UNIVERSITÀ DEGLI STUDI DI SIENA

Dipartimento di Ingegneria dell'Informazione e Scienze Matematiche

Combinatorial models and algorithms for resource allocation problems in healthcare management and logistics

Ilaria Salvadori

PhD in Information Engineering and Science

Supervisor
Prof. Alessandro Agnetis

Co-supervisor
Prof. Marco Pranzo

Examination Commitee
Prof. Domenico Conforti
Prof. Paolo Detti
Prof. Gaia Nicosia

Thesis reviewers
Prof. Domenico Conforti
Prof. Gaia Nicosia

Siena, 31/12/2025

Contents

List of Figures

List of Tables

Acknowledgements

I am deeply grateful to my supervisor, Prof. Alessandro Agnetis, whose guidance has been essential to this work and to my development as a researcher. He has taught me not only the fundamentals of rigorous research but also what it means to approach academic work with integrity and dedication. Beyond his professional expertise, I am sincerely appreciative of his patience and kindness, qualities that have made this challenging journey not only manageable but genuinely enriching. His constant support exemplifies the profound impact that the right supervisor can have on a doctoral experience, and I feel truly fortunate to have worked with him.

I wish to express my thanks to my co-supervisor, Prof. Marco Pranzo, whose collaborative approach has greatly enriched this research. Our extensive brainstorming sessions were both intellectually stimulating and highly productive, helping to shape the direction of this work. I have felt privileged to be part of a cooperative team, working together toward a common goal.

I would like to extend my sincere thanks to Prof. Roel Leus for hosting me during my visiting period at KU Leuven. This experience was extremely valuable both professionally and personally, offering me the opportunity to engage with different research perspectives and methodologies. Our collaboration was highly enriching, allowing me to observe both the commonalities and distinctions in our respective research approaches. This period broadened my understanding of research, and I am grateful for the knowledge and experience gained during this time.

I wish to thank Emmeline Perneel for her significant contribution to our work. Her innovative thinking, commitment, and focus on producing the highest-quality research have been truly impressive. Working with her has been an inspiring experience that has elevated the quality of our research considerably.

A special thanks goes to my colleagues from the University of Siena. Although our paths did not always cross in terms of direct professional collaboration, spending time with them has always been a pleasure. Throughout this journey, I have had the privilege of meeting remarkably intelligent, interesting, and inspiring individuals. Our exchanges and conversations provided not only intellectual stimulation but also invaluable support during each stage of this doctoral experience.

I would also like to express my gratitude to my colleagues at KU Leuven. Despite the relatively brief duration of my stay, they made me feel immediately welcome and at home. Our shared lunches and daily interactions, during which we enjoyed discovering our cultural differences, created a warm sense of community that I deeply appreciated. I was fortunate to meet such engaging people, always ready with an encouraging word, humor, or an interesting story. Their companionship greatly enriched my time in Leuven.

I would also like to acknowledge the valuable contributions of the professionals at Azienda Usl Toscana Sud Est and Azienda Ospedaliero Universitaria Senese, who collaborated with us by providing data and practical insights that enriched this research. Their input has been instrumental in grounding our academic work in real-world contexts and ensuring its practical applicability. Their perspective has significantly strengthened the relevance and impact of these studies.

Finally, I would like to thank my family and friends for being by my side from the very beginning until today, and for all that is yet to come. Their constant support and belief in me have been my foundation throughout this journey.

Chapter 1

Introduction

Decision-making permeates every aspect of human life, ranging from simple everyday choices to complex, high-stakes decisions that shape collective outcomes. It is not always possible to determine whether a decision is optimal or even appropriate; nevertheless, decisions must be made. In smaller, low-complexity environments, decisions based on intuition and limited quantitative analysis may suffice. However, in large-scale systems, effective decision-making demands a synthesis of quantitative methods and qualitative insights, including intuition, experience, and common sense [1]. It is precisely in such contexts that scientific disciplines like Decision Science provide powerful tools, enabling rational evaluation by systematically considering all relevant aspects of a situation. By developing abstract models of real-world processes, such as service operations, logistical systems, or market dynamics, Decision Science enhances our understanding of system behavior and identifies opportunities for performance improvement [2]. The uncertainty inherent in many decision contexts reinforces the need for structured and rigorous methodologies, a goal at the core of Operations Research.

The title of this thesis, "Combinatorial Models and Algorithms for Resource Allocation Problems in Healthcare Management and Logistics", reflects this perspective. Each chapter is unified by the overarching theme of resource allocation, explored across different application domains. The thesis addresses distinct yet interrelated problems in which combinatorial optimization techniques can enhance efficiency and support informed, data-driven decision-making.

Resource allocation problems concern the distribution of limited resources among competing activities, projects, or agents in a manner that optimizes one or more objectives, such as efficiency, cost reduction, timeliness, or overall utility, satisfying operational constraints, such as capacity limits, deadlines, and task precedence. Each possible allocation represents a choice among many alternatives, requiring the decision-maker to assess the implications of assigning resources to one task rather than another. In organizational settings, these challenges are intensified by concrete resource constraints, such as limited financial capacity, insufficient facilities, time shortages, or a lack of specialized expertise, which make it impossible to pursue all potential initiatives. Because dedicating resources to one project reduces what is available for others, ineffective choices can result in substantial losses in the form of unrealized benefits from more valuable alternatives. To navigate these constraints, organizations typically rely on formal planning and evaluation processes designed to compare proposed projects, assess their feasibility, and determine which initiatives should be prioritized, postponed, or rejected. These processes are often complex: proposals can be numerous, relevant information is distributed across

many individuals, and there may be competing interests. Given this complexity and the uncertainty surrounding many decisions, analytical tools and structured decision-making methods are essential for improving resource allocation and ensuring organizations derive maximum benefit from their scarce resources [3]. The study of resource allocation problems benefits from the formalism of combinatorial models, which provide a structured way to represent allocation options and constraints. By mathematically modeling tasks, resources, and objectives, these models enable decision-makers to systematically explore feasible solutions and identify those that best satisfy the desired criteria.

