decision that could impact a patient's health. To mitigate this, scoring-based penalty models have emerged: each surgeon is given a daily budget that allows them to join better-matched teams and access preferred resources while maintaining balance in the assignments. These systems are incorporated into mathematical models but are often computationally heavy, requiring long execution times—realities that few hospitals can afford without affecting clinical practice. Thus, this thesis proposes a framework for generating a metaheuristic algorithm capable of solving the elective surgery assignment problem within short time frames and offering competitive solutions. The automatic generation explores the entire parameter space without the need for manual calibration, guaranteeing a quality difficult to reach with conventional tuning. For calibration, twelve diversification operators, ten intensification operators, two destruction operators, three initial-solution constructors and three acceptance criteria are designed. This set of operators balances global and local search, increasing the metaheuristic's robustness across diverse scenarios. A framework was implemented to perform automatic calibration, iteratively selecting the best combinations of operators and parameters. The resulting metaheuristic was compared with mathematical models from the literature and with classical metaheuristics under three defined time limits. The results show that the automatically generated algorithm produces better solutions than the mathematical models in the same time and ranks first compared to classical metaheuristics. Its implementation in a hospital offers shorter computation times and greater operational efficiency in the face of a constant influx of patients, as well as forming surgical teams with better affinity, positively impacting the well-being of both staff and patients.

Keywords – Surgery scheduling, Metaheuristics, Automatic Algorithm Configuration

Contents

ACKNOWLEDGMENTS i

Resumen ii

Abstract iii

1 Introduction 1
- 1.1 Motivation . . . 1
- 1.2 Hypothesis . . . 4
- 1.3 Objectives . . . 4
  - 1.3.1 General Objective . . . 4
  - 1.3.2 Specific Objectives . . . 5
- 1.4 Document outline . . . 5

2 Elective surgery scheduling problem considering affinity and preferences 6
- 2.1 Overall context . . . 6
- 2.2 Problem description . . . 7
- 2.3 Mathematical model formulation . . . 11
- 2.4 Literature Review . . . 14

3 Methodology 18
- 3.1 Framework to generate metaheuristic algorithms . . . 18
  - 3.1.1 Simulated annealing . . . 19
  - 3.1.2 Iterated local search . . . 20
  - 3.1.3 Tabu search . . . 20
  - 3.1.4 Variable neighborhood search . . . 20
  - 3.1.5 Greedy randomized adaptive search procedure . . . 21
- 3.2 Template components . . . 21
  - 3.2.1 Solution representation and evaluation function . . . 22
  - 3.2.2 Initial solutions . . . 23
  - 3.2.3 Diversification operators . . . 26
  - 3.2.4 Intensification operators . . . 27
  - 3.2.5 Destruction operators . . . 29
  - 3.2.6 Acceptance criteria . . . 29
- 3.3 Automatic parameter tuning with `irace` . . . 30

4 Computational results **32**
4.1 Instances . . . 32
4.2 Experimental setup . . . 33
4.3 Parameterization of the metaheuristic . . . 33
4.4 Analysis of the `irace` calibration . . . 34
4.5 Automatically generated metaheuristic vs. mathematical models . . . 38
4.6 Automatically generated metaheuristic vs. primitive metaheuristics . 41
4.7 Statistical analysis . . . 45
4.8 Generalization and applicability . . . 46

5 Conclusions **49**

References **52**

List of Tables

1 Nomenclature used in the definition of the ILP model. 13

2 Benchmark instance characteristics. 33
3 `irace` parameter configuration 35
4 Gap (%) comparison for ILP, CP and MH at 60 s, 300 s and 600 s . . 40
5 Gap (%) to BKS (60 s) 42
6 Gap (%) to BKS (300 s) 42
7 Gap (%) to BKS (600 s) 48

List of Figures

1 Empty schedule. 9
2 Feasible schedule. 9

3 Solution representation for the example shown in Figure 2. 22

4 Operator probabilities given by `irace` 38
5 Gap (%) comparison between ILP and MH. 39
6 Gap (%) comparison between CP and MH. 40
7 Comparison of metaheuristic performance. 44
8 Statistical comparison of metaheuristic performance. 45

Chapter 1

Introduction

This chapter presents the motivation for the automatic generation of metaheuristics for the elective surgery scheduling problem, and states the hypothesis and objectives that guide this thesis.

1.1 Motivation

The Elective Surgery Scheduling Problem (ESSP) is a complex combinatorial optimization problem that involves assigning planned (non-emergency) surgical procedures to operating rooms (ORs) within a given planning horizon. ESSP plays a critical role in healthcare operations by directly influencing patient waiting times, resource utilization, and overall system efficiency. Estimates suggest that OR time costs between $22 and $133 per minute, meaning even marginal improvements in scheduling can yield savings of hundreds of thousands, or even millions, of dollars each year (Zaribafzadeh et al., 2023).

Classical models for the ESSP typically account for constraints such as OR's availability, equipment constraints, length-of-stay forecasts, and clinical priorities (Cardoen et al., 2010; Guerriero and Guido, 2011). However, the interpersonal dimension (affinity among surgeons, nurses, and anaesthetists, together with each surgeon's individual preferences) has received little explicit attention, despite its documented impact on team performance, job satisfaction, and patient safety (Kurmann et al., 2012; Sun et al., 2018; Hull et al., 2012).

Despite these findings on the importance of team dynamics and individual preferences,

existing approaches still rely on manual scheduling or static integration of human relational factors (HRF) within models, limiting flexibility and adaptation to evolving hospital requirements. Moreover, traditional scheduling models also overlook the dynamic nature of team interactions, failing to capture shifting relationships and preferences that naturally arise over time. This static approach can lead to suboptimal scheduling decisions, negatively affecting team cohesion and surgeon satisfaction.

After decades of research in surgical scheduling, dating back to foundational works in the 1960s and 1970s (Barnoon and Wolfe, 1968; Magerlein and Martin, 1978), recent studies have begun to incorporate HRF into optimization models. Meskens et al. (2013) introduced affinity considerations within their model, aiming to enhance team cohesion and operational efficiency. Similarly, Dios et al. (2015) developed a decision-support framework that integrates affinity factors to optimize nursing schedules in surgical contexts. More recently, Park et al. (2021), Ghasemi et al. (2023), and Ríos-Fierro et al. (2025) have extended these ideas by explicitly embedding surgeon preferences and human decision-making styles into optimization models, demonstrating notable improvements in OR utilization and staff satisfaction.

Ríos-Fierro et al. (2025) introduce the ESSP considering both affinity and preferences (ESSP-AP). They show that considering affinity, defined as the comfort and effective communication among surgical team members (Meskens et al., 2013), and accommodating chief surgeons' resource preferences can enhance teamwork, reduce errors, and improve outcomes (Kurmann et al., 2012; Sun et al., 2018). Positive interpersonal dynamics have been linked to fewer intraoperative complications, shorter procedures, and faster recoveries (Catchpole et al., 2007; Hull et al., 2012). Preferred accommodation likewise boosts job satisfaction, lowers burnout, and improves retention, benefiting institutional stability and continuity of care (Dimou et al., 2016; Shanafelt and Noseworthy, 2017).

However, a key critique of this work is its reliance on mathematical programming models, which, although effective in optimizing team dynamics, are computationally demanding and impractical for real-time scheduling in a rapidly changing hospital environments. These models' complexity often results in huge computational times, which restricts their utility in dynamic scenarios where scheduling decisions must swiftly adapt to real-time variations and resource availabilities. The inherent complexity arises from the highly combinatorial nature of scheduling, compounded by HRF and surgeon-specific preferences. As highlighted by (Wang et al., 2025),

traditional optimization approaches like these mathematical models struggle to explore the vast solution space and swiftly adapt to evolving operational conditions, emphasizing the urgent need for more flexible and efficient methodologies.

In response to these limitations, metaheuristic methods offer a compelling alternative due to their inherent computational efficiency. The strength of metaheuristics lies in their ability to rapidly navigate extensive solution spaces through intelligent search strategies and adaptive heuristics. Furthermore, automating the parameter tuning and adjusting the behavior of metaheuristics enhances their robustness and ensures consistently high-quality solutions (Glover and Kochenberger, 2003; Talbi, 2009).

Recent advances in automated algorithm design and hyper-heuristics have demonstrated significant potential in dynamically generating tailored metaheuristics, effectively addressing complex scheduling problems (Burke et al., 2009; Martín-Santamaría et al., 2024; Parada, 2025). Such automated frameworks can continuously adapt to changing hospital environments, team compositions, and surgeon preferences, thus providing robust and flexible scheduling solutions without requiring extensive manual intervention or specialized optimization expertise.

To address the difficulty researchers experience when facing manual parametrization while developing metaheuristics, automated algorithm configuration (AAC) has emerged as a robust framework capable of optimizing algorithm parameters in a data-driven manner without extensive manual intervention. AAC systematically identifies high-performing parameter sets across diverse problem instances, employing methods such as racing algorithms, genetic algorithms, and sequential model-based optimization. As reviewed by Schede et al. (2022), AAC methods excel at managing large and heterogeneous configuration spaces, dynamically adjusting parameter choices to changing operational contexts. Among AAC methodologies, `irace` (López-Ibáñez et al., 2016) was specifically chosen due to its robust and effective implementation of the iterated racing procedure. `irace` efficiently explores parameter spaces by sequentially evaluating and discarding parameter sets that underperform compared to others, thus significantly reducing computational overhead and quickly converging toward optimal configurations. In addition, `irace`'s versatility enables it to accommodate complex metaheuristics, making it suitable for the broad range of optimization scenarios encountered in research. Its statistical validation mechanism further ensures the reliability and reproducibility of identified optimal parameters. Thus, the adoption of `irace` facilitates an efficient, systematic, and statistically

rigorous approach to algorithm configuration, ultimately improving both solution quality and efficiency in metaheuristic development.

This research addresses these gaps by proposing an automatic generation framework for metaheuristic algorithms to address the ESSP-AP. Therefore, this thesis proposes a novel hybrid metaheuristic generated automatically using a specialized tool designed for hyperparameter tuning. We utilize components from five metaheuristics: iterated local search (ILS), variable neighborhood search (VNS), tabu search (TS), simulated annealing (SA), and greedy randomized adaptive search procedure (GRASP). Twelve diversification operators, ten intensification operators, three initial solution construction algorithms, and two destruction methods are considered. A metaheuristic template provides the structural foundation for assembling complete algorithms automatically. To evaluate the proposed approach, 15 instances from the literature are used to test the behavior of the automatically generated metaheuristic, simulating several real-life scenarios by varying the number of days in the planning horizon and the available number of surgeons. Computational experiments involved comparisons with baseline metaheuristics to evaluate computing time and solution quality, and with mathematical programming models to assess solution quality. Finally, results show that the automatically generated metaheuristic reaches its best results when the availability of surgeons and days is broad, and struggles a little when the time frame is more constrained. Experiments show that it can reach results that rival the best known solution given by the mathematical models in a fraction of the time, surpassing base metaheuristics in terms of solution quality.

1.2 Hypothesis

This work hypothesizes that an automatically generated hybrid metaheuristic approach for the ESSP-AP will significantly improve the performance, both in computing times and in solution quality, compared to the state-of-the-art.

1.3 Objectives

1.3.1 General Objective

To develop and implement a framework for automatically generated metaheuristic algorithms capable of solving the ESSP-AP, aiming to produce competitive results

in terms of solution quality and computing times compared to the state-of-the-art.

1.3.2 Specific Objectives

1. Review the state-of-the-art for the ESSP-AP.
2. Design a framework based on `irace` for the ESSP-AP, aimed at automating the construction of a hybrid metaheuristic.
3. Implement the designed framework.
4. Evaluate the effectiveness of the automatically generated metaheuristic by comparing its performance metrics with those of the existing programming models and traditional metaheuristics.

1.4 Document outline

The remaining chapters of this thesis are organized as follows. Chapter 2 outlines the ESSP-AP, emphasizing the complexity introduced by affinity among surgical team members and surgeon preferences. Chapter 3 details the automatic generation framework for designing hybrid metaheuristics. Chapter 4 presents the findings of the proposed solution, demonstrating its effectiveness and adaptability compared to traditional methods. Finally, Chapter 5 summarizes the contributions of this thesis, discusses the results and main findings, and outlines potential avenues for future research.

Chapter 2

Elective surgery scheduling problem considering affinity and preferences

This chapter describes the ESSP-AP, from its components to the critical factors influencing effective scheduling decisions, and provides a literature review focused on solving frameworks for the problem.

2.1 Overall context

The ESSP is a complex and central challenge in hospital operations, with direct implications for patient care quality, resource utilization, and overall healthcare system efficiency. It involves coordinating multiple interdependent factors, making it a multifaceted problem that requires careful consideration of both operational constraints and human-centered elements.

For starters, hospitals often face a backlog of elective surgeries, each varying in urgency, complexity, and resource requirements. Prioritizing which surgeries to schedule involves considering factors such as:

- Some elective procedures, while not emergencies, may become urgent if delayed.
- Certain surgeries require specialized equipment or personnel, influencing scheduling decisions.
- Surgeries with higher risk profiles may necessitate additional postoperative care resources.

Hospitals often manage extensive waiting lists for elective surgeries, needing a good approach to prioritize procedures. Effective prioritization ensures patients receive timely care, aligning surgical interventions with clinical needs and resource availability. Furthermore, allocating resources encompasses assigning ORs and surgical teams to scheduled procedures. This process must account for the compatibility of surgical specialties with available ORs, the availability and expertise of surgeons and support staff, and the scheduling of postoperative care facilities. Efficient resource allocation minimizes idle times, reduces scheduling conflicts, and optimizes the use of hospital infrastructure.

Determining the order of surgeries is also fundamental since it is critical to maintaining a smooth workflow within the surgical department. Sequencing decisions consider factors such as the estimated duration of procedures, the need for specific equipment setups, and the coordination of multidisciplinary teams. Proper sequencing reduces patient wait times, minimizes turnover periods between surgeries, and enhances overall surgical throughput.

2.2 Problem description

The ESSP-AP considers that integrating qualitative factors like team affinity and individual preferences benefits the scheduling process. Team affinity refers to the collaborative efficiency among surgical team members, which can influence the success of complex procedures (Meskens et al., 2013). Affinity between surgeons is represented using a Likert scale, where 4 means maximum affinity and 0 means complete lack of it. Thus, an affinity matrix is built taking into account the affinity each chief surgeon declares to feel towards each of their colleagues.

Acknowledging surgeon preferences for specific ORs or time slots can lead to increased job satisfaction and performance (Park et al., 2021). Quantifying these elements allows for a more personalized and effective scheduling system. Like the affinity matrix defined before, a preference matrix can be constructed, representing how much a chief surgeon values working with any resource.

ESSP-AP considers a penalty score system that allows incorporating both affinity and preferences (Ríos-Fierro et al., 2025). In this approach, each chief surgeon is allocated a fixed daily score, which is reduced based on the quality of their assigned teams (affinity) and the extent to which their resource preferences are met. Higher-quality

team formations and greater preference satisfaction result in larger score deductions. A surgeon cannot be scheduled if their remaining score falls below the minimum required to form a team, effectively preventing overuse. As a result, the schedule may include fewer patients than one without a penalty score system, making it crucial to carefully calibrate the daily score allocation so the total number of scheduled patients does not decrease.

The ESSP-AP can be formally defined, given a set of patients requiring surgery (P), which must be scheduled within a planning horizon spanning a set of days (D), each divided into 16 time-slots ($t \in T = \{0, 1, \ldots, 15\}$). Depending on the surgery type, patients are assigned to an OR from a set of O available rooms, Each surgical operation requires a specific team of human resources, including two categories of surgeons: chief (S^1) and assistant (S^2), alongside a set of additional surgical team members as nurses and anesthetists, E_{otd} ($\forall o \in O, \forall t \in T, \forall d \in D$). The objective is to maximize patients' priority value.

Given an empty schedule for a specific day, each time slot is assigned a unique identifier, as illustrated in Figure 1. A set of patients is awaiting scheduling, alongside a pool of surgeons who may serve as either chief or assistant surgeons. Figure 2 shows an example of a feasible daily schedule, where 23 patients have been scheduled, some requiring surgeries that span multiple time slots. The scheduling process adheres to several critical constraints: a surgeon cannot participate in two surgeries simultaneously, nor can they serve as both chief and assistant in the same procedure. Additionally, surgeries requiring multiple time slots must be scheduled in consecutive blocks, and the same surgical team must be assigned throughout the entire procedure.

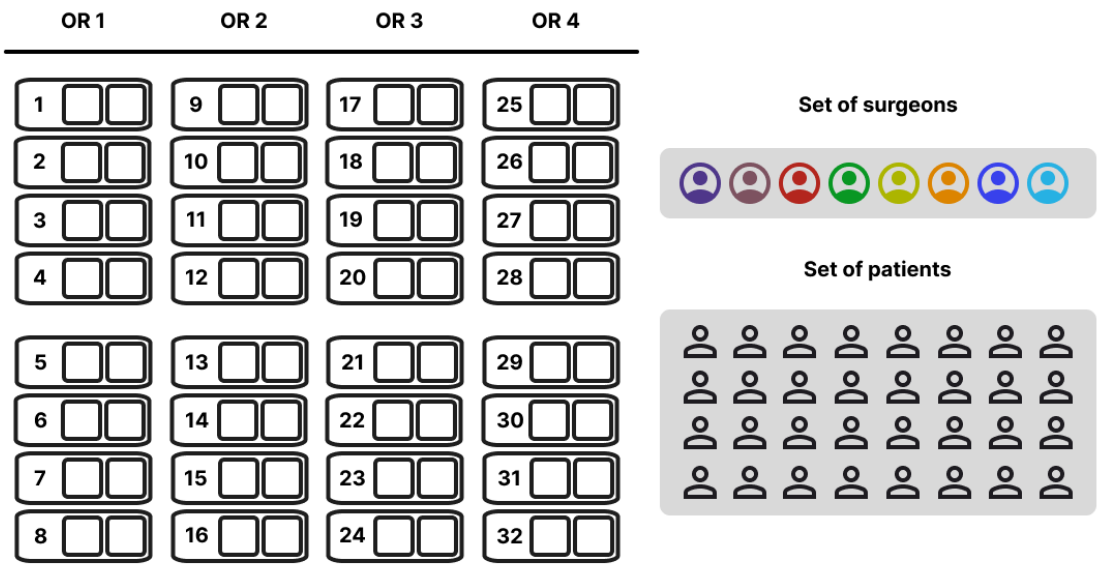


Figure 1: Empty schedule.

This is an empty schedule for an hypothetical eight time-block, one day schedule.

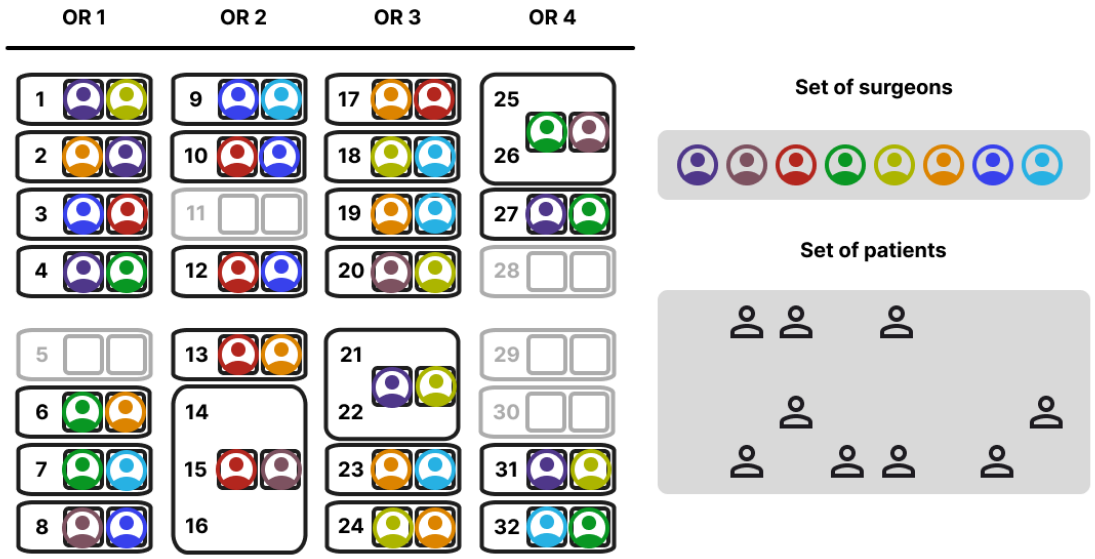


Figure 2: Feasible schedule.

In this solution example, there's some empty time-blocks that didn't get any patient scheduled. This may have happened for a variety of reasons, including lack of available surgeons, patients needing specific, non-available ORs, patients having a surgery duration higher than what's allowed by the space that's left, etc. Grouped blocks, like 14-15-16, mean that a patient's surgery duration takes multiple time-blocks.

The priority value of patient p on day d, denoted as I_{pd}, $\forall p \in P, \forall d \in D$, is proposed

by Ríos-Fierro et al. (2025) and calculated using Equation (1). Where τ_p represents the waiting time of patient p, σ_p is the age of each patient p, and γ is a parameter serving as the weighting factor that helps to balance the priority values so they are integers close to 1. The function $\lceil x \rceil$ denotes the smallest integer greater than or equal to x (the ceiling of x).

$$I_{pd} = 1 + \left\lceil \frac{\gamma \times \tau_p \times \sigma_p}{d} \right\rceil \tag{1}$$

This priority value is used as a criterion to systematically escalate a patient's urgency as their waiting time τ_p increases. By explicitly incorporating both the duration of waiting and patient age σ_p, this format dynamically adjusts the priority, progressively elevating patients who have been on the waiting list for longer periods or who are older. This ensures a structured progression in priority assignment, inherently preventing situations in which patients stagnate indefinitely at lower priorities. Consequently, this approach guarantees that no patient remains overlooked, which mitigates the risk of prolonged waiting periods.

Ríos-Fierro et al. (2025) also propose a formula to get a reasonable approximation for the amount of score to give each surgeon, which is presented in Equation (2). Where $\hat{A}$ means the maximum affinity score possible in the Likert scale (in this study, 4); $|T|$ is the number of time slots in a day; $|O|$ is the number of available ORs; $\hat{E}$ is the number of team members participating in each surgery (related to affinity); $\hat{R}$ is the number of available resources per surgery (related to preferences); $|S^1|$ is the number of available surgeons; and δ is a parameter serving as the adjust factor of the daily score. The function $\lfloor x \rfloor$ denotes the largest integer less than or equal to x (the floor of x). The square root of $|S^1|$ in the denominator ensures that, as the pool of surgeons grows, each individual's score decreases sub-linearly, avoiding an excessive drop-off in score per surgeon when $|S^1|$ is large.

$$F = \left\lfloor \frac{\hat{A} \times |T| \times |O| \times \hat{E} \times \hat{R} \times \delta}{\sqrt{|S^1|}} \right\rfloor \tag{2}$$

ESSP-AP has the following assumptions:

- All necessary data for the models are known in advance and are available to be used by the person responsible for making decisions regarding surgery assignment.

- Each OR is prepared to attend to patients with a specific type of required surgery according to the time slot. Patients whose surgical needs differ from what the OR offers cannot be operated on in this OR during that time.
- For each OR and time slot, a dedicated team of nurses, anesthetists, medical supplies, medical equipment, and/or any other key member involved in performing surgery is assigned. This means that all necessary specialists and resources are always available for any surgery.
- Once a surgery starts, it cannot be interrupted.
- Each surgery requires a chief surgeon and an assistant surgeon since most procedures require a chief surgeon supported by an assisting counterpart.
- It is necessary that the chief surgeon's specialty be the same as what the patient requires, but the assistant surgeon may have any specialty.
- Time is discretized in 30-minute intervals.
- It is assumed that surgeries do not experience delays.
- The preparation time of the OR and the cleaning time after the surgery are considered within the total duration.
- The duration of the surgeries is accounted for in terms of time slots rather than hours and minutes.
- For simplicity, affinity between surgeons represents the whole team's affinity.

2.3 Mathematical model formulation

The ESSP-AP can be formulated as an integer linear programming (ILP) model. This model is the same model Ríos-Fierro et al. (2025) implemented, and considers the objective function presented in Equation (3) and the constraints (4)-(30). Table 1 shows the sets, parameters, and decision variables used in the model definition. Therefore, the ILP model for the ESSP-AP reads as follows:

$$\text{maximize} \sum_{p\in P}\sum_{o\in O}\sum_{d\in D} \alpha I_{pd} y_{pod} - \sum_{c\in S^1}\sum_{d\in D} f_{cd}\frac{|D|}{d+1} \tag{3}$$

subject to:

$$\sum_{o\in O}\sum_{c\in S^1}\sum_{a\in S^2}\sum_{d\in D} x_{pocatd} \leq 1 \quad \forall p \in P, \forall t \in T \tag{4}$$

$$\sum_{p\in P}\sum_{c\in S^1}\sum_{a\in S^2} x_{pocatd} \leq 1 \quad \forall o \in O, \forall t \in T, \forall d \in D \tag{5}$$

$$\sum_{p\in P}\sum_{o\in O}\sum_{c\in S^1} x_{pocatd} \leq 1 \quad \forall a \in S^2, \forall t \in T, \forall d \in D \tag{6}$$

$$\sum_{p\in P}\sum_{o\in O}\sum_{a\in S^2} x_{pocatd} \leq 1 \quad \forall c \in S^1, \forall t \in T, \forall d \in D \tag{7}$$

$$\sum_{p\in P}\sum_{o\in O}\sum_{t\in T}\sum_{d\in D} \beta_{ca} x_{pocatd} \leq 0 \quad \forall c \in S^1, \forall a \in S^2 \tag{8}$$

$$\sum_{o\in O}\sum_{d\in D} y_{pod} \leq 1 \quad \forall p \in P \tag{9}$$

$$\sum_{c\in S^1}\sum_{a\in S^2} z_{pca} \leq 1 \quad \forall p \in P \tag{10}$$

$$\sum_{o\in O}\sum_{d\in D} y_{pod} \leq \sum_{c\in S^1}\sum_{a\in S^2} z_{pca} \quad \forall p \in P \tag{11}$$

$$\sum_{o\in O} x_{pocatd} \leq D_{catd} z_{pca} \quad \forall p \in P, \forall c \in S^1, \forall a \in S^2, \forall t \in T, \forall d \in D \tag{12}$$

$$\sum_{o\in O}\sum_{c\in S^1}\sum_{a\in S^2}\sum_{t\in T} x_{pocatd} \leq |T| J_{pd} \quad \forall p \in P, \forall d \in D \tag{13}$$

$$\sum_{o\in O}\sum_{a\in S^2}\sum_{t\in T}\sum_{d\in D} x_{pocatd} \leq |T| G_{pc} \quad \forall p \in P, \forall c \in S^1 \tag{14}$$

$$\sum_{c\in S^1}\sum_{a\in S^2} x_{pocatd} \leq Q_{potd} \quad \forall p \in P, \forall o \in O, \forall t \in T, \forall d \in D \tag{15}$$

$$\sum_{o\in O}\sum_{t\in T}\sum_{d\in D} x_{pocatd} \geq L_p z_{pca} \quad \forall p \in P, \forall c \in S^1, \forall a \in S^2 \tag{16}$$

$$t_p^s \leq t \sum_{o\in O}\sum_{c\in S^1}\sum_{a\in S^2}\sum_{d\in D} x_{pocatd} + |T| \left(1 - \sum_{o\in O}\sum_{c\in S^1}\sum_{a\in S^2}\sum_{d\in D} x_{pocatd}\right)$$
$$\forall p \in P, \forall t \in T \tag{17}$$

$$t_p^f \geq t \sum_{o\in O}\sum_{c\in S^1}\sum_{a\in S^2}\sum_{d\in D} x_{pocatd} \quad \forall p \in P, \forall t \in T \tag{18}$$

$$t_p^s = t_p^f - L_p + 1 \quad \forall p \in P \tag{19}$$

$$\sum_{o\in O}\sum_{c\in S^1}\sum_{a\in S^2} x_{poca\left[\frac{|T|}{2}-1\right]d} \leq 1 - \sum_{o\in O}\sum_{c\in S^1}\sum_{a\in S^2} x_{poca\left[\frac{|T|}{2}\right]d}$$
$$\forall p \in P, \forall d \in D \tag{20}$$

$$\sum_{t\in T}\sum_{s\in S^1}\sum_{a\in S^2} x_{posatd} \leq T_p |O| y_{pod} \quad \forall p \in P, \forall d \in D, \forall o \in O \tag{21}$$

$$\sum_{o\in O}\sum_{d\in D} y_{pod} \leq 1 \quad \forall p \in P \tag{22}$$

$$f_{c0} = 0 \quad \forall c \in S^1 \tag{23}$$

$$f_{cd} = f_{cd-1} + F - \sum_{p \in P} \sum_{o \in O} \sum_{a \in S^2} \sum_{t \in T} \left(A_{ca} + \sum_{e \in E_{otd}} A_{ce} + P_{ct} + P_{cd} \right) \frac{1}{L_p} x_{pocatd}$$
$$\forall c \in S^1, \forall d \in D \tag{24}$$

$$x_{pocatd} \in \{0,1\} \quad \forall p \in P, \forall o \in O, \forall c \in S^1, \forall a \in S^2, \forall t \in T, \forall d \in D \tag{25}$$

$$y_{pod} \in \{0,1\} \quad \forall p \in P, \forall o \in O, \forall d \in D \tag{26}$$

$$z_{pca} \in \{0,1\} \quad \forall p \in P, \forall c \in S^1, \forall a \in S^2 \tag{27}$$

$$t_p^s \in \mathbb{Z}^+ \quad \forall p \in P \tag{28}$$

$$t_p^f \in \mathbb{Z}^+ \quad \forall p \in P \tag{29}$$

$$f_{cd} \in Z^+ \quad \forall c \in S^1, \forall d \in D \tag{30}$$

Table 1: Nomenclature used in the definition of the ILP model.