Scheduling plays a critical role within this framework, extending the classical allocation process to include the temporal dimension. Beyond determining which resources to assign to each task, it specifies when and in what order activities should be executed, accounting for precedence, availability, and duration constraints. Scheduling objectives are diverse and can include minimizing the total completion time, reducing late tasks, or balancing workloads and resource utilization. Resources and tasks vary widely depending on the application: resources may consist of production machines, airport runways, construction crews, or computing units, while tasks may represent manufacturing operations, take-offs and landings, construction phases, or program executions. Each task is typically defined by attributes such as priority, earliest start time, and due date [4]. By explicitly incorporating scheduling into resource allocation, time is treated as a scarce resource, enhancing the efficiency, feasibility, and overall effectiveness of the allocation plan. Achieving this requires a careful balance between efficiency and flexibility, short-term constraints and long-term goals, as well as quantitative optimization and qualitative judgment. This combination makes scheduling both a rich field for theoretical research and a practical tool for solving real-world problems.

Resource allocation problems represent a well-known and fundamental challenge across a wide range of domains, including industrial production, healthcare management, information processing, transportation, logistics, and service operations, illustrating their universality and importance. In healthcare, assigning nurses to shifts requires balancing coverage, legal requirements, and employee preferences. In computing and cloud infrastructure, CPU cycles or bandwidth must be allocated among competing tasks to maintain efficiency and performance. In project management, budgets and personnel are distributed among initiatives to maximize value while minimizing risk. In all these contexts, effective decision-making is essential to achieving optimal or near-optimal outcomes.

Among these application domains, healthcare represents a particularly critical context, and a central component of this thesis addresses specific management challenges within this sector. The scarcity of medical resources (including staff, operating rooms, diagnostic equipment, beds, and pharmaceuticals) combined with the high stakes involved, makes optimization both complex and essential. Poor scheduling can lead to underutilized capacity, prolonged waiting times, increased costs, and, most critically, adverse clinical outcomes. Healthcare systems operate under multiple constraints, including budget limitations, regulatory requirements, and capacity restrictions, while simultaneously responding to unpredictable demand and emergencies. Effective decision-making in this context ensures that available capacity is used wisely, prioritizes critical needs, and balances short-term operational requirements with long-term health outcomes. It also promotes

equity, ensuring that those with urgent needs receive timely care, and contributes to overall system resilience.

In this spirit, Chapters 2 and 3 respectively address two specific cases: the scheduling of operating rooms and the allocation of healthcare resources across a territory. Both problems are motivated by concrete operational needs and are analyzed with the dual objective of methodological rigor and practical relevance.

Despite differences across application domains, resource allocation challenges share common structural features. While arising in highly specific operational contexts, these problems can often be formalized within a broad class of scheduling models extensively studied in Operations Research. Among these, the Resource-Constrained Project Scheduling Problem (RCPSP) provides a fundamental framework for representing tasks, precedence relations, and limited resource capacities. Its generality and analytical tractability make it a natural starting point for understanding complex scheduling problems. For this reason, Chapter 4 examines the RCPSP as a foundational model for understanding and addressing complex scheduling problems in the field of project management. The study investigates the preemptive RCPSP, focusing on the balance between reducing project makespan and managing the complexity introduced by preemptions. Allowing an activity to be interrupted and resumed later can lead to shorter project durations, but at the same time it creates additional challenges for coordination, resource allocation, and overall project management. In addition to the theoretical analysis, a real-world case study is presented to illustrate the practical implications of the proposed approach.

Finally, Chapters 5 and 6 explore scheduling problems in environments where machines are subject to failures. Two distinct variants of this problem are examined, both analyzed from a theoretical standpoint. In these settings, interruptions are non-recoverable: when a machine fails, not only is the job in progress lost, but all subsequent jobs scheduled on the same machine are also affected. From a theoretical perspective, the chapters assess the computational complexity of these models, highlighting the difficulties in designing efficient scheduling strategies under such constraints. Together, these contributions provide perspectives on how machine failures impact scheduling decisions.

1.1 List of publications

Part of the work presented in this dissertation has already been published, while other parts are currently under review. The related publications are listed below.

- I. Salvadori and A. Agnetis, "The impact of the number of preemptions in resource-constrained project scheduling problems with time-varying resources: computational experiments and a case study," *Annals of Operations Research*, 2025.

- A. Agnetis and I. Salvadori, "Scheduling jobs on unreliable machines subject to linear risk," *Logistics*, vol. 9, no. 4, p. 157, 2025.

- A. Agnetis, R. Leus, E. Perneel, and I. Salvadori, "The Unreliable Job Selection and Sequencing Problem," arXiv: 2511.17105 [cs.DM].

- A. Agnetis, C. Bonavita, V. Mezzatesta, M. Pranzo, and I. Salvadori, "Un approccio collaborativo alla programmazione della chirurgia elettiva: il caso dell'Azienda Ospedaliero Universitaria Senese," accepted for publication in *Mecosan*.

Part I

Resource allocation models in healthcare management

Introduction to resource allocation models in healthcare

Optimization challenges in healthcare have been a subject of research for more than thirty years. However, their significance has grown over the years due to demographic trends, such as declining birth rates in most developed countries and the global rise in life expectancy. As populations grow older, the demand for healthcare services intensifies, while the workforce faces reductions due to retirements, burnout, and escalating job demands. This combination worsens the shortage of skilled professionals and specialized equipment, making the effective management of these limited and costly resources crucial. Operations Research offers powerful analytical tools to address complex challenges in healthcare, facilitating the efficient allocation and utilization of scarce resources, the effective management of costs, and ultimately enhancing both patient outcomes and satisfaction.