Sets	
P	Set of patients awaiting surgery (indexed by p).
O	Set of available ORs for surgeries (indexed by o).
S^1	Set of chief surgeons (indexed by c).
S^2	Set of assistant surgeons (indexed by a).
T	Set of time slots repeated each day (indexed by t).
D	Set of days in the planning horizon (indexed by d).
$\overline{D}$	Set of days in the planning horizon with an additional day before the initial day, $\overline{D} = D \cup \{0\}$.
E_{otd}	Set of other surgical team members, nurses and anesthetists, present in the OR o in the time slot t of day d (indexed by e), $\forall o \in O, \forall t \in T, \forall d \in D$.
Parameters	
α	Weighing factor in the objective function that represents how much more important is to schedule a patient compared to penalizing surgeon's unused scores.
I_{pd}	Priority of patient p on day d, $\forall p \in P, \forall d \in D$.
β_{ca}	Takes value 1 if chief surgeon c is the same as assistant surgeon a, 0 otherwise, $\forall c \in S^1, \forall a \in S^2$.
D_{catd}	Takes value 1 if chief surgeon c and assistant surgeon a are available in the time slot t of day d, 0 otherwise, $\forall c \in S^1, \forall a \in S^2, \forall t \in T, \forall d \in D$.
J_{pd}	Takes the value 1 if patient p is available for service in day d, 0 otherwise, $\forall p \in P$, $\forall d \in D$.
G_{pc}	Takes the value 1 if patient p can be operated on by chief surgeon c, 0 otherwise, $\forall p \in P, \forall c \in S^1$.
Q_{potd}	Takes the value 1 if patient p can be operated in OR o in the time slot t on day d, 0 otherwise, $\forall p \in P, \forall o \in O, \forall t \in T, \forall d \in D$.
L_p	Operating time for patient p, $\forall p \in P$.
A_{ca}	Affinity score between chief surgeon c and assistant surgeon a, $\forall c \in S^1$, $\forall a \in S^2$.
A_{ce}	Affinity score between chief surgeon c and team member e, $\forall c \in S^1, \forall e \in E$.
P_{ct}	Preference score of chief surgeon c by time slot t, $\forall c \in S^1, \forall t \in T$.
P_{cd}	Preference score of chief surgeon c by day d, $\forall c \in S^1$, $\forall d \in D$.
M	Minimum value of affinity and preference.
F	Daily affinity and preference score of each surgeon at the start of each day in the planning horizon.
Decision variables	
x_{pocatd}	Binary variable equals 1 if patient p is operated in the OR o by chief surgeon c and assistant surgeon a in time slot t of day d, 0 otherwise, $\forall p \in P, \forall o \in O, \forall c \in S^1, \forall a \in S^2, \forall t \in T$, $\forall d \in D$.
y_{pod}	Binary variable equals 1 if patient p is scheduled to be operated in OR o on day d, 0 otherwise, $\forall p \in P, \forall o \in O, \forall d \in D$.
z_{pca}	Binary variable equals 1 if patient p is scheduled to be operated on by chief surgeon c and assistant surgeon a, 0 otherwise, $\forall p \in P, \forall c \in S^1, \forall a \in S^2$.
t_p^s	Integer variable representing the time slot when the operation for patient p starts, $\forall p \in P$.
t_p^f	Integer variable representing the time slot when the operation for patient p finishes, $\forall p \in P$.
f_{cd}	Integer variable representing the remaining score of chief surgeon c on day d, $\forall c \in S^1$, $\forall d \in \overline{D}$.

The objective function (3) is based on Equation (1) and aims to maximize the priority value of the assigned patients while penalizing by subtracting all the unspent score. Constraints (4) ensure that a patient is not assigned to more than one chief surgeon, assistant surgeon, OR, and day other than that of a given time slot. Constraints (5) prevent more than one surgery from being assigned to an OR in the same time slot on the same day. Constraints (6) and (7) state that only one chief surgeon and one

assistant surgeon can be assigned to a surgery, respectively. Constraints (8) prohibit a surgeon from acting as both chief and assistant surgeon in the same operation. Constraints (9) indicate that a patient cannot be assigned to more than one OR and more than one day. Constraints (10) stipulate that a patient cannot be assigned to more than one chief surgeon and more than one assistant surgeon. Constraints (11) interconnect the y and z variables. Constraints (12) ensure only available chief and assistant surgeons can be assigned to a surgery. Constraints (13) ensure that patients can only be operated on if they are available. Constraints (14) state that a chief surgeon can operate on a patient if and only if their specialty matches that required by the patient. Similarly, constraints (15) allow for assigning only a surgery in an OR if it meets the necessary conditions to operate on the patient. Constraints (16) link the x and z variables. Constraints (17)-(19) ensure that surgeries that take more than one time slot are assigned in consecutive slots. Constraints (20) prevent a surgery from being assigned to both a morning and an afternoon slot, even if they are consecutive. Constraints (21) and (22) prevent a surgery from starting in one OR and finishing in another. Constraints (23) establish that the score on the day before the start of the planning, represented by the index 0, is 0. Constraints (24) compute the score invested per day for each surgeon by taking the cumulative score up to the previous day plus the score contributed according to parameter F. Then, the affinity and preference values of all the relationships formed in the surgery assignment are subtracted. In addition, the result is multiplied by L_p^{-1} so that surgeries lasting more than one slot are only considered a score investment. Finally, constraints (25)-(30) define the variable domains.

2.4 Literature Review

This section offers a review of existing literature on ESSP, highlighting gaps specifically related to affinity and preference considerations.

Meskens et al. (2013) is the first known work to introduce the concept of surgical team affinity into the OR scheduling problem. They implement a constraint programming (CP) model incorporating various real-world constraints, such as staff availability and team affinity. Their research aims to optimize OR utilization by minimizing the makespan, reducing overtime hours, and maximizing the affinities among members of each surgical team.

Dios et al. (2015) develop a pre-processing system prior to the execution of an ILP model, referred to as a decision support system. This system filters nurses based on affinity and other relevant criteria. It is capable of generating a scheduling plan up to six months in advance, estimating the most suitable week for each patient's surgery. This allows for the timely coordination of both materials and human resources required for the procedures.

Breuer et al. (2020) develop a centralized optimization framework that addresses both long-term OR block planning and short-term personnel scheduling under uncertainty in surgery durations, staff availability, and emergency arrivals. Their MILP model integrates various factors via weightings, including preference considerations. This approach was tested on data from a major academic medical center, yielding a 68% reduction in overtime and a 13% decrease in day-of-surgery schedule adjustments, with only a 6% average increase in operating costs, while maintaining high utilization and service levels.

Park et al. (2021) propose a novel approach to daily OR scheduling that integrates both surgeon preferences for specific time slots and ORs, as well as the complex coordination required for cooperative surgeries, where multiple surgeons may participate in a single operation over overlapping or staggered intervals. By formulating the problem as an MILP model and proposing a time-efficient algorithm based on bundle and knapsack approaches, they demonstrate significant improvements over manual and traditional scheduling strategies in terms of OR utilization and surgeon satisfaction.

Ghasemi et al. (2023) integrate human-centric variables, particularly the dynamic decision-making styles of surgical staff, into a multi-objective optimization model for the OR scheduling problem. Unlike traditional models that treat human resources as static and interchangeable units, their approach accounts for staff compatibility, experience, efficiency, and the probabilistic switching between primary and backup daily management systems. These concepts are mathematically modeled to minimize incompatibility, maximize team efficiency, and reduce scheduling costs associated with OR utilization and staff waiting time. Additionally, the model addresses sequence-dependent setup times between surgeries and applies a hybrid MILP model with a CP model (MILP-CP) solution method to ensure computational tractability at scale.

Rahimi et al. (2023) address the no-wait open-shop surgical case scheduling problem, which involves assigning surgeries to identical ORs while minimizing makespan and

considering complex transportation times between surgeries. The authors develop both a MILP model for small instances and a novel hybrid metaheuristic based on SA for large-scale cases. Their hybrid approach initiates the search from a high-quality heuristic solution and then applies SA to intensify the search in promising regions. Experimental results demonstrate that the proposed hybrid SA consistently outperforms the classic SA in both solution quality and computational efficiency, particularly on large instances.

Azab et al. (2025) tackle the elective surgery scheduling problem by explicitly incorporating surgeons' preferences for specific ORs and time slots, as well as the coordination of sequential collaborative surgeries under uncertain procedure durations. They propose a bi-objective stochastic MILP model that simultaneously minimizes OR operational costs (including overtime and idle time) and maximizes surgeon satisfaction with their assigned schedules. The model accounts for turnover times between surgeries, mandatory breaks for surgical teams, surgeon unavailability windows, and represents surgical duration variability using a lognormal distribution. Computational experiments on synthetic data demonstrate that the proposed approach yields robust schedules under duration uncertainty, offering multiple trade-off solutions between cost and surgeon satisfaction.

Xue et al. (2025) address the problem of elective surgical scheduling under multiple resource constraints, including ORs, recovery beds, and medical staff—with the aim of minimizing average patient recovery time, staff overtime, and total cost. To this end, a MILP model with a multiobjective function is formulated, and a metaheuristic based on the Honey Badger Algorithm is proposed. Each solution encodes both material and human resource assignments as well as the surgical sequence, and fitness is evaluated by a Nash-based aggregation of the three objectives. Computational experiments using real data from a university hospital demonstrate that their model outperforms existing benchmark algorithms in both efficiency and solution quality.

Ríos-Fierro et al. (2025) introduce the ESSP-AP that simultaneously incorporates two human relation factors within the surgical team, a combination largely absent from the existing literature: affinity and preferences. While previous works have explored these dimensions independently and often treated them as rigid constraints, this research proposes a flexible score-based penalty approach that integrates affinity and preferences directly into the objective function, preserving the feasibility of high-priority surgeries. Three model versions are developed, using both ILP and CP

models to compare the performance of traditional, constraint-based, and penalty-based formulations. Experimental results, based on real data from a Chilean public hospital, demonstrate that integrating HRF not only enhances team cohesion and surgeon satisfaction but does so with negligible impact on the total number of surgeries scheduled. The main difference between their work and this work is that they only used exact methods, which require large amounts of computational power and resolution time. In contrast, this thesis introduce a metaheuristic that is not only set to solve instances in record times, but also is automatically generated so no manual calibration is needed.

Many other works have explored variations of the ESSP that do not consider human relation factors, but incorporate very high-performance metaheuristics. Such works include those made by Landa et al. (2016) with a hybrid local search and neighborhood search, Dellaert and Jeunet (2017) with a VNS, and Ayob and Mahmuda (2023) with a hybrid elitism-based genetic algorithm hyper-heuristic. Nonetheless, constructing effective metaheuristic approaches requires significant expertise and extensive experimentation. Researchers must identify suitable algorithmic components and appropriate methods for combining them efficiently, a process traditionally reliant on intuition and manual trial and error. Despite technological advancements that enable extensive computational experimentation, systematic frameworks or guidelines for automating and streamlining this design task are still scarce. Consequently, researchers explore only a fraction of potential algorithm configurations, possibly overlooking more efficient or problem-specific solutions. Hence, developing automated methodologies to rigorously and systematically explore the combinatorial space of metaheuristic algorithms remains a critical and promising research avenue.

Chapter 3

Methodology

This chapter outlines the methodology used to solve the ESSP-AP, describing the framework to build automatically generated metaheuristic algorithms.

3.1 Framework to generate metaheuristic algorithms

Metaheuristic frameworks provide a structure for creating algorithms that iteratively explore and exploit the solution space. At their core, they alternate between stochastic perturbations, which diversify the search by moving the solution into new regions, and improvement routines, which intensify around promising areas. Acceptance criteria regulate whether new candidates replace the current solution, while occasional destruction steps prevent premature convergence. This modular design allows for the tailoring of each component (diversification, intensification, acceptance criterion, destruction) and their probabilities to balance exploration and exploitation.

The template shown in Algorithm 1 illustrates the general structure of the metaheuristic approach, designed to iteratively enhance the initial solution through controlled diversification and intensification strategies. The process begins with an initial solution that provides a starting point for the search (line 1). Subsequently, the solution undergoes diversification operators (Γ), introducing stochastic alterations to explore new regions of the solution space (line 3). Optionally, intensification operators Δ can be applied to intensively explore the neighborhood of the diversified solution (lines 4-5), guided by a predefined probability ($\mathbb{P}$). The acceptance criterion (Λ)

evaluates the improved solution to determine if it should replace the current solution (line 6). Further diversity is maintained by occasionally applying a destruction operator (Ω), which partially deconstructs the solution, thereby preventing premature convergence and promoting exploration (lines 7-8). The iterative process continues until predefined stopping criteria (maximum iterations or time limits) are met (line 9).

The motor used for running the framework is `irace`, an automatic algorithm configuration tool designed to explore and fine-tune the parameter space of optimization algorithms. It employs iterative racing procedures to discard poorly performing configurations early, focusing computational effort on the most promising candidates. This makes it suitable for calibrating metaheuristics, which often rely on carefully tuned parameters to perform effectively. This work considers five traditional metaheuristics: SA, ILS, TS, VNS, GRASP.

Algorithm 1 Template structure

Input: Diversification operators Γ; Intensification operators Δ; Acceptance criteria Λ; Destruction operators Ω; Initial solutions θ; Operator probability $\mathbb{P}$
Output: Solution s

1: $s \leftarrow$ `initial-solution`(θ, $\mathbb{P}(\theta)$)
2: **repeat**
3: $\quad s \leftarrow$ `diversification`(s, Γ, $\mathbb{P}(\Gamma)$)
4: $\quad$ **if** `random`() $\leq \mathbb{P}(\Delta)$ **then**
5: $\quad\quad s \leftarrow$ `intensification`(s, Δ, $\mathbb{P}(\Delta)$)
6: $\quad s \leftarrow$ `acceptance-criterion`(s, $f(s)$, Λ)
7: $\quad$ **if** `random`()$\leq \mathbb{P}(\Omega)$ **then**
8: $\quad\quad s \leftarrow$ `destruction`(s, Ω, $\mathbb{P}(\Omega)$)
9: **until** stopping criteria met

3.1.1 Simulated annealing

SA is a metaheuristic inspired by the physical annealing process in metallurgy, where a material is heated and then slowly cooled to reduce defects and reach a low-energy state (Kirkpatrick et al., 1983; Cerný, 1985). SA approximates the global optimum of a function by iteratively exploring the solution space: starting from an initial solution, it randomly diversifies the current state to generate a candidate solution, which is accepted if it improves the objective or with a probability decreasing over time (temperature), even if it worsens it. Accepting worse solutions allows SA to escape local minima and explore globally.

3.1.2 Iterated local search

ILS is a technique that iteratively applies intensification procedures to solutions diversified from previously found local optima, effectively exploring a reduced search space of locally optimal solutions (Lourenço et al., 2003, 2018). Starting from an initial solution, ILS applies diversification operators to the current local optimum to escape local minima and then intensifies to find a new local optimum. This new solution is accepted based on an acceptance criterion, allowing the algorithm to perform a biased stochastic walk over the space of local optima. The perturbation must be carefully balanced: too small diversification lead to limited exploration, while too large perturbations reduce the search to random restarts.

3.1.3 Tabu search

TS is a metaheuristic that improves upon traditional local search by using adaptive memory structures (tabu lists) to avoid cycling, the repeated revisiting of previously explored solutions, and to help the search escape from local optima. TS was introduced by Glover (1986) and guides the search process by temporarily forbidding certain moves or solutions, based on their recency or frequency, thus encouraging the exploration of new regions in the solution space. This method balances intensification and diversification strategies, and may accept non-improving moves to avoid stagnation, making it highly effective for complex combinatorial optimization problems. Aspiration criteria can override the tabu status for particularly promising solutions, ensuring both flexibility and systematic search (Glover, 1986, 1997).

3.1.4 Variable neighborhood search

VNS is a metaheuristic optimization method introduced by Mladenović and Hansen (1997), designed to solve combinatorial and global optimization problems by systematically changing neighborhood structures during the search. The key idea is that a local minimum concerning one neighborhood may not be a local minimum for another, and a global minimum is a local minimum across all neighborhoods. VNS explores increasingly distant neighborhoods of the current solution by applying a shaking procedure to generate a new candidate, followed by an intensification to find a local optimum within that neighborhood. If an improved solution is found, it replaces the current one and the search restarts with the first neighborhood;

otherwise, the algorithm moves to the next neighborhood.

3.1.5 Greedy randomized adaptive search procedure

GRASP (Feo and Resende, 1995) is a multi-start metaheuristic optimization method, designed to solve a wide range of combinatorial optimization problems by iteratively combining randomized greedy construction with intensification. Each GRASP iteration consists of two phases. In the constructive phase, a feasible solution is built element by element: at each step, candidate elements are evaluated according to a greedy function, a restricted candidate list is formed using a parameter α to control the level of randomness, and one element is chosen at random from the list. The following intensification phase applies standard neighborhood operators to the constructed solution to reach a local optimum. The best solution found over all iterations is returned. This balance between greedy selection and randomized diversification, followed by intensive local improvement, enables GRASP to explore the solution space effectively and escape poor-quality basins.

3.2 Template components

This section describes the key components of the solution framework implemented to address the elective surgery scheduling problem. The proposed metaheuristic approach utilizes structured procedures for generating initial solutions, exploring the search space through diversification mechanisms, refining solutions via intensification operators, and evaluating schedule quality through an objective function.

First, three heuristics are presented for constructing initial feasible schedules, balancing determinism and randomized diversification to ensure both solution validity and variability. Next, a set of twelve diversification operators is detailed, enabling the exploration of new configurations and the escape from local optima by altering surgeon assignments, time slots, and the pool of scheduled patients. Then, we introduce ten intensification strategies designed to incrementally improve schedule quality by enhancing affinity, optimizing resource utilization, or prioritizing higher-scoring patients. Finally, we define the evaluation function that quantitatively assesses each solution based on multiple operational and human relation factors.

3.2.1 Solution representation and evaluation function

Solution representation for the ESSP-AP in this work is represented by three data structures, capable of capturing all possible representations of the solutions. These data structures are:

- A list of patients, where each position in the list corresponds with the ID of a patient. Each position's value represents the ID of the time-block where the patient has been scheduled, and if the patient has not been assigned a time-block for surgery, then it takes the value -1.
- Two lists of tuples (time-block, surgeon), where each time slot is paired with the ID of the surgeon assigned to it. There is a list for chief surgeons and another for assistant surgeons. If no surgery is scheduled for a time slot, and in consequence no surgeons are assigned to it, then the time slot is paired with the value -1.

Figure 3 presents the structural representation of the example schedule shown in Figure 2. It can be seen that the list of patients contains the ID of the time slot where the patient has been scheduled. For example, patient 1 has been scheduled to time-block 9, whereas patients 2 and 3 have not been scheduled. The surgeon's tuples show the purple surgeon being assigned to time-block 1 as chief surgeon and to time-block 2 as assistant surgeon. Since no surgery is scheduled in time slot 5, no surgeon is allocated. Together, these three data structures can represent any feasible schedule, and, if needed, can also express infeasible ones for alternative ESSP-AP solution methods.

Figure 3: Solution representation for the example shown in Figure 2.

The evaluation function assesses the quality of a given solution representation corresponding to a surgical scheduling by calculating a value based on patient prioritization, surgeon compatibility, score constraints (constraints associated to score management, namely Equation 23 and Equation 24), and scheduling feasibility. For each OR, the function computes a value by rewarding the scheduling of high-priority patients and penalizing violations such as incompatible surgeon assignments, fragmented surgery blocks, incomplete or non-consecutive scheduling of surgeries, etc. Additionally, the function tracks resource usage (the score) for each surgeon across days, applying penalties if score constraints are violated or if resources remain unused. The final score aggregates these rewards and penalties across all ORs and surgeons, normalizing the result against the best known solution (BKS) to provide a relative measure of solution quality.

The evaluation function described in Algorithm 2 must be applied to decode a solution from its encoded solution representation. The input includes the surgeon, patient, room, day, slot sets, and all parameter matrices. The output consists of Ψ^*, the relative gap to the BKS. Line 1 initializes the score matrix $\Gamma_{s,d} \leftarrow 0$. Line 2 sets the global accumulator $\Psi \leftarrow 0$. Lines 3–14 form the outer loop over rooms o, where line 4 resets the room penalty π_o and room score Ψ_o, lines 5–12 iterate over each patient p in room o unpacking (s, a, d_p, t_p) (line 6), updating Γ_{s,d_p} (line 7), adding priority reward to Ψ_o (line 8) and accumulating penalties δ_1, δ_2 when compatibility or duration constraints fail (line 9 and 10). Line 13 subtracts π_o from Ψ_o and line 14 adds Ψ_o to Ψ. Lines 15–18 enforce daily limits by penalizing any $\Gamma_{s,d} > \Lambda$ (line 16) and unused score $\Lambda - \Gamma_{s,d}$ (line 17). Finally, line 18 computes $\Psi^* = 1 - \Psi/\Psi_{\text{BKS}}$ and line 19 returns the relative gap to the BKS Ψ^*.

3.2.2 Initial solutions

Three distinct constructive heuristics were implemented to generate initial feasible solutions for the ESSP-AP: a deterministic heuristic, a randomized greedy heuristic (Feo and Resende, 1995), and a complete random heuristic. When required, the probability of one of the following of getting chosen is given by the expression $\mathbb{P}(\theta)$; furthermore, $\mathbb{P}(\theta_1) + \mathbb{P}(\theta_2) + \mathbb{P}(\theta_3) = 1$.

- `deterministic-heuristic` (θ_1): This heuristic constructs an initial solution by systematically assigning patients to available ORs, days, and time slots. Patients are sorted in descending order based on their priority scores, ensuring

Algorithm 2 Evaluation function

Input: Sets of main surgeons $\mathcal{S}$, secondary surgeons $\mathcal{A}$, patients $\mathcal{P}$, ORs $\mathcal{O}$, days $\mathcal{D}$, time slots $\mathcal{T}$; compatibility matrix $C_{s,p}$; availability tensor $R_{o,d,t}$; priorities ρ_p; durations τ_p; affinities/preferences $\alpha_{s,s'}$, π_s; daily limit Λ; best known solution Ψ_{BKS}.

Output: Normalized evaluation score Ψ^*

Initialize score matrix $\Gamma_{s,d} \leftarrow 0$ for all $s \in \mathcal{S}$, $d \in \mathcal{D}$
$\Psi \leftarrow 0$
for each room $o \in \mathcal{O}$ **do**
    Penalty accumulator $\pi_o \leftarrow 0$, room score $\Psi_o \leftarrow 0$
    for each patient $p \in \mathcal{P}$ scheduled in o **do**
        Let assigned surgeons be (s, a) on day d_p at slot t_p
        $\Gamma_{s,d_p} \leftarrow \Gamma_{s,d_p} + \text{cost}(s, p)$
        $\Psi_o \leftarrow \Psi_o + \rho_p$
        if $C_{s,p} = 0$ or infeasible assignment **then**
            $\pi_o \leftarrow \pi_o + \delta_1$
        if duration exceeded **then**
            $\pi_o \leftarrow \pi_o + \delta_2$
    $\Psi_o \leftarrow \Psi_o - \pi_o$
    $\Psi \leftarrow \Psi + \Psi_o$
for each $s \in \mathcal{S}$, $d \in \mathcal{D}$ **do**
    if $\Gamma_{s,d} > \Lambda$ **then**
        $\Psi \leftarrow \Psi - \delta_3 \left(\Gamma_{s,d} - \Lambda\right)$
    $\Psi \leftarrow \Psi - \delta_4 \left(\Lambda - \Gamma_{s,d}\right)$
$\Psi^* \leftarrow 1 - \dfrac{\Psi}{\Psi_{\text{BKS}}}$

that higher-priority patients are scheduled first. The heuristic evaluates feasible assignments by verifying OR availability, surgeon compatibility, and score constraints for each patient. In particular, it identifies compatible chief and assistant surgeon pairs who are available and have no conflicts, which ensures that the assignment respects surgeon affinity and availability constraints. Once a feasible assignment is found, the patient is scheduled, and the corresponding resources and surgeon schedules are updated accordingly. This deterministic approach guarantees a structured and reproducible initial solution, prioritizing patient urgency and surgeon compatibility.

- `randomized-greedy-heuristic` (θ_2): This heuristic introduces controlled randomness into the initial solution construction process to diversify the search space. Like the deterministic heuristic, the randomized greedy heuristic evaluates feasible assignments based on OR availability, surgeon compatibility, and score constraints. However, instead of always selecting the highest-priority patient, it constructs a restricted candidate list of patients whose priority scores fall within a specified threshold determined by a parameter. Then, a patient is randomly selected from this list, and the heuristic attempts to assign this patient to a feasible OR, day, and time slot. This randomized selection process allows the heuristic to explore diverse initial solutions, potentially leading to better-quality solutions after subsequent local search improvements. The parameter controls the balance between greediness and randomness: the closer to 0, the less randomness, and viceversa, which enables flexibility in solution construction. When chosen, this initial solution activates other parameters, namely: `grasp-alpha`, the parameter that controls the level of randomness when choosing the elements; and `prob-grasp-1`, `prob-grasp-2` and `prob-grasp-3`, which mean different ways of sorting patients to pick them for the schedule (based on their priority, based on their surgery duration or based on their duration-to-priority ratio). Since they are probabilities, `prob-grasp-1` + `prob-grasp-2` + `prob-grasp-3` = 1.

- `complete-random-heuristic` (θ_3): This heuristic generates an initial solution by randomly selecting patients and assigning them to randomly chosen feasible time slots and ORs. For each iteration, the heuristic randomly picks an unscheduled patient, selects a random time slot, and verifies if a feasible assignment can be made considering surgeon availability, OR capacity, and score constraints. If a valid assignment is possible, it is committed; if not,

the heuristic continues sampling new random configurations. This approach does not prioritize any operational objective or patient score, instead aiming to rapidly produce varied feasible solutions that serve as diverse starting points for the metaheuristic.

3.2.3 Diversification operators

Each time the diversification strategy is invoked in Algorithm 1, line 3, one of the twelve predefined diversification procedures is selected and applied to help diversify the solution space and escape local optima. The choice of the diversification procedure is determined by a randomly generated value between 0 and 1, which corresponds to specified probabilities associated with each diversification procedure, denoted by Γ_1 through Γ_{12}. It is important to note that these probabilities are tuning parameters, and their sum is constrained by the condition $\mathbb{P}(\Gamma_1)+\mathbb{P}(\Gamma_2)+\ldots+\mathbb{P}(\Gamma_{11})+\mathbb{P}(\Gamma_{12}) = 1$. Next, the diversification procedures are divided into four groups.

- The first group of perturbations involves swapping surgical team members between scheduled surgeries. Specifically, the `swap-surgeon-chief` (Γ_1) and `swap-surgeon-assistant` (Γ_2) operators exchange primary and secondary surgeons, respectively, between pairs of scheduled patients. These perturbations ensure feasibility by verifying surgeon compatibility, availability, and sufficient score allocation. By altering the surgical team composition, these operators explore alternative team assignments that may better align with surgeon preferences and affinity constraints, which can potentially improve schedule quality.

- The second group of perturbations focuses on repositioning scheduled surgeries within the existing timetable. The `move-patient-1` (Γ_3) operator shifts a patient's surgery to an adjacent time slot within the same day, while the `move-patient-2` (Γ_4) operator moves a surgery to the same time slot on an adjacent day. These perturbations allow the metaheuristic to explore temporal adjustments that could lead to improved resource utilization and better accommodation of surgical team preferences.

- The third group comprises perturbations that modify the set of scheduled patients. The `remove-patient` (Γ_5) operator randomly removes a scheduled patient from the timetable, freeing resources and enabling more beneficial future assignments. Conversely, the `add-patient-1` (Γ_6) and `add-patient-2` (Γ_7)

operators attempt to schedule previously unscheduled patients into available time slots. These perturbations dynamically adjust the patient set, enabling the metaheuristic to explore solutions with varying patient compositions and potentially higher overall priority scores. Finally, the `destroy-and-add-10` (Γ_8) operator combines destructive and constructive strategies by simultaneously removing up to ten randomly selected scheduled patients and subsequently attempting to schedule up to ten unscheduled patients. This perturbation significantly alters the current solution, promoting exploration of diverse regions within the solution space. By periodically applying this operator, the metaheuristic can escape local optima and discover novel scheduling configurations that improve the solution.

- The fourth group consists in affinity and preference score destruction, with the objective of lowering the amount of spent score and thus opening new possibilities of surgeon scheduling. `destroy-score-one` (Γ_9) looks for worse combinations of chief and secondary surgeon for one scheduled surgery, while `destroy-score-all` (Γ_{10}) does this for all the scheduled chief surgeons in the planning horizon. `obliterate-score` (Γ_{11})chooses one scheduled surgeon, and tries to drop their total spent score to zero across al their scheduled surgeries, so the surgeon can be scheduled in more surgeries. Finally, `worst-OR` (Γ_{12}) randomly moves one surgeon to an OR with worse preference score to lower their total spent score as well.