This introduction draws on a number of literature review articles focusing on Operations Research in healthcare ([5],[6]) and provides a general overview of the topic, establishing the context for the detailed analyses presented in Chapters 2 and 3.

Healthcare management spans several key domains. It involves planning and managing resources, including service planning, staff scheduling, operating room scheduling, and the coordination of non-emergency patient transport. Healthcare logistics covers the management of critical supplies such as blood products, organ transplantation, and the coordination of supply chains. Another important domain is disease diagnosis, prevention, and management, ranging from clinical assessment to preventive strategies and the control of infectious and chronic diseases. Complementing these dimensions are diagnostic imaging and public health interventions, both essential for early detection and for guiding effective health policies. This part of the thesis concentrates on operating room scheduling and allocation of services across territories, examining how optimization techniques can be employed to improve efficiency, accessibility, and outcomes in these critical healthcare sectors.

Operating room scheduling

Efficient planning and scheduling of operating rooms is crucial for hospital performance. However, operating room scheduling is inherently complex due to multiple interrelated factors. Firstly, decisions involve multiple stakeholders, including patients, surgeons, nurses and managers, often having competing priorities, making it challenging to align interests effectively. Additionally, patient classifications introduce further variability: they can be elective or non-elective, with elective cases further divided into inpatients and outpatients, each with distinct requirements and hospital stays. Emergency cases add unpredictability, requiring the minimization of deviations from preferred schedules while ensuring sufficient physician coverage to handle unpredictable patient arrivals.

Decisions in operating rooms planning occur at multiple organizational levels. At the strategic level, hospital managers focus on capacity planning, determining how much operating room time should be allocated to each surgeon or surgical specialty. At the tactical level, they allocate surgical blocks through the Master Surgical Schedule, coordinating surgical blocks and resources across multiple specialties while accounting for operating room availability, post-surgical resources, and overall hospital capacity. At the operational level, specific patients are scheduled, cases are sequenced, and ongoing surgical processes are monitored. These decisions operate on different time scales and must be closely aligned to maintain hospital efficiency.

To summarize, hospital decision-making in operating room planning is characterized by a wide range of often conflicting objectives that must be carefully balanced. These include minimizing operating room idle time and overtime, while at the same time maximizing the satisfaction of surgeons, staff, and patients, and maintaining a leveled bed occupancy across wards. Additional challenges stem from uncertainties such as variable surgery durations, patient no-shows, and other unpredictable events. Poor management of these complexities can result in longer waiting times, higher overtime costs, and increased procedure cancellations, highlighting the critical importance of effective operating room scheduling.

In this context, Chapter 2 presents a study introducing a collaborative approach to surgical planning at the operational level, aimed at balancing operating room efficiency with the clinical needs of patients.

Population health

The term population health refers to the overall health outcomes of a group of individuals and focuses on preventing disease, promoting healthy behaviors, and extending healthy life across the entire population, not just among patients who are already receiving care. Population health interventions target both healthy individuals and those in early, asymptomatic stages of disease, ranging from individual-level advice, such as smoking cessation, to subgroup-specific programs, such as childhood vaccinations and other prevention campaigns.

Analytical and quantitative methods have assumed an increasingly important role in supporting public health decision-making. Early applications focused on evaluating

screening policies for cancer and infectious diseases. The scope has since broadened to include strategic and policy-level decisions, such as forecasting healthcare service demand, selecting preventive programs to ensure long-term cost savings, and assessing the severity of disease groups through structured decision-making that integrates both policy-maker and patient perspectives. Such applications have enhanced the consistency, efficiency, and transparency of public health evaluations. Beyond disease prevention, quantitative modeling has also informed strategies to promote healthier lifestyles, including risk-based assessments of alcohol regulation, agent-based simulations of adolescent smoking behaviors, evaluations of targeted school interventions, and multi-objective optimization models for developing balanced national dietary plans. Together, these approaches underscore the importance of understanding how government policies shape individual behaviors, an area that benefits from interdisciplinary collaboration between Operations Research specialists, social scientists, policymakers, psychologists, and clinicians.

Extending these applications, analytical methods are also applied to the design and organization of healthcare delivery systems themselves. In healthcare location planning the objective is to determine the optimal placement and configuration of facilities to ensure equitable access to services and efficient resource utilization. This challenge is especially relevant in settings where demographic changes or in rural areas with sparse populations. Key considerations in location planning and dimensioning include determining the size and location of departments, minimizing patient travel, reducing costs, preventing service shortages, and limiting disruptions during reallocation. Identifying optimal sites for community healthcare facilities requires balancing top-down strategic planning with bottom-up local needs to maximize the number of sustainable facilities. Preventive healthcare networks, blood banking systems, and other specialized services also require strategic location decisions to balance supply and demand, ensure adequate coverage of donor areas, coordinate inventory distribution, and optimize operational efficiency. Overall, healthcare location planning is a multi-objective challenge that must integrate considerations of accessibility, population needs, service capacity, cost efficiency, equity, and sustainability across diverse and often unpredictable contexts.

Within this framework, Chapter 3 introduces a model for allocating resources to Community Houses, healthcare facilities newly established by the Ministerial Decree (DM) 77/2022, where citizens can access a range of health services.

Chapter 2

A collaborative approach to surgery case assignment

This chapter presents a decision support framework for elective surgery planning that aims to balance efficiency in the use of hospital resources with the need for flexibility in surgical team decision-making.

The chapter is organized as follows. Section 2.1 provides a brief review of related literature. Then, we illustrate the proposed surgical planning process (Section 2.2), followed by a discussion of the clustering procedure in Section 2.3.1. Section 2.4 introduces the mathematical model for case selection. Thereafter, we describe the validation of the model, carried out by means of a long-term simulation of our decision support tool. In this study, the real surgical data across a one-year horizon have been employed (Section 2.5). In this experimentation, different clustering strategies have been implemented and various performance indicators are observed, allowing a comparison with the current policy. Finally, implementation details are described (Section 2.6) and some conclusions are drawn (Section 2.7).