3.2.4 Intensification operators

Ten intensification operators to iteratively improve the quality of the surgical scheduling solution are proposed. Juts like with the diversification procedures, the choice of the intensification procedure is determined by a randomly generated value between 0 and 1, which corresponds to specified probabilities associated with each operator, denoted by Δ_1 through Δ_{10}. It is important to note that these probabilities are tuning parameters, and their sum is constrained by the condition $\mathbb{P}(\Delta_1)+\mathbb{P}(\Delta_2)+\ldots+\mathbb{P}(\Delta_9)+\mathbb{P}(\Delta_{10}) = 1$. The intensification procedures are divided into three groups:

- The first group of intensification operators focuses on improving the affinity of the surgical team assigned to scheduled surgeries. Specifically, the `improve-affinity-chief` (Δ_1) operator evaluates feasible replacements of

the primary surgeon assigned to a patient, seeking alternative surgeons with higher affinity scores while ensuring surgeon availability and score constraints are respected. Similarly, the `improve-affinity-assistant` (Δ_2) operator targets the secondary surgeon, identifying feasible replacements that enhance affinity without violating scheduling constraints.

- The second group of intensification operators aims to optimize the timing of scheduled surgeries. The `bring-forward-day` (Δ_3) operator attempts to move scheduled surgeries to earlier days, evaluating earlier feasible time slots and ORs to improve resource utilization and patient satisfaction. Complementing this, the `bring-forward-all` (Δ_4) operator systematically examines all scheduled surgeries, iteratively attempting to shift each surgery to earlier feasible slots, and thereby enhancing overall schedule compactness. Additionally, the `best-OR` (Δ_5) operator optimizes OR assignments by evaluating alternative ORs for scheduled surgeries, selecting feasible moves that maximize affinity or reduce scheduling costs.

- The third group of intensification operators differ in how they identify and execute exchanges: `swap-patient-1` (Δ_6) uses a first-improvement strategy, scanning until it finds the first feasible swap between a scheduled and an unscheduled patient, executing that single exchange without comparing alternatives; `swap-patient-2` (Δ_7) performs a best-improvement search by evaluating all feasible swaps, computing each unscheduled patient's priority-to-duration ratio, and then carrying out exactly the swap with the highest ratio to maximize priority gain per time unit; `swap-patient-3` (Δ_8) employs a greedy, multi-swap procedure, iterating through each unscheduled patient, committing the first feasible swap found for each, and thus potentially executing multiple swaps in one pass to fill as many open slots as possible. `swap-patient-4` (Δ_9) selects the single unscheduled patient with the highest priority-to-duration ratio, then identifies the scheduled patient with the lowest such ratio whose replacement is feasible, executing exactly one swap that maximizes the immediate improvement in priority usage. Finally, `swap-patient-5` (Δ_{10}) exhaustively evaluates all feasible swaps, calculating the difference in priority-to-duration ratios between pairs of scheduled and unscheduled patients, and then executes the single swap that achieves the greatest positive ratio improvement, ensuring a strictly beneficial exchange.

3.2.5 Destruction operators

Destructor operators apply a strong perturbation to the current solution. For this thesis, destruction operators still ensure feasibility across all important domains (patient-surgeon-OR specialty compatibility, surgeon score validity, etc.). Two destruction operators are proposed:

- `reset-solution` (Ω_1: this operator simply saves the current best solution to an elite list, so the solution isn't lost, and starts again by building a new solution from scratch using one of the initial solution operators (θ_1, θ_2 or θ_3).
- `destroy-or` (Ω_2): this operator wipes out a complete OR in a random day, which is basically a soft-restart, but only for one subset of time-blocks. It's considerably stronger than a diversification, so it is considered a destruction operator.

3.2.6 Acceptance criteria

A candidate solution is considered acceptable if it meets all of the following requirements:

- No surgeon is scheduled in overlapping time-blocks beyond their individual capacity, whether they are chief or assistant.
- The roles of chief surgeons respect specialty qualification.
- Surgeries that take more than one time-block to be completed are scheduled in consecutive time-blocks.
- Surgeries do not start in one OR and finish in another.
- No surgeon has a score budget lower than zero.

However, three acceptance criteria operators were considered:

- `simulated-annealing` (Kirkpatrick et al., 1983): this acceptance criterion implements accepting a worse solution with probability

$$\exp\left(-\frac{\Delta}{T}\right)$$

  where Δ is the increase in cost and T is the current temperature, which is decreased according to a cooling schedule. When `simulated-annealing`

is selected, other operators become available, namely: `sa-temp`, the initial temperature; `sa-alpha`, the cooling-rate; and `temp-intensification`, a special parameter that allows a dynamic calculation of the probability of applying an intensification each iteration, by multiplying the actual probability by the current temperature.

- `iterated-local-search` (Lourenço et al., 2003, 2018): this acceptance criterion allows worse solutions if they are within range of a predefined deviation parameter δ_{ILS}, accepting any solution whose cost increase Δ satisfies $\Delta \leq \delta_{\text{ILS}}$, thereby promoting diversification and helping the search escape local optima. If this acceptance criterion is selected, `ils-extra` is activated, representing δ_{ILS}.
- `none`: it is also an option to simply not implementing a particular acceptance criterion. If this operator is selected, only solutions that are strictly better than the current one will be accepted.

3.3 Automatic parameter tuning with `irace`

`irace` (López-Ibáñez et al., 2016) employs an iterated racing strategy that repeatedly evaluates candidate configurations on increasing subsets of training instances, pruning statistically inferior candidates with non-parametric tests. It concentrates evaluations on the most promising regions of the parameter space, and thus, `irace` discovers robust, high-quality configurations while requiring far less computational effort than exhaustive grids.

`irace` was selected as the backbone of the tuning framework because its sequential racing mechanism naturally allocates effort to the most promising parameter combinations, automatically handles complex conditional dependencies, and provides statistically sound elimination decisions even under noisy performance measurements. This combination of adaptive budget allocation, built-in support for high-dimensional and conditional spaces, and proven efficiency in finding top configurations makes `irace` one of the most suitable method for constructing a reliable, scalable, and reproducible tuning pipeline.

In this work, we tuned (i) perturbation operators, (ii) local-search neighborhoods, (iii) acceptance criteria, and (iv) initial-solution constructors. Parameter domains and conditional constraints were provided to `irace`, considering 10000 experiments,

which returned the configuration that minimized the average objective value (negated to convert the original maximization problem into a minimization) on a subset of the training instances.

Chapter 4

Computational results

This chapter presents the computational results obtained by executing the experiments, including comparing the proposed metaheuristic, the existing mathematical programming models, and the traditional metaheuristics.

4.1 Instances

To test the behavior of the proposed metaheuristic automatically generated, we compared it to a balanced suite of 15 benchmark instances that reproduce the most salient sources of variability encountered in elective surgery scheduling, proposed first by (Ríos-Fierro et al., 2025). Every instance fixes the number of patients to 150, a workload representative of a medium-sized surgical department, but systematically varies two key dimensions:

- **Surgeon pool size** $\in \{12, 18, 24\}$, capturing different levels of staffing pressure.
- **Planning horizon** $\in \{1, 3, 5, 7, 10\}$ days, reflecting increasingly long-term schedules with tighter cumulative resource constraints.

This design (3×5) produces instances that range from highly constrained (few surgeons, short horizon) to comparatively flexible (many surgeons, long horizon). This diversity enables a rigorous assessment of the algorithm's robustness and its ability to adapt to varying capacity conditions.

Table 2: Benchmark instance characteristics.

#	|P|	|S|	|D|
I_1	150	12	1
I_2	150	18	1
I_3	150	24	1
I_4	150	12	3
I_5	150	18	3
I_6	150	24	3
I_7	150	12	5
I_8	150	18	5
I_9	150	24	5
I_{10}	150	12	7
I_{11}	150	18	7
I_{12}	150	24	7
I_{13}	150	12	10
I_{14}	150	18	10
I_{15}	150	24	10

4.2 Experimental setup

Experiments are conducted using realistic elective surgery scheduling scenarios derived from real Chilean hospital data. Both the ILP model and the proposed metaheuristic algorithm were implemented and tested under identical conditions, including the same input data, constraints, and objective functions. The ILP and CP models were solved using a state-of-the-art optimization general-purpose solver, Cplex 12.9 (although Cplex is currently in its version 22.0) for Python 3.10, while all metaheuristic algorithms and mathematical models were implemented in Python 3.10. The automatic configuration tool used is `irace` version 3.5.1, implemented in R 4.0. The computational experiments were conducted on the supercomputing infrastructure of the NLHPC, using a Lenovo ThinkSystem SR645 V3 node with two AMD EPYC 9754 processors with 2.25 GHz, each with 128 cores and running 768 GB RAM. All experiments were run with a single thread using the CentOS Linux 7 operating system (64-bit).

4.3 Parameterization of the metaheuristic

The framework for generating metaheuristic algorithms is built upon a template structure, incorporating a comprehensive set of parameters optimized using

irace. These parameters include probabilities associated with diversification and intensification operators, acceptance criteria parameters, initial solution construction methods, and other relevant components, as outlined in Algorithm 1. The full list of parameters, along with their respective ranges or conditions, and values obtained through irace tuning are detailed in Table 3.

- Each diversificator was assigned a probability parameter ranging from 0.00 to 1.00, determining the likelihood of its application during the diversification phase. These operators include the diversificators defined in the previous section of this work.
- Similarly, intensificators were parameterized with probabilities ranging from 0.00 to 1.00, controlling their frequency of application during the intensification phase.
- The greedy initial solution heuristic's randomness parameter was tuned within the range 0.01–1.00. Additionally, parameters controlling the elite solution set size (integer range 1–10) and probabilities for GRASP variants were also optimized.
- The acceptance criterion parameter selects among three strategies: no acceptance criterion, SA, or ILS. When SA is selected, additional parameters become active, including the initial temperature (range 100.0–2000.0) and the cooling rate (range 0.90–0.999).

4.4 Analysis of the irace calibration

In the fine-tune process by irace, we selected a subset of five instances based on preliminary computational results. The tuning tool utilized the average results of five runs per selected instance to explore various configurations for the algorithm's parameters. Each run had a time limit of 60 seconds. The parameter tuning process consumed approximately 17.10 hours, utilizing 46 threads in the NLHPC supercomputing infrastructure.

The best configuration of the matheuristic's parameters, as shown in Table 3, features a deterministic-favored distribution among the three initial solution strategies: prob-deterministic = 0.52, prob-random = 0.24, prob-greedy = 0.24. The higher weight on the deterministic constructor reflects the algorithm's preference

Table 3: irace parameter configuration

Parameter	Class	Representation	Range	Value
destruct	Destruction	-	(10, 5000)	1393
destruct-type	Destruction	-	{No, Yes}	Yes
reset-solution	Destruction	$\mathbb{P}(\Omega_1)$	(0.00, 1.00)	0.3681
destroy-or	Destruction	$\mathbb{P}(\Omega_2)$	(0.00, 1.00)	0.0835
elite-size	Destruction	-	(1, 10)	10
prob-deterministic	Initial Solution	$\mathbb{P}(\theta_1)$	(0.00, 1.00)	0.5189
prob-greedy	Initial Solution	$\mathbb{P}(\theta_2)$	(0.00, 1.00)	0.2389
prob-random	Initial Solution	$\mathbb{P}(\theta_3)$	(0.00, 1.00)	0.2422
grasp-alpha	Initial Solution	-	(0.00, 1.00)	0.4382
prob-grasp-1	Initial Solution	-	(0.00, 1.00)	0.3027
prob-grasp-2	Initial Solution	-	(0.00, 1.00)	0.2046
prob-grasp-3	Initial Solution	-	(0.00, 1.00)	0.4927
acceptance-criterion	Acceptance Criterion	-	{No, SA, ILS}	SA
sa-temp	Acceptance Criterion	-	(100.0, 2000.0)	1418.3
sa-alpha	Acceptance Criterion	-	(0.90, 0.999)	0.9054
temp-local-search	Acceptance Criterion	-	{Yes, No}	No
ils-extra	Acceptance Criterion	-	(0.00, 0.20)	-
tabu	Acceptance Criterion	-	{No, Yes}	Yes
tabu-len	Acceptance Criterion	-	(1, 10000)	7387
swap-surgeon-chief	Diversification	$\mathbb{P}(\Gamma_1)$	(0.00, 1.00)	0.0397
swap-surgeon-assistant	Diversification	$\mathbb{P}(\Gamma_2)$	(0.00, 1.00)	0.0976
move-patient-1	Diversification	$\mathbb{P}(\Gamma_3)$	(0.00, 1.00)	0.0801
move-patient-2	Diversification	$\mathbb{P}(\Gamma_4)$	(0.00, 1.00)	0.0141
remove-patient	Diversification	$\mathbb{P}(\Gamma_5)$	(0.00, 1.00)	0.0499
add-patient-1	Diversification	$\mathbb{P}(\Gamma_6)$	(0.00, 1.00)	0.1404
add-patient-2	Diversification	$\mathbb{P}(\Gamma_7)$	(0.00, 1.00)	0.1609
destroy-and-add-10	Diversification	$\mathbb{P}(\Gamma_8)$	(0.00, 1.00)	0.0013
destroy-score-all	Diversification	$\mathbb{P}(\Gamma_9)$	(0.00, 1.00)	0.1461
destroy-score-one	Diversification	$\mathbb{P}(\Gamma_{10})$	(0.00, 1.00)	0.1070
obliterate-score	Diversification	$\mathbb{P}(\Gamma_{11})$	(0.00, 1.00)	0.1485
worst-OR	Diversification	$\mathbb{P}(\Gamma_{12})$	(0.00, 1.00)	0.0143
prob-local-search	Intensification	$\mathbb{P}(\Delta)$	(0.00, 1.00)	0.7958
improve-affinity-chief	Intensification	$\mathbb{P}(\Delta_1)$	(0.00, 1.00)	0.0311
improve-affinity-assistant	Intensification	$\mathbb{P}(\Delta_2)$	(0.00, 1.00)	0.1667
bring-forward-day	Intensification	$\mathbb{P}(\Delta_3)$	(0.00, 1.00)	0.0330
bring-forward-all	Intensification	$\mathbb{P}(\Delta_4)$	(0.00, 1.00)	0.1126
best-OR	Intensification	$\mathbb{P}(\Delta_5)$	(0.00, 1.00)	0.1430
swap-patient-1	Intensification	$\mathbb{P}(\Delta_6)$	(0.00, 1.00)	0.1729
swap-patient-2	Intensification	$\mathbb{P}(\Delta_7)$	(0.00, 1.00)	0.1319
swap-patient-3	Intensification	$\mathbb{P}(\Delta_8)$	(0.00, 1.00)	0.0842
swap-patient-4	Intensification	$\mathbb{P}(\Delta_9)$	(0.00, 1.00)	0.0783
swap-patient-5	Intensification	$\mathbb{P}(\Delta_{10})$	(0.00, 1.00)	0.0465

for structured initialization, while the balanced probabilities between random and greedy approaches ensure adequate exploration. The GRASP component (with a mid-range `alpha` = 0.44) is activated in approximately 30% of runs, with the three GRASP restricted candidate lists showing varied preferences: `prob-grasp-3` = 0.49 dominates, while `prob-grasp-1` = 0.30 and `prob-grasp-2` = 0.20 provide complementary diversity.