As widely recognized, effective scheduling of operating rooms (OR) is crucial to optimize surgical workflows, enhancing patient outcomes, and maximizing the utilization of healthcare resources, and the overall quality of patient care [7].

Despite the extensive (and ever-growing) literature on surgical planning problems, it has been observed by [8] that only a small fraction (7%) of all models has been implemented and is used in practice. As the authors note, this is often due to behavioral aspects of the stakeholders involved which are rarely reported in the research. One notable exception is [9], which interestingly elaborate on barriers to the implementation of a surgery sequencing problem. Indeed, many models typically overlook the interplay between hospital managers and surgical teams, as well as the process through which surgical plans are actually developed in practice.

In OR planning and scheduling processes, a relevant issue is how *decision-making authority* is distributed among stakeholders (clinicians, managers, patients). In *centralized* scheduling, decisions are consolidated by a dedicated central office, and the Master Surgical Schedule (MSS) is populated by a centralized planner. In this case, surgical cases are selected on the basis of some parameter summarizing clinical priority (*score*) as well as of resource efficiency criteria. Hence, the choice of surgical cases to be operated during the next period (e.g., one week) remains with the management, who is typically concerned with efficiency and cost. Centralization can lead to greater resource utilization; however, if not managed carefully, it can negatively impact staff and surgeon morale.

Conversely, *decentralized* models distribute decision-making authority to lower levels within the organization leaving to surgeons or their immediate teams direct control over their schedules at the risk of inefficiencies and inconsistencies across different units. This typically results in surgeons establishing a routine for their practice and better plan for future cases, thereby improving clinical care [10]. In this case, surgeons make patient selection decisions based on clinical criteria, but studies report that surgeons' criteria may vary considerably among them, and typically with little consideration of overall system utilization issues (see Johnston et al. [11]). Indeed, as noted by Johnston et al., the hospital administrators' desire to maximize resource utilization may "conflict with the priorities of surgeons to maintain flexibility [...] all while appropriately addressing a patient's ever shifting medical needs."

Finally, *hybrid* models try to balance the efficiency of centralized decision making with the flexibility of decentralization. Hybrid models can be achieved by centralizing OR planning while allowing for more decentralized operational execution and adjustments. The surgical plan may be specified at the level of surgery types (e.g., ICD code) in order to retain flexibility in patient selection. This approach allows for a balance within the clinicians' and the managements needs. As stated by [12], the complexity and unpredictability in healthcare processes may require high levels of clinician autonomy. The key challenge in hybrid approaches lies in ensuring that autonomy is granted where it is genuinely necessary. Anyhow, both the managers and the surgeons have to be involved in the decision process if one wants to obtain solutions balancing the various stakeholders' objectives.

Our project has been carried out in close coordination with the management of the University Hospital of Siena (Azienda Ospedaliero Universitaria Senese, AOUS), Italy, the main hospital of Southern Tuscany. In the managers' experience, surgeons consider a certain degree of autonomy as highly valuable for their professional success, and are not prone to accept a surgical plan imposed by hospital managers. On the other hand, an efficient use of resources (operating rooms and wards) may require selecting appropriate subsets of surgeries. As a consequence, the final detailed surgical plan is usually the result of a negotiation between managers and surgical teams, and it often represents a compromise among different goals. Our study proposes an approach whose aim is to facilitate such a negotiation process, pursuing a satisfactory solution for both stakeholders and ultimately, for the patients.

We focus on the operational phase [13] of the process. Given the MSS, i.e., a cyclic (e.g. weekly) assignment of operating room time blocks to surgical teams [14], the problem is to allocate elective patients in the waiting list(s) to each MSS block. In the context of the Surgical Case Assignment Problem (SCAP), we are concerned with *advance scheduling*, i.e., the assignment of patients from the list to a certain operating room and time block (surgical session) [15].

We aim to design a decision process ensuring a certain degree of efficiency while maintaining practical feasibility, assuming that the MSS is predetermined by the hospital organization (*block scheduling*). The key idea of our modeling approach is to partition all surgeries into a limited number of surgical groups called *clusters*, each cluster being relatively homogeneous in terms of resource consumption, i.e., surgical procedures in the same cluster will have similar duration and similar post-surgical length of stay. Hence,

rather than specifying the individual patients that must be operated, the managers require the surgeons to perform a certain number of procedures from each cluster in each session of the next planning period. With this collaborative approach to surgical planning, *advance scheduling* is solved by a mathematical model, thus ensuring efficiency goals, while the surgeons are required to solve the allocation scheduling and thus they still have considerable flexibility in selecting individual patients from each cluster.

The main contributions of this study are the following:

- We propose a mathematical model working on clusters of surgeries that allows an efficient use of the available resources while granting surgeons' autonomy.
- The model is extensively tested on real world data from the University Hospital of Siena, showing the viability of the proposed approach. The results are compared with both historical data and the results that could have been obtained through a purely centralized approach (without collaborative decision making), hence disregarding surgeons' autonomy.
- We developed an application to enable the use of our model directly within the hospital. It is a prototype that employs an integer linear programming algorithm to assign patients to surgical sessions while respecting constraints such as bed capacity, session duration, and clinical priorities. The system architecture consists of two main components: the optimization algorithm implemented using Gurobi, and a web interface that serves as a bridge to the optimizer, allowing users to interact with the model in an intuitive and user-friendly way.

2.1 Literature Review

The literature on surgical scheduling is extensive and growing. We limit ourselves to some recent surveys offering a systematic review of the main contributions in this field and then focus on a few papers specifically relevant for our study.

A recent review [6] summarizes the evolving role of operations research in addressing complex challenges within various healthcare domains. The authors observe that, despite the great volume of research, the actual impact has been limited and that the involvement of different stakeholder perspectives is essential for achieving greater societal impact.

A survey of the last twenty years of research on operating room scheduling [7] identifies key factors, issues, and algorithms for both certain and uncertain conditions. [16] review OR scheduling models integrating multiple hierarchical decision levels, noting growing interest in their potential despite their complexity. [17] focus their review on the differences between *inpatients* (overnight stay) and *outpatients* (same-day) surgeries. They observe that outpatient procedures generally perform better due to their lower complexity and variability. [18] review AI tools for real-world patient scheduling noting how only a small number of applications to the operating room scheduling. While in [19] the focus is on machine learning applications for operating room management.

An extensive review [8] classifies OR planning and scheduling research based on various characteristics and highlights three common pitfalls that hinder the adoption of research

findings in practice: *(i)* the lack of a clear target audience (researchers vs. practitioners), *(ii)* the use of inappropriate performance measures, and *(iii)* insufficient reporting on the hospital setting and model assumptions.

As highlighted by several of these authors, the human factor is a decisive aspect for real-world applications of advanced decision making techniques. Optimizing surgery schedules requires models that can somehow account for, or at least provide flexibility for, human and team-based elements (Pasquer et al. [20]), rather than focusing solely on resource allocation or patient prioritization in isolation. Pasquer et al. argue that to be effective, a surgical planning decision support system should include both *hard* constraints like the availability of a specific instrument or an open OR slot and *softer*, less quantifiable factors such as team compatibility, individual experience and preferences. Including hard constraints is usually easy from a mathematical standpoint since they are easily quantifiable, much less so is the inclusion of softer constraints, as they not always easy to quantify.

In the OR setting, surgeons are the most relevant stakeholders and usually prefer to schedule their own surgeries rather than delegating it to a centralized scheduling systems [11]. This resistance to yielding control is due also to the need to address patient's ever shifting medical needs and the absence of a timely data-driven feedback on the impact of their scheduling decisions. The authors emphasize that successful implementation of standardized and centralized scheduling routines in surgical environments requires a deep understanding of surgeons' concerns. Although the study is limited to a single hospital, the conclusions appear to have general validity.

Another relevant issue is the choice of performance measures. [21] suggest that the goal of OR management should be to minimize the inefficiency of use of OR time, which can be measured by a weighted sum of under-utilized and over-utilized OR times. They highlight the presence of cognitive biases that significantly affect decision-making of OR managers and staff, often leading to suboptimal OR allocation, unless mitigated by data-driven approaches and informed leadership. They strongly advocate a data-driven approach, based on 7-12 months of OR data, while they observe a limited impact of the specific scheduling heuristics adopted.

Indeed, various studies suggest that surgical teams highly value decisional autonomy, and organizations should support it also to enhance the surgeons' identification with the organization [22].

In the literature, the number of models that explicitly incorporate surgical team's wishes in the establishment of an operating room schedule are relatively few, as observed by [23]. These authors developed a constraint programming-based approach including various real-life issues, such as the maximization of affinities among staff members and various surgeons' and anesthesiologists' preferences (e.g., operating rooms or people to work with). The model is conceived in a modular way, so to be adapted to a specific hospital's needs.

[24] propose a scheduling model in which surgeons express preferences on the use of various time slots in specific rooms, and the problem is treated as a two-sided (i.e., bicriteria) matching problem. However, in this model surgeries are already selected and

only their time allocation is determined accounting for the surgeons' preferences.

Also [25] consider a surgical planning problem in which surgeons' have preferred operating rooms and times. The authors propose a model that computes a surgical schedule that minimizes both the number of ORs used and the overtime, while keeping the surgeons' preferences into account.

[26] address the SCAP while considering the conflicting priorities and perspectives of the management and the surgeons. In fact, hospital administration prioritizes scheduling surgeries based on their due dates and priority levels, while the authors observe that a Last-In, First-Out strategy reflects the surgeons' current practice. The authors propose and evaluate deterministic and robust optimization models to create surgical schedules for a one-week planning horizon.

The idea to group surgeries in homogeneous clusters has already been proposed in the literature. For example, [27] consider a cardio-thoracic department of a Dutch hospital and propose a mathematical model to determine the number of patients of a given category operated in a given day. Categories are defined according to the following axes: child/adult, short/long duration, short/long LOS. Results aim to measure the impacts of different scenarios by changing the number of available resources.

More recently, [28] propose a mathematical model with the aim of producing an efficient, balanced or robust plan. Clusters are obtained by broadly grouping surgeries according to their duration (less than an hour, between one and two hours and more than two hours) and the expected post-surgical length of stay (expressed in days). Tests are carried out based on real world data from an Italian hospital, and schedules are tested by simulation to take into account the difference between expected and actual duration. From the results, it appears that no objective function performs better according to all the objectives.

This work distinguishes itself by providing a comprehensive assessment of the practical impact of surgery groups, with explicit consideration of surgeons' flexibility in patient selection, an aspect often overlooked in the literature. Different patient selection policies and clustering strategies are proposed and compared, with results benchmarked against the actual plan implemented at the hospital.

2.2 The decision process

Hereafter we describe the planning process we envision for elective surgery planning. According to this process, surgical plans are the result of the collaboration between hospital management and surgical teams.

For each operating theater, the hospital adopts a MSS, based on long-term average demand for various surgical specialties. Each specialty has therefore a certain set of weekly surgical sessions, which have to be populated by surgeries of the corresponding specialty. Selection of surgical cases and their allocation to these sessions takes place every Friday afternoon two weeks in advance (i.e., on Friday of week t the surgical plan for week $t + 2$ is decided). This advance scheduling is necessary to allow sufficient time for pre-hospitalization procedures, including preoperative assessments, diagnostic tests, and patient preparation. These steps are crucial to ensure that all required medical

evaluations are completed before surgery, minimizing the risk of last-minute cancellations and optimizing operating room efficiency.