In terms of acceptance criteria, SA was chosen over ILS or none. The initial temperature is high (`sa-temp` = 1418.3, approximately 71% of the maximum range), while the cooling rate is moderate (`sa-alpha` = 0.91). This configuration creates a controlled exploration phase where uphill moves are accepted early but gradually restricted, balancing diversification with convergence. The configuration enables tabu search (`tabu` = Yes) with a substantial tenure (`tabu-len` = 7387), providing strong short-term memory to prevent cycling.

The twelve diversification operators show a relatively balanced distribution, with probabilities ranging from 0.0013 to 0.1609. The highest-weighted operators are `add-patient-2` (0.1609), `obliterate-score` (0.1485), and `destroy-score-all` (0.1461), suggesting preference for patient addition and score-based destruction mechanisms. While it may sound counterintuitive, destroying affinity and preference scores might return some score budget to a surgeon, giving them the chance to be scheduled again. The near-zero probability for `destroy-and-add-10` (0.0013) confirms that large-scale random perturbations are counterproductive. For intensification, the framework invokes local search after approximately 80% of accepted solutions (`prob-local-search` = 0.80). The intensification operators show varied preferences, with `improve-affinity-assistant` (0.17) and `swap-patient-1` (0.17) receiving the highest probabilities, indicating the importance of surgeon preference optimization and patient reallocation.

Notably, `irace` chose a high tabu tenure of 7387, indicative of the importance placed on preventing cycling behavior. This suggests that the metaheuristic benefits significantly from maintaining extensive solution history to effectively navigate complex search landscapes. Furthermore, the relatively high probabilities assigned to patient reallocation and addition operations highlight the algorithm's emphasis on refining patient scheduling to minimize conflicts and optimize resource usage dynamically.

Another interesting observation is the intermediate randomness selected for the

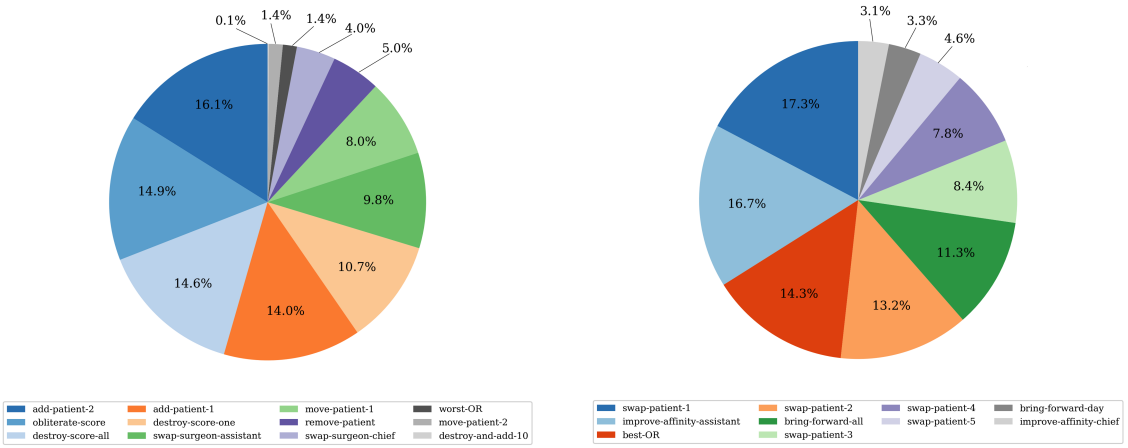
greedy heuristic (grasp-alpha = 0.44). This moderate value indicates that neither purely deterministic nor highly randomized initializations are optimal. Instead, a controlled balance between exploration and exploitation at the early stages of solution construction consistently resulted in improved final outcomes, demonstrating the advantage of maintaining flexibility during initial solution generation.

The systematic parameter tuning performed by irace significantly improved the overall algorithmic performance compared to manual tuning efforts. This automated approach not only provided precise parameter values but also yielded insights into which algorithmic features were essential for achieving competitive results. Consequently, it highlights the value of automatic parameter calibration tools in reducing the burden on human experts and consistently delivering near-optimal configurations tailored to the specifics of the scheduling problems being solved.

In summary, irace consistently promoted mechanisms that balance structured initialization with adaptive diversification:

- The preference for deterministic initialization combined with selective GRASP activation provides a foundation of quality solutions while maintaining necessary diversity.
- SA with moderate temperature and cooling rate, enhanced by tabu memory, creates effective exploration without excessive randomness.
- The probability distributions across operators (θ, ϕ, Γ, Δ) sum to 1.0 within each category, ensuring proper stochastic selection while preventing operator dominance.
- Score-based destruction and surgeon affinity improvements emerged as key mechanisms, reflecting the problem's dual nature of resource optimization and preference satisfaction.

Overall, the calibrated configuration reveals a preference for structured exploration with targeted intensification, where deterministic initialization provides stability, moderate SA parameters ensure controlled diversification, and balanced operator probabilities prevent premature specialization while allowing effective learning. Figure 4 summarizes the probabilities for both diversification and intensification operators given by irace.



(a) Diversification (Γ) probabilities **(b)** Intensification (Δ) probabilities

Figure 4: Operator probabilities given by `irace`

4.5 Automatically generated metaheuristic vs. mathematical models

In this experiment, we compare the performance of the proposed metaheuristic algorithm against the baseline ILP and CP models proposed by (Ríos-Fierro et al., 2025). The primary goal is to evaluate the trade-off between solution quality and computing times.

Table 4 summarizes the comparative results obtained from the ILP model and the proposed automatically generated metaheuristic algorithm, for 60, 300, and 600 seconds. For the proposed metaheuristic, standard deviations are shown between parenthesis next to the mean gap found.

The most immediate observation is the clear dominance of the MH on average. As can be seen in Figures 5 and 6, already after 60 seconds, the metaheuristic records a mean gap of 8.74%, whereas the ILP and CP models are still at 13.30% and 12.15%, respectively. When the time budget is increased the trend is amplified: after 300 seconds the gap of the MH drops to 7.37%, beating ILP (10.95%) and CP 9.25%). After 600 seconds, it reaches 7.29% while the mathematical models stabilize at 10.95% (ILP) and 8.76% (CP). Hence, in aggregate terms the proposed MH delivers solutions that are, on average, 3.5-4.5% closer to the optimum than those produced by the exact methods within the same CPU time.

Increasing the allowance from 60 to 600 seconds reduces the MH gap by only 1.45%,

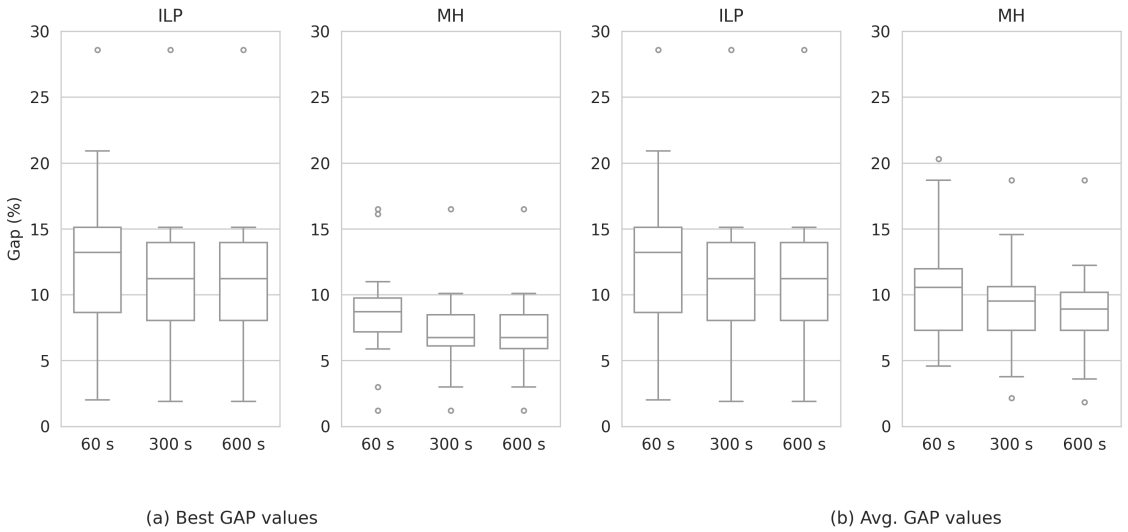


Figure 5: Gap (%) comparison between ILP and MH.

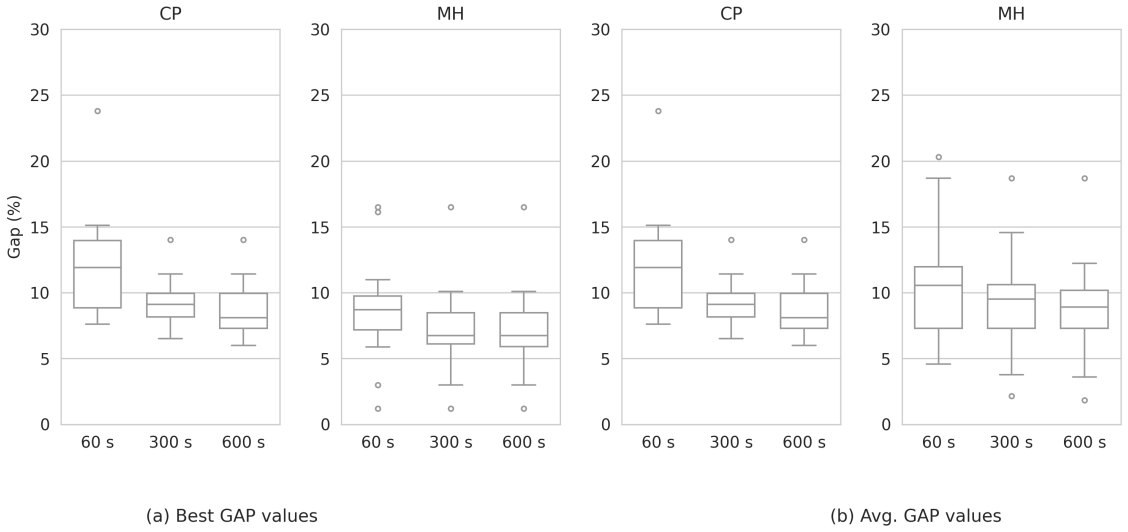
whereas the ILP improves by 2.35 points and CP by 3.39 points. This apparently modest improvement for the metaheuristic hides an important fact: most of the quality is achieved extremely quickly. Figure 7c shows that more than 85% of the total gain obtained by the MH at 600 seconds is already realized within the first 60 seconds. By contrast, the mathematical models exhibit a much slower convergence pattern, spending the first minutes performing preprocessing, root–node cutting and constraint propagation before entering an effective search regime.

Also, the empirical standard deviations confirm the heuristic's steadier behavior, since at 60 seconds the dispersion of the MH gaps (4.02) is already lower than that of ILP (6.46), and the gap narrows further to 3.41 at 300 seconds and 3.44 at 600 seconds, while ILP variability remains unchanged. CP is more stable than ILP but still exhibits larger variance than MH for the two longer time limits. This robustness is very important in hospital scheduling, where reliability often is better than little optimality improvements.

From an operational perspective, these findings strongly support a size-dependent algorithm selection policy. For small, routine scheduling tasks where the number of surgeries or assignments per day is limited, ILP or CP should be favored to guarantee optimality. For larger, real-world instances, or in situations characterized by high variability and late-breaking changes (emergency insertions, staff absences, equipment failures, etc.), the MH is clearly the superior choice. It has the ability to rapidly re-optimize schedules, which enables hospitals to adapt dynamically to unforeseen events, minimizing disruptions and maintaining high resource utilization

Table 4: Gap (%) comparison for ILP, CP and MH at 60 s, 300 s and 600 s

Inst.	60 s			300 s			600 s		
	ILP	CP	MH	ILP	CP	MH	ILP	CP	MH
I_1	2.0	13.1	6.27 (.54)	2.0	9.1	4.67 (.46)	2.0	8.1	4.33 (.46)
I_2	20.9	10.9	5.27 (.60)	3.9	9.9	4.73 (.63)	3.9	9.9	4.46 (.60)
I_3	20.5	8.7	3.21 (.29)	1.9	8.7	2.46 (.42)	1.9	8.7	2.21 (.43)
I_4	28.6	23.8	12.58 (1.31)	28.6	14.0	10.13 (.75)	28.6	11.4	9.37 (.60)
I_5	15.1	15.1	6.05 (.96)	15.1	8.5	4.68 (.92)	15.1	7.9	3.99 (.68)
I_6	14.0	14.0	7.60 (.86)	14.0	9.5	5.20 (.72)	14.0	14.0	4.46 (.64)
I_7	15.1	14.1	6.83 (.90)	15.1	8.8	5.90 (.64)	15.1	7.7	5.46 (.55)
I_8	7.8	7.6	4.92 (.83)	7.8	6.8	4.29 (.72)	7.8	6.0	3.99 (.86)
I_9	8.3	8.3	5.89 (.80)	8.3	7.5	4.39 (.84)	8.3	7.2	3.82 (.86)
I_{10}	11.9	11.9	7.82 (.80)	11.9	10.0	6.86 (.87)	11.9	9.5	6.41 (.82)
I_{11}	9.0	9.0	6.53 (1.29)	9.0	7.8	5.83 (.86)	9.0	7.4	5.23 (.90)
I_{12}	8.3	8.1	6.61 (1.35)	8.3	6.5	4.59 (.83)	8.3	6.1	3.96 (1.31)
I_{13}	13.9	13.9	6.50 (1.05)	13.9	11.4	5.61 (.72)	13.9	10.0	5.45 (.91)
I_{14}	11.2	11.2	6.63 (1.61)	11.2	9.3	4.90 (1.16)	11.2	7.1	4.22 (.80)
I_{15}	13.2	12.5	10.61 (2.66)	13.2	11.0	6.41 (1.59)	13.2	10.4	5.21 (1.42)
Avg	13.3	12.1	6.89 (1.06)	11.0	9.2	5.38 (.81)	11.0	8.4	4.84 (.79)



(a) Best GAP values

(b) Avg. GAP values

Figure 6: Gap (%) comparison between CP and MH.

and patient throughput.

Furthermore, hybrid strategies offer promising directions. For borderline instance sizes, a two-phase approach can be adopted: first, use ILP or CP to generate an initial high-quality solution within a restricted time frame, then use the MH to refine and diversify the solution. The other way around could also be used, since solutions generated by the MH can be provided to exact solvers as high-quality incumbent solutions, potentially accelerating branch-and-bound convergence and improving the final solution.

The practical consequences of this performance extend to resource allocation, software licensing, and broader hospital logistics. Hospitals with limited access to high-performance computing infrastructure or commercial solver licenses could benefit significantly by adopting the MH as their default scheduling engine for large-scale and time-sensitive scenarios. Meanwhile, exact methods can be reserved for benchmarking, policy analysis, or small-batch, high-value optimization tasks.