The decision process involves different stakeholders, having in general (partially) different goals. Surgeons are typically focused on maximizing the chances of successful operations, taking into account an up-to-date view of individual patients' clinical conditions. While surgeons always care about the individuals' health outcome and hence the timeliness of the intervention, the line of reasoning through which they build their preferences over a set of patients is not easy to formalize or to make explicit (see e.g. [11]). In most situations, surgeons highly value control on selecting the set of surgeries to be performed ("having the final say"). On the other hand, managers are also concerned with system-wide KPIs such as overall throughput and resource utilization. Hence, the surgical plan which is eventually established represents a compromise solution and a trade-off among several goals. As a consequence, the current practice is to reach a consensus over a set of patients through informal discussions, which may result lengthy and complex.

We focus specifically on this selection process, which is the result of a negotiation among managers and surgical teams. Hence, if one wants to design a decision support tool to be implemented in a real-life context, the tool should be flexible enough to accommodate surgeons' preferences within a planning framework designed by hospital managers. Here we propose a decision tool which is based on a formally established negotiation process, working as follows:

1. All surgical procedures are categorized into a number of *clusters*, so that procedures in the same cluster are relatively homogeneous for what concerns resource absorption, namely operating room time and post-surgical length of stay (LOS). The cluster generation process takes place once in a while, i.e., clusters are reviewed only if significant changes in the repertoire of offered surgeries arise.

2. As soon as a new patient is added to the waiting list, he/she is assigned to one cluster, according to a criterion of proximity. In this phase, specific clinical conditions of individuals can be taken into account.

3. Every week, given the current MSS, the hospital management devises next week's surgical case assignment specifying how many cases *from each cluster* should be performed during each session of the MSS.

4. Once the SCAP at the cluster level is decided, surgeons freely decide which patients should be selected from each cluster.

Notice that the clustering approach is based on resource absorption, not on surgery disciplines. This implies that operations from different surgery types may be assigned to the same cluster, and viceversa it may happen that on the basis of individual clinical conditions, different patients undergoing the same surgery type may have different values of expected surgery duration and LOS (as reported in the waiting list). The idea is that managerial goals will be secured by the way the surgical plan is specified at the cluster level, whatever the choice made by the surgeons for case selection.

This basic idea requires a number of steps, which can be outlined as follows.

- **Clustering**. A key point of the above process is to devise the "right" number of clusters which best balances different objectives. A large number of clusters may allow a more accurate allocation of OR capacity, but may overly limit surgeons' options. Moreover, from a practical viewpoint, having too many clusters can make them less easily recognizable by the surgeons, and hence may threaten the validity of the process itself. In fact, if a surgeon is told to select one patient from a cluster X (expected duration 55 minutes and 2 days of stay) and one from cluster Y (expected duration 70 minutes and 1.5 days of stay), the surgeon may regard this distinction as artificial and not really impactful. On the contrary, surgeons would prefer fewer, easily-recognizable clusters, but this would come at the expense of homogeneity within each cluster, and hence might result in less predictable outcomes. When deciding the number of clusters in a dataset, both uncertainty and robustness play a crucial role. Uncertainty reflects the confidence in the clustering structure: when data are noisy or cluster boundaries are poorly defined, selecting the appropriate number of clusters becomes challenging. Robustness, on the other hand, measures the stability of the chosen number of clusters under variations in the data or algorithm parameters. A robust solution consistently identifies the same number of clusters despite such variations, indicating a reliable and trustworthy clustering structure. Indeed, for the approach to be successful, it is mandatory that the surgical teams agree about the "right" level of aggregation entailed by a certain number of clusters.

- **Cluster assignment**. Once the set of clusters is defined, each surgical case in each waiting list is assigned to a cluster. This depends on the individual features of each surgical case, such as the clinical condition of each patient, although of course many patients requiring surgeries having the same ICD code are likely to be assigned to the same cluster. Here we simply consider that each procedure has an expected duration and an expected post-surgical LOS.

- **Generation of the surgical plan at the cluster level**. Thereafter, on the basis of the MSS, a mathematical optimization model is run selecting clusters, i.e., deciding how many cases from each cluster should be selected for each surgical session of the MSS. The model accounts for the capacity of both operating rooms and post-surgical wards. The objective of the model concerns the maximization of operating room utilization, as well as the score of the surgeries currently populating each cluster. The model explicitly focuses on managerial objectives, which are quantifiable and can be easily integrated into a mathematical model.

- **Patient selection**. Complying with the solution proposed by the model at the cluster level, the surgeons select patients from each cluster. In this way, individual patients' clinical aspects are taken into account in the final phase of the process.

2.3 Algorithmic approach

2.3.1 Clustering and cluster assignment

Two clustering analyses were performed according to the surgeries' regime type: one for inpatient procedures and another for day-surgery procedures, both using the *k-means* algorithm [29]. This unsupervised method partitions a set of points of R^n into a predefined number of clusters by iteratively refining centroid positions. The algorithm initializes centroids randomly, assigns each point to the nearest centroid based on Euclidean distance, and updates each centroid as the mean of the assigned points. The process repeats until centroids stabilize. This algorithm aims to minimize intra-cluster variance, defined as the sum of squared distances between data points and their respective centroids. This ensures that points within a cluster are as similar as possible while maintaining clear separations between different clusters.

For each patient ℓ two dimensions were taken into account: the duration of the surgical procedure (*surgical time* $ST(\ell)$) and the subsequent *length of stay* in the ward $LOS(\ell)$. In the case of day surgery procedures, the clustering is carried out in a single dimension, as we consider $LOS(\ell)$ always equal to 1. Therefore, the centroid of cluster k is represented by the pair $(ST(k), LOS(k))$.