4.6 Automatically generated metaheuristic vs. primitive metaheuristics

This section has the objective of showing the performance of all the metaheuristics side by side. Tables 5, 6, and 7 report, respectively, the performance of the five baseline metaheuristics and the proposed automatically generated metaheuristic (MH) under time limits of 60, 300, and 600 seconds. For each instance $I_1 - I_{15}$, the columns labeled Best and Avg. denote the best and average percentage gaps to the best known solution (BKS), respectively.

At 60 seconds, ILS achieves the lowest mean best gap (3.77%), followed by the proposed MH (4.48%) and SA (4.75%). MH's mean average gap is 6.89%, essentially on par with SA (6.88%) and better than ILS (6.98%). It also outperforms TS (6.75%), VNS (7.69%), and GRASP (8.13%). Notably, ILS finds a new best-known (negative gap) on I_{14}, and posts best gaps below 5% on eight instances (I_1, I_2, I_3, I_5–I_9).

At 300 seconds, the proposed MH further closes the gap: its mean best gap drops to 3.22%, almost identical to ILS (3.19%) and ahead of SA (3.52%). In terms of average performance, MH leads with 5.38%, outperforming SA (5.58%) and ILS (6.70%). It decisively outstrips TS (6.38%), VNS (7.01%), and GRASP (8.05%).

Table 5: Gap (%) to BKS (60 s)

Inst.	ILS		SA		TS		VNS		GRASP		MH	
	Best	Avg.	Best	Avg.	Best	Avg.	Best	Avg.	Best	Avg.	Best	Avg.
I_1	9.06	14.27	4.02	5.68	8.13	11.80	19.24	19.25	14.28	20.27	4.00	6.27
I_2	5.89	8.99	3.92	5.75	7.04	9.64	14.88	14.89	9.04	14.86	4.90	5.27
I_3	5.78	8.56	2.88	4.49	7.92	9.37	14.59	14.59	8.88	13.58	1.52	3.21
I_4	8.53	13.39	7.67	10.85	10.09	12.06	18.53	19.15	14.91	19.64	9.05	12.58
I_5	2.46	8.63	4.29	6.56	4.07	6.53	12.12	13.36	6.42	11.09	3.65	6.05
I_6	3.30	6.98	3.70	5.49	4.52	6.15	4.83	5.26	5.43	9.86	3.87	7.60
I_7	4.86	6.69	5.02	6.85	6.69	7.34	5.20	6.41	7.32	7.67	5.02	6.83
I_8	3.09	4.81	4.32	5.76	5.48	5.83	2.94	3.74	5.88	6.12	3.57	4.92
I_9	2.18	4.54	4.06	5.96	5.22	5.88	3.68	4.63	5.83	6.10	3.10	5.89
I_{10}	4.93	6.83	6.95	8.12	7.29	8.20	5.45	6.63	8.07	8.51	6.25	7.82
I_{11}	3.21	5.16	4.38	6.76	6.94	7.17	4.56	4.89	6.89	7.35	5.28	6.53
I_{12}	1.27	3.63	2.64	5.46	5.56	5.99	1.93	2.89	5.67	6.15	4.17	6.61
I_{13}	2.92	5.35	6.84	8.75	7.00	8.99	2.94	4.67	7.33	10.13	4.34	6.50
I_{14}	-0.90	2.60	6.14	7.83	7.16	7.93	1.84	2.66	7.33	8.94	3.94	6.63
I_{15}	0.02	4.28	4.36	8.96	8.10	9.32	2.62	3.43	8.75	10.48	4.49	10.61
Avg	3.77	6.98	4.75	6.88	6.75	8.15	7.69	8.43	8.13	10.72	4.48	6.89

Table 6: Gap (%) to BKS (300 s)

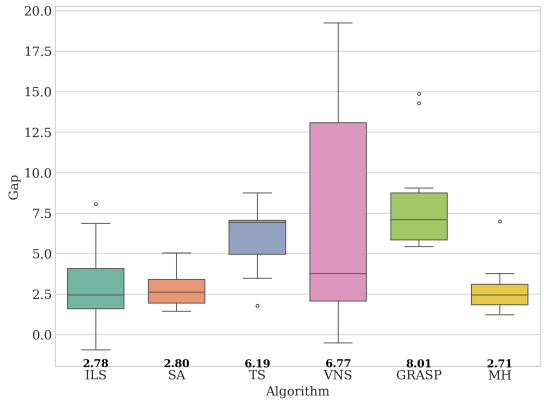
Inst.	ILS		SA		TS		VNS		GRASP		MH	
	Best	Avg.	Best	Avg.	Best	Avg.	Best	Avg.	Best	Avg.	Best	Avg.
I_1	8.06	14.20	4.01	4.53	8.13	11.29	19.24	19.25	14.28	20.24	2.99	4.67
I_2	5.89	8.92	3.91	4.71	7.03	9.04	14.88	14.89	9.04	14.83	3.01	4.73
I_3	4.81	8.63	2.86	3.46	6.95	8.69	14.59	14.59	8.87	13.58	1.46	2.46
I_4	6.86	13.39	6.08	8.70	8.75	11.51	18.53	18.97	14.90	19.64	7.42	10.13
I_5	2.44	7.83	2.54	4.58	3.88	5.60	11.57	13.15	5.91	11.10	1.83	4.68
I_6	2.78	6.79	2.49	4.42	3.97	5.64	4.27	5.06	5.43	9.86	2.05	5.20
I_7	4.86	6.70	4.08	5.86	6.07	7.01	4.72	6.05	7.32	7.67	4.44	5.90
I_8	2.47	4.67	3.21	4.85	4.77	5.64	2.38	3.44	5.85	6.12	2.95	4.29
I_9	2.11	4.48	2.02	4.94	5.00	5.61	3.09	4.26	5.83	6.10	2.17	4.39
I_{10}	4.78	6.82	5.66	7.09	7.07	8.02	5.27	6.21	8.03	8.51	4.84	6.86
I_{11}	2.53	5.00	3.23	5.71	6.89	7.12	3.50	4.63	6.87	7.35	4.17	5.83
I_{12}	0.57	3.47	2.24	4.44	4.92	5.86	1.15	2.35	5.61	6.14	1.61	4.59
I_{13}	1.58	5.01	4.43	7.09	7.00	8.56	1.80	4.15	7.03	10.13	4.22	5.61
I_{14}	-0.94	1.72	2.43	6.42	7.15	7.92	0.62	2.16	7.09	8.94	2.45	4.90
I_{15}	-0.95	2.88	3.54	6.92	8.10	9.02	-0.45	2.09	8.63	10.48	2.68	6.41
Avg	3.19	6.70	3.52	5.58	6.38	7.77	7.01	8.08	8.05	10.71	3.22	5.38

By 600 seconds, MH attains the best overall best gap (2.71%), edging out ILS (2.78%) and SA (2.80%). Its mean average gap falls to 4.84%, marginally better than SA (4.90%) and substantially better than all other baselines.

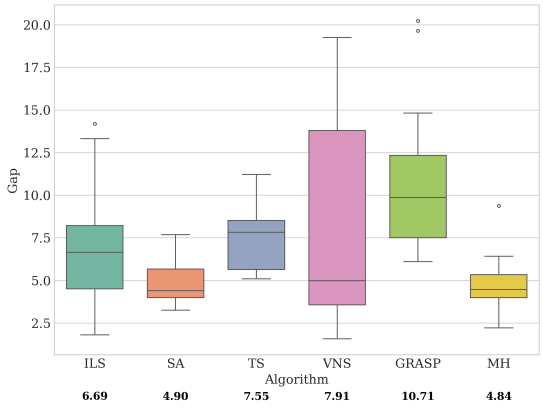
Across all time limits, the proposed MH consistently ranks among the top two in best-gap performance and leads in average-gap from 300 seconds onward. The similarities in results with ILS and SA might come from the fact that they share similar structures; in fact, the proposed MH utilizes a temperature-based system just like SA does. Since the primitive metaheuristics received the same diversification and intensification operators, with the same choosing probabilities assigned to them by `irace` when tuning the automatically generated metaheuristic, it's no surprise that they share behavior as well. On I_{15}, the biggest instance, SA performs significantly better compared to MH, highlighting maybe the biggest limitation of the algorithm: it's considerably heavier to run compared to a simple SA, and thus, can perform less iterations in the complete runtime.

ILS, on the other hand, scored similarly to both SA and MH in terms of best solutions, but in averages, it is considerably worse. This comes from the core structure of ILS, where it needs a nicely diversified solution in order to intensify and get good results. This goes both ways: in some instances, ILS is actually able to get negative gaps, which mean that it found solutions better than the BKS, despite having averages way higher and thus indicating a clear lack of consistency. Furthermore, in other instances like I_1, I_4 and I_5, it gets considerably worse results compared to MH. This can also be checked in that ILS performs way better in big instances, as can be seen in Figure 7d. This happens because big instances has lots of room for easy diversification, since time and resource constraints are not as tight as in smaller instances. Thus, ILS can easily diversify with simple destruction operators, and then do what it does best when intensifying the solution to maximize its potential.

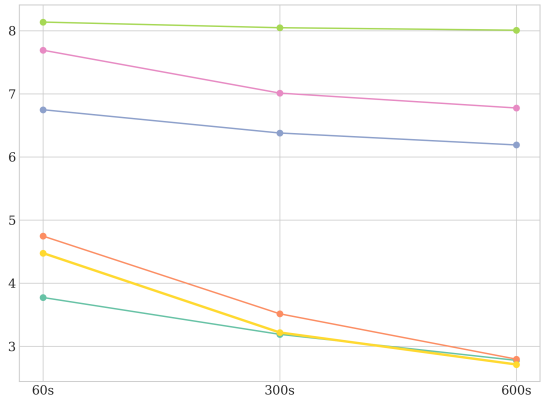
From an operational perspective the findings suggest that hospitals can rely on the automatically generated MH to obtain schedules of comparable quality to state-of-the-art, labor-intensive heuristics, while avoiding the considerable modeling overhead they require. The method's robustness and its ability to deliver high-quality solutions in a matter of seconds can make it attractive for re-scheduling during daily disruptions, thereby enhancing service levels without increasing computational or human resources.



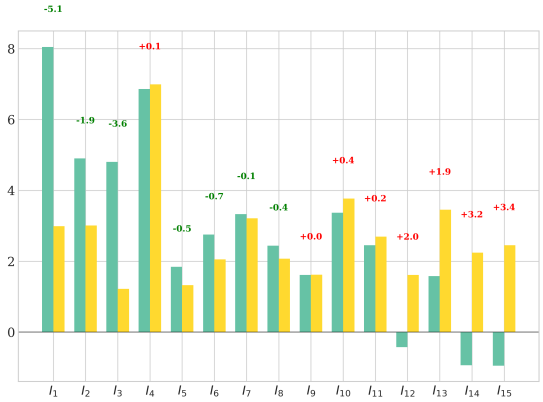
(a) Boxplot of best percentage gap within 600 s. Mean values annotated.



(b) Boxplot of average percentage gap at 600 s. Mean values annotated.



(c) Line plot of average best gap over time limits (60, 300, 600 s).



(d) Instance-wise bar chart comparing ILS vs. MH gaps at 600 s.

Figure 7: Comparison of metaheuristic performance.

4.7 Statistical analysis

To assess and validate the significance of performance differences across the evaluated algorithms, we employed critical difference (CD) diagrams as proposed by Demšar (2006). CD diagrams summarize nonparametric statistical comparisons of multiple algorithms over multiple benchmark instances. Our analysis utilized two performance metrics: the minimum and average gaps to the BKS across all instances.

Initially, the Friedman test (Friedman, 1940) was conducted to determine if any statistically significant differences exist in the overall performance of the algorithms, thereby testing the null hypothesis of equal performance. Upon rejecting this null hypothesis, pairwise comparisons were applied using the Wilcoxon signed-rank test (Wilcoxon, 1945) with Holm's correction (Holm, 1979; García and Herrera, 2008) to control for multiple comparisons. All analyses were performed at a 95% confidence level.

(a) CD diagram for minimum gap (%). **(b)** CD diagram for average gap (%).

Figure 8: Statistical comparison of metaheuristic performance.

Figure 8a illustrates the CD diagram for the minimum gap metric. Here, ILS obtained the best (lowest) average rank, closely followed by MH and SA, all grouped into a non-significant clique, indicating statistically indistinguishable performance among these algorithms. Conversely, algorithms such as GRASP, TS, and VNS demonstrated statistically significant worse performance compared to ILS, MH, and SA, although VNS displayed overlap with MH and SA.

Figure 8b presents the CD diagram based on the average gap metric. The results show that MH and SA were jointly the best-performing algorithms, forming a primary non-significant clique alongside VNS. TS exhibited performance statistically indistinguishable from VNS but significantly worse than MH and SA. Notably, algorithms ILS and GRASP ranked significantly worse across the evaluated instances, highlighting clear statistical differences from the leading methods.

Summarizing, the MH and SA algorithms consistently demonstrated superior, competitive performance across both metrics, while GRASP consistently performed

the poorest, indicating significant potential for method selection based on these empirical insights.

4.8 Generalization and applicability

The proposed metaheuristic has the potential to be used not only in hospital contexts, but also in any scheduling task that involves assigning resources to specific time-blocks. These kind of tasks include:

- Classroom allocation in educational institutions
- Employee shift scheduling in companies
- Project management timelines
- Maintenance scheduling for industrial equipment
- Transportation fleet routing
- Event scheduling

This is because the proposed metaheuristic can be generalizable to any task that includes any number of resource sets that need to be scheduled within a time frame. Operators can adjust parameters such as resource availability, time constraints, and priority criteria to tailor the metaheuristic precisely to their specific scheduling needs. Any evaluation function could be proposed, so depending on the task decision-makers could try to optimize for any desired metric.

Automating the generation of the metaheuristic also has the added benefit of ensuring quick and quality iterations, so adaptability is increased, facilitating prompt responses to changing requirements or unforeseen circumstances. Consequently, decision-makers can efficiently explore diverse scheduling scenarios, leading to better-informed choices and consistently optimized outcomes.

Another significant advantage of using this metaheuristic approach is its scalability. The underlying algorithm can easily accommodate an increased number of tasks, resources, constraints, etc., without a substantial degradation in performance. This allows organizations, regardless of their size or complexity, to apply the metaheuristic effectively, facilitating growth and operational expansion without necessitating frequent redesigns of scheduling processes or the adoption of entirely new methods. Furthermore, the inherent modularity of the algorithm supports

continuous improvement, where individual components can be refined or replaced independently, enhancing long-term sustainability.

Thus, the flexibility, adaptability, scalability, and modularity of the proposed metaheuristic make it a robust solution for a wide range of scheduling problems. It has the potential to produce customized, efficient, and reliable schedules in different contexts, which highlights its practical utility, providing concrete improvements in productivity and resource use across various domains.