Since the data source used for clustering contains fractional values, the LOS may assume non-integer values during the clustering phase. Nevertheless, for the purposes of the mathematical model, the LOS assigned to each cluster is rounded to the nearest integer. Note that not all the specialties may have cases belonging to all the clusters. To each cluster we also associate a score which is used to guide the optimization process since it reflects the importance of the surgical cases for each specialty present in each cluster.

2.4 The optimization model

In this section we formally introduce and illustrate the mathematical model that allocates clusters to the surgical sessions of a certain planning horizon.

The input of our problem is a certain set of surgical specialties, each with an associated waiting list divided into clusters, and a MSS specifying the surgical specialty assigned to each surgical session across the planning horizon. Also, a set of post-surgical wards is given, each associated with a set of surgical specialties. Given the set of clusters and the current state of the waiting lists, the problem is to decide the case assignment at the cluster level so that an appropriate objective is optimized, respecting the capacity of operating rooms and post-surgical wards. The above aspects are now described in detail.

Surgical specialties and Sessions. In the hospital, there is a set I of *surgical disciplines* or *specialties*. Let $i \in I$ represent a single specialty. A session s is a specific time block of a specific OR. A Master Surgical Schedule (MSS) is given, specifying the specialties' allocation of the sessions over time. The MSS consists of a set of *sessions* S. Each session $s \in S$ is characterized by a certain *duration* $d(s)$, the *day* $g(s)$ in which the session takes place, a surgical specialty $i(s)$ and a *regime* $r(s)$, which is either *day surgery* $(r(s) = 1)$ or *ordinary* $(r(s) = 0)$. Day surgery patients are discharged on the same day

and can be assigned to both day surgery or ordinary sessions. Ordinary patients can only be assigned to ordinary sessions. Each session has to be populated by cases of the corresponding specialty. We let $S(i) \subset S$ denote the set of sessions devoted to specialty i. Within each session, we assume a sequence-independent setup time d_{setup} between two surgeries.

Wards. A set W of post-surgical *wards* exists. Let $w \in W$ represent a single ward and let $W(i) \subset W$ be the subset of wards accommodating patients of specialty i. After surgery, the patients of surgical discipline $i \in I$ have to be dismissed to a ward $w \in W(i)$. Multiple specialties may be dismissed to the same ward. We let $S(w)$ denote the set of sessions whose patients are dismissed to ward w. Each ward $w \in W$ has a number of (identical) beds $b(w)$.

Clusters. We let K denote the set of all clusters. Each cluster $k \in K$ is associated with a nominal *surgical time* $ST(k)$, *length of stay* $LOS(k)$ and an associated *cluster regime* $r(k)$ (with $r(k) = 1$ for day surgeries and 0 for ordinary surgeries). Note that, day surgery clusters can be assigned to any session, whereas ordinary clusters can be assigned only to ordinary sessions.

Waiting lists. Let $L(i)$ be the *waiting list* containing all the patients of specialty i waiting at time t for their surgical procedure. Regardless of their cluster k, each patient $\ell \in L(i)$ is associated with a surgical procedure characterized by an *expected surgical time* $ST(\ell)$ and an *expected length of stay* $LOS(\ell)$ in the post-surgical ward.

Score of a patient. At any planning time t, each patient ℓ in a waiting list is assigned a *priority score* $\sigma(\ell)$ based on their urgency category. The score is computed considering the *classes* defined by the Italian Ministry of Health and the days spent by the patient in the waiting list. These are classified as follows [30]. For each class c, a *deadline* D_c is defined, indicating the time (days) from insertion in the waiting list within which the surgery should be performed:

- Class A. These cases may quickly worsen and/or delays may significantly impact the prognosis. They require being performed within $D_A = 30$ days from insertion in the waiting list.
- Class B. These cases may entail the danger of severe pain or disability but no immediate risk, hence they require admission within $D_B = 60$ days.
- Class C. This class includes cases with minor symptoms, aiming for admission within $D_C = 180$ days.
- Class D. It covers cases without pain or disability, has a maximum wait time of $D_D = 360$ days.

Note that not all classes may be present within each surgical discipline. At any given time t, the score is given by:

$$\sigma(\ell) = \frac{f_t}{D_\ell}, \tag{2.1}$$

where f_t denotes the flow time (i.e., days elapsed from the insertion in the waiting list), D_ℓ is the deadline of the class the surgery belongs to.

Pool of candidate surgeries. While in principle one might consider as eligible all patients in a list, we restrict a-priori the set of candidates in order to limit the complexity of the model and avoid generating many useless variables. Assuming that the patients in each waiting list $L(i)$ are ordered by decreasing score $\sigma(\ell)$, let the *pool of candidate surgeries* $P_h(i)$ denote the set of the first h cases in list $L(i)$, and let $ST_h(i)$ denote the sum of their surgical times. Let q denote the smallest number such that $ST_q(i)$ exceeds by a certain proportion β (e.g., $\beta = 0.5$) the total time available to specialty i in the MSS:

$$ST_q(i) \geq (1+\beta) \sum_{s \in S(i)} d(s).$$

In our model we only consider the set $P_q(i)$ as candidates, and we set $n(i)$ equal to the number of patients in $P_q(i)$, i.e., $n(i) = |P_q(i)|$. We let $n(i,k)$ denote the number of surgeries in $P_q(i)$ belonging to cluster k. Since the optimization model does not select individual patients, restricting the pool of cases to $P_q(i)$ implicitly ensures that consideration is given to highest-priority patients. A smaller value of β restricts the surgeons' choice to a smaller number of candidates, while a larger β allows more flexibility.