Table 7: Gap (%) to BKS (600 s)

Inst.	ILS		SA		TS		VNS		GRASP		MH	
	Best	Avg.	Best	Avg.	Best	Avg.	Best	Avg.	Best	Avg.	Best	Avg.
I_1	8.05	14.19	2.99	4.18	8.13	10.98	19.24	19.25	14.28	20.23	2.99	4.33
I_2	4.90	8.93	3.90	4.40	7.02	8.57	14.88	14.89	9.04	14.82	3.01	4.46
I_3	4.80	8.60	2.86	3.25	6.92	8.33	14.59	14.59	8.87	13.56	1.22	2.21
I_4	6.86	13.31	5.04	7.68	8.75	11.22	17.53	18.76	14.86	19.64	6.99	9.37
I_5	1.84	7.83	1.90	3.81	1.78	5.09	11.57	12.99	5.45	11.10	1.32	3.99
I_6	2.75	6.91	1.98	3.80	3.48	5.30	3.77	4.98	5.43	9.86	2.05	4.46
I_7	3.33	6.65	2.61	5.18	6.07	6.90	4.72	5.92	7.32	7.67	3.21	5.46
I_8	2.44	4.64	2.51	4.16	4.68	5.46	2.37	3.28	5.85	6.12	2.07	3.99
I_9	1.61	4.37	1.80	4.22	5.00	5.52	2.66	4.01	5.83	6.10	1.62	3.82
I_{10}	3.37	6.83	4.54	6.45	7.07	7.86	4.75	6.04	8.03	8.51	3.77	6.41
I_{11}	2.45	5.08	2.68	5.11	6.86	7.07	3.04	4.47	6.87	7.35	2.69	5.23
I_{12}	-0.43	3.40	1.44	3.72	4.92	5.75	0.65	2.10	5.63	6.15	1.61	3.96
I_{13}	1.58	5.02	3.82	6.26	7.02	8.47	1.77	3.85	6.92	10.13	3.45	5.45
I_{14}	-0.94	1.80	2.27	5.53	7.04	7.82	0.59	1.91	7.09	8.94	2.24	4.22
I_{15}	-0.95	2.77	1.61	5.81	8.10	8.92	-0.51	1.57	8.63	10.48	2.45	5.21
Avg	2.78	6.69	2.80	4.90	6.19	7.55	6.77	7.91	8.01	10.71	2.71	4.84

Chapter 5

Conclusions

The experiments conducted in this thesis provide important insights into the comparative performance and operational suitability of the proposed automatically generated metaheuristic relative to both classical metaheuristics (ILS, SA, TS, VNS and GRASP) and exact optimization methods (ILP, CP) in the context of elective surgery scheduling.

First, the results confirm that the proposed automatically generated metaheuristic achieves competitive solution quality and demonstrates remarkable improvement potential with increased computational time. While ranking second among metaheuristics by average gap at the 60 second limit (6.89% gap), MH progresses to first place at 300 seconds (5.38% gap) and consolidates this leadership at 600 seconds (4.84% gap), with an almost negligible 0.06 percentage-point margin over the runner-up SA. This substantial improvement trajectory (29.8% reduction in gap from 60 seconds to 600 seconds) significantly outpaces the modest improvements shown by the other metaheuristics, showing MH's superior learning capability and adaptive potential.

Second, the most compelling aspect of these results is that MH achieves this competitive performance without any human expertise in algorithm design, parameter tuning, or domain-specific customization. The fact that an automatically generated metaheuristic can match or closely approach the performance of carefully crafted, expert-designed algorithms represents a significant advancement in automated optimization. This finding validates the potential of automatic algorithm generation techniques to democratize high-quality optimization solutions across diverse problem domains.

An additional observation worth highlighting is the notable similarity between the performance profiles of MH and SA. This resemblance arises primarily because the automated parameter tuning tool `irace` selected Simulated Annealing as the acceptance criterion for MH. Consequently, the primitive SA metaheuristic evaluated in this thesis was executed using exactly the same acceptance parameters (initial temperature, cooling rate, etc.) that `irace` assigned to the automatically generated MH. This parameter sharing explains the comparable results between MH and SA, emphasizing how algorithmic structure and parameter selection directly influence solution quality and convergence behavior. However, this similarity should not be misinterpreted as an endorsement that the primitive SA is inherently sufficient or universally preferable. Rather, it indicates that SA, with the specific parameters chosen by `irace`, is particularly well-suited to this specific ESSP-AP and the associated set of benchmark instances. In broader contexts or with different scheduling problems, automatically generating a metaheuristic remains advantageous because it systematically explores a broader algorithmic and parameter space, adapting precisely to varying problem structures without human bias or predefined algorithmic assumptions.

Third, when compared to exact methods, MH consistently delivers feasible and high-quality schedules in only a fraction of the time required by ILP and CP on medium and large instances. The algorithm's capacity to generate consistent solutions, combined with its scalability and rapid convergence at longer time horizons, makes it exceptionally well-suited for dynamic and time-sensitive hospital environments where quick decision-making is crucial. Moreover, MH exhibits markedly lower variability than the exact methods: the standard deviation of its average gap shrinks from 1.06 at 60 s to 0.79 at 600 s. Such statistical stability is invaluable in hospital operations, where decision-makers often prefer a solution that is consistently good to one that is occasionally optimal but unreliable.

An additional insight concerns hybridization. Since the proposed metaheuristic can rapidly provide high-quality initial solutions, these can serve as strong incumbents for exact solvers, significantly reducing the branch-and-bound search space and allowing the exact solver to certify optimality faster. This two-phase workflow combines MH's speed with ILP's guarantees and deserves future exploration.

However, the analysis reveals that MH requires sufficient computational time to fully realize its potential, showing more modest performance at very short time

limits. This characteristic suggests that MH's sophisticated search mechanisms need adequate time to explore and exploit the solution space effectively. Additionally, the variability in performance across different instances indicates opportunities for instance-specific adaptations or hybrid approaches.

The practical implications for hospital scheduling are significant. MH provides a compelling alternative that combines competitive solution quality with the substantial advantage of requiring no domain expertise for algorithm development. For practical deployment, a time-adaptive approach that allocates sufficient computational resources to leverage MH's improvement potential would maximize both efficiency and solution quality. The automated nature of the algorithm also means it can be rapidly deployed across different hospital systems without requiring specialized optimization expertise.

Thus, this research demonstrates the transformative potential of automated metaheuristic design in addressing complex real-world optimization problems. The proposed MH algorithm successfully balances solution quality, computational efficiency, and accessibility in ways that are highly aligned with the practical demands of healthcare scheduling. Most importantly, it shows that automated algorithm generation can produce solutions competitive with expert-designed methods, opening new possibilities for widespread adoption of advanced optimization techniques.

Future research should focus on developing adaptive time allocation strategies to maximize MH's improvement potential, exploring instance-specific automatic algorithm generation, and investigating real-time integration within hospital operations. Additionally, extending automatic algorithm generation techniques to other healthcare optimization problems could further validate this approach's broad applicability. The ultimate goal should be creating intelligent, self-adapting scheduling systems that can continuously improve their performance while remaining accessible to healthcare practitioners without optimization expertise.

Bibliography

Ayob, M. and Mahmuda, D. (2023). Elitism-based genetic algorithm hyper-heuristic for solving real-life surgical scheduling problem. In *Proceedings of the 15th Asian Conference on Intelligent Information and Database Systems (ACIIDS 2023)*, volume 1863 of *Communications in Computer and Information Science*, pages 510–523. Springer.

Azab, R., Eltawil, A., and Gheith, M. (2025). A bi-objective stochastic model for operating room scheduling considering surgeons' preferences and collaborative surgeries. *Decision Analytics Journal*, 14:100544.

Barnoon, S. and Wolfe, H. (1968). Scheduling a multiple operating room system: A simulation approach. *Health Services Research*, 3(4):272.

Breuer, D. J., Lahrichi, N., Clark, D. E., and Benneyan, J. C. (2020). Robust combined operating room planning and personnel scheduling under uncertainty. *Operations Research for Health Care*, 27:100276.

Burke, E. K., Hyde, M. R., Kendall, G., Ochoa, G., Ozcan, E., and Woodward, J. R. (2009). Exploring hyper-heuristic methodologies with genetic programming. pages 177–201.

Cardoen, B., Demeulemeester, E., and Belien, J. (2010). Operating room planning and scheduling: A literature review. *European Journal of Operational Research*, 201(3):921–932.

Catchpole, K. R., Giddings, A. E., Wilkinson, M., Hirst, G., Dale, T., and de Leval, M. R. (2007). Improving patient safety by identifying latent failures in successful operations. *Surgery*, 142(1):102–110.

Cerný, V. (1985). Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. *Journal of Optimization Theory and Applications*, 45(1):41–51.

Dellaert, N. and Jeunet, J. (2017). A variable neighborhood search algorithm for the surgery tactical planning problem. *Computers & Operations Research*, 84:216–225.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7:1–30.

Dimou, F. M., Eckelbarger, D., and Riall, T. S. (2016). Surgeon burnout: a systematic review. *Journal of the American College of Surgeons*, 222(6):1230–1239.

Dios, M., Molina-Pariente, J. M., Fernandez-Viagas, V., Andrade-Pineda, J. L., and Framinan, J. M. (2015). A decision support system for operating room scheduling. *Computers & Industrial Engineering*, 88:430–443.

Feo, T. A. and Resende, M. G. C. (1995). Greedy randomized adaptive search procedures. *Journal of Global Optimization*, 6:109–133.

Friedman, M. (1940). A comparison of alternative tests of significance for the problem of m rankings. *The Annals of Mathematical Statistics*, 11(1):86–92.

García, S. and Herrera, F. (2008). An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons. *Journal of Machine Learning Research*, 9(89):2677–2694.

Ghasemi, S., Tavakkoli-Moghaddam, R., and Hamid, M. (2023). Operating room scheduling by emphasising human factors and dynamic decision-making styles: a constraint programming method. *International Journal of Systems Science: Operations & Logistics*, 10(1):2224509.

Glover, F. (1997). Tabu search and adaptive memory programming—advances, applications and challenges. In Barr, R. S., Helgason, R. V., and Kennington, J. L., editors, *Interfaces in Computer Science and Operations Research*, volume 7 of *Operations Research/Computer Science Interfaces Series*. Springer, Boston, MA.

Glover, F. W. (1986). Future paths for integer programming and links to artificial intelligence. *Computers & Operations Research*, 13(5):533–549.

Glover, F. W. and Kochenberger, G. A., editors (2003). *Handbook of Metaheuristics*, volume 57. Springer Science & Business Media, Boston, MA.

Guerriero, F. and Guido, R. (2011). Operational research in the management of the operating theatre: a survey. *Health Care Management Science*, 14:89–114.

Holm, S. (1979). A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6(2):65–70.

Hull, L., Arora, S., Aggarwal, R., Darzi, A., Vincent, C., and Sevdalis, N. (2012). The impact of nontechnical skills on technical performance in surgery: a systematic review. *Journal of the American College of Surgeons*, 214(2):214–230.

Kirkpatrick, S., Gelatt, C. D. J., and Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220(4598):671–680.

Kurmann, A., Tschan, F., K. Semmer, N., Seelandt, J., Candinas, D., and Beldi, G. (2012). Human factors in the operating room – the surgeon's view. *Trends in Anaesthesia and Critical Care*, 2(5):224–227.

Landa, P., Aringhieri, R., Soriano, P., Tànfani, E., and Testi, A. (2016). A hybrid optimization algorithm for surgeries scheduling. *Operations Research for Health Care*, 8:103–114.

Lourenço, H. R., Martin, O. C., and Stützle, T. (2003). Iterated local search. In Glover, F. W. and Kochenberger, G. A., editors, *Handbook of Metaheuristics*,

volume 57 of *International Series in Operations Research & Management Science*, pages 320–353. Springer US, Boston, MA.

Lourenço, H. R., Martin, O. C., and Stützle, T. (2018). Iterated local search: Framework and applications. In Gendreau, M. and Potvin, J.-Y., editors, *Handbook of Metaheuristics*, pages 129–168. Springer International Publishing, Cham.

López-Ibáñez, M., Dubois-Lacoste, J., Pérez-Cáceres, L., Stützle, T., and Birattari, M. (2016). The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives*, 3:43–58.

Magerlein, J. M. and Martin, J. B. (1978). Surgical demand scheduling: a review. *Health Services Research*, 13(4):418.

Martín-Santamaría, R., López-Ibáñez, M., Stützle, T., and Colmenar, J. M. (2024). On the automatic generation of metaheuristic algorithms for combinatorial optimization problems. *European Journal of Operational Research*, 318:740–751.

Meskens, N., Duvivier, D., and Hanset, A. (2013). Multi-objective operating room scheduling considering desiderata of the surgical team. *Decision Support Systems*, 55(2):650–659.

Mladenović, N. and Hansen, P. (1997). Variable neighborhood search. *Computers & Operations Research*, 24(11):1097–1100.

Parada, V. (2025). *Automatic Generation of Algorithms*. CRC Press, Boca Raton, USA.

Park, J., Kim, B.-I., Eom, M., and Choi, B. K. (2021). Operating room scheduling considering surgeons' preferences and cooperative operations. *Computers & Industrial Engineering*, 157:107306.

Rahimi, A., Hejazi, S. M., Zandieh, M., and Mirmozaffari, M. (2023). A novel hybrid simulated annealing for no-wait open-shop surgical case scheduling problems. *Applied System Innovation*, 6(1):15.

Ríos-Fierro, F., Latorre-Núñez, G., and Contreras-Bolton, C. (2025). Surgery scheduling problem considering the affinity and preferences in the surgical team. *In revision*.

Schede, E., Brandt, J., Tornede, A., Wever, M., Bengs, V., Hüllermeier, E., and Tierney, K. (2022). A survey of methods for automated algorithm configuration. *Journal of Artificial Intelligence Research*, 75:425–487.

Shanafelt, T. D. and Noseworthy, J. H. (2017). Executive leadership and physician well-being: nine organizational strategies to promote engagement and reduce burnout. *Mayo Clinic Proceedings*, 92(1):129–146.

Sun, R., Marshall, D. C., Sykes, M. C., Maruthappu, M., and Shalhoub, J. (2018). The impact of improving teamwork on patient outcomes in surgery: A systematic review. *International Journal of Surgery*, 53:171–177.

Talbi, E.-G. (2009). *Metaheuristics: From Design to Implementation.* John Wiley & Sons, Hoboken, NJ.

Wang, Z., Bai, R., and Zhang, T. (2025). Towards constraint-based adaptive hypergraph learning for solving vehicle routing: An end-to-end solution. Preprint.

Wilcoxon, F. (1945). Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83.

Xue, J., Li, Z., and Zhang, S. (2025). Multi-resource constrained elective surgical scheduling with nash equilibrium toward smart hospitals. *Scientific Reports*, 15(1):3946.

Zaribafzadeh, H. M. S., Webster, W. L., Vail, C. J., Daigle, T., Kirk, A. D., Allen, P. J., Henao, R., and Buckland, D. M. (2023). Development, deployment, and implementation of a machine learning surgical case length prediction model and prospective evaluation. *Annals of Surgery*, 278(6):890–895.