Score of a cluster. For each session of each specialty, the model determines how many cases should be selected from each cluster during the next planning horizon. An upper limit for this value is $n(i,k)$. Let $\sigma(i,k)$ be a score associated to cluster k of each specialty i. Denoting by $P_q(i,k)$ the surgeries of $P_q(i)$ belonging to cluster k, we define $\sigma(i,k)$ as the average score of the patients in $P_q(i,k)$:

$$\sigma(i,k) = \frac{\sum_{\ell \in P_q(i,k)} \sigma(\ell)}{n(i,k)}.$$

Table 2.1 summarizes the notation introduced for data.

The decision model employs a number of variables. The main decision variables are $x_{k,s}$, representing how many surgeries from cluster $k \in K$ are assigned to each session $s \in S$. Moreover, we denote by $x_{k,t,w}$ the patients from cluster k scheduled on day t and entering ward w, and by $x_{t,w}$ the total number of patients operated on day t and entering ward w. Another set of variables is used to describe the dynamics of the wards. Variables $z_{t,w}$ denote the number of patients currently in ward w on day t, and $y_{t,w}$ the number of patients that are dismissed from w on day t (and hence do not occupy a bed on day t). Table 2.2 summarizes the notation introduced for variables.

We can now present the complete integer linear programming formulation of our cluster selection model.

$$\max \quad \alpha \cdot \sum_{k \in K} \sum_{s \in S} ST(k) x_{k,s} + (1-\alpha) \cdot \sum_{k \in K} \sum_{s \in S} \sigma(i(s),k) x_{k,s} \tag{2.2}$$

$$x_{k,s} = 0 \quad \forall k, s : (r(k), r(s)) = (0,1) \tag{2.3}$$

$$\sum_{s \in S(i)} x_{k,s} \leq n(i,k) \quad \forall i, k \tag{2.4}$$

I, i	set of surgical specialties and single specialty index
S, s	set of all sessions in the MSS and single session index
W, w	set of wards and single ward index
K, k	set of clusters and single cluster index
T, t	time horizon and time index
$S(w)$	set of sessions dismissing patients to ward w
$S(i)$	set of sessions of specialty i
$d(s)$	duration of session s
$g(s)$	weekday of session s
$r(s)$	regime of session s (1: day surgery, 0: ordinary)
$i(s)$	specialty of session s
$ST(k)$	nominal surgical time (hours) for a procedure in cluster k
$LOS(k)$	nominal length of stay (days) of cluster k
$r(k)$	regime of cluster k (1: day surgery, 0: ordinary)
d_{setup}	(sequence-independent) setup time before each procedure
$b(w)$	number of beds in ward w
$z_{0,w}$	number of beds occupied at the beginning of the planning horizon
$n(i,k)$	number of procedures from specialty i, cluster k in the pool of candidate surgeries

Table 2.1: Notation for problem data.

$x_{k,s}$	number of surgeries from cluster k assigned to session s
$x_{k,t,w}$	number of surgeries from cluster k scheduled on day t in a session of $S(w)$
$x_{t,w}$	number of patients scheduled entering ward w on day t
$z_{t,w}$	bed occupancy of ward w on day t
$y_{t,w}$	number of patients of ward w discharged on day t

Table 2.2: Notation for problem variables.

$$\sum_k (ST(k) + d_{setup}) x_{k,s} - d_{setup} \leq d(s) \quad \forall s \tag{2.5}$$

$$x_{k,t,w} = \sum_{s:g(s)=t, s\in S(w)} x_{k,s} \quad \forall k,t,w \tag{2.6}$$

$$x_{t,w} = \sum_{k\in K} x_{k,t,w} \quad \forall t,w \tag{2.7}$$

$$y_{k,t,w} = x_{k,t-LOS(k),w} \quad \forall k,t,w \tag{2.8}$$

$$y_{t,w} = \sum_{k\in K} x_{k,t-LOS(k),w} \quad \forall t,w \tag{2.9}$$

$$z_{t,w} = z_{t-1,w} + x_{t,w} - y_{t,w} \quad \forall t,w \tag{2.10}$$

$$z_{t,w} \leq b(w) \quad \forall t,w \tag{2.11}$$

$$z_{0,w} = z_{T,w}^{\text{prev}} \quad \forall w \tag{2.12}$$

$$x_{k,s} \in \mathbf{Z}^+ \quad \forall k,s \tag{2.13}$$

$$z_{t,w} \in \mathbf{Z}^+ \quad \forall t, w \tag{2.14}$$

In (2.2), we consider two different objectives, namely maximizing operating room utilization and an index of procedure priority. The former objective corresponds to maximizing the total nominal length of scheduled clusters, the latter to the sum of the scores of scheduled clusters. Constraints (2.3) forbid the assignment of day-surgery operations to inappropriate sessions. (2.4) enforce the limit on the total number of surgeries from each cluster to be scheduled throughout the MSS. (2.5) establish that the surgeries scheduled in a session cannot exceed the duration of the session. We assume that the first surgery can start at the beginning of the session, while a lag of d_{setup} minutes exists between two consecutive surgeries. Constraints (2.6) and (2.7) define variables $x_{k,t,w}$ and $x_{t,w}$. Notice that the constraints (2.6) establish a link between sessions (and hence, specialties) and wards. As a consequence, even if surgical sessions are pre-allocated to the various surgical disciplines in the MSS, the problem cannot be decomposed by specialties, because patients from different specialties may share the same post-surgical ward. Constraints (2.8) establish the connection between operating rooms and wards. For each ward and cluster, the number of patients dismissed from the ward on day t equals the number of patients who were operated (and hence, who entered the ward) $t - LOS(k)$ days before. Constraints (2.9) define the total number of patients discharged from ward w on day t. Constraints (2.10) are the classical inventory balancing equations, stating that bed occupancy in each ward w on day t is obtained from the occupancy of day $t-1$ by adding new patients and removing discharged patients. Ward capacity is enforced by constraints (2.11). Initial conditions for each ward are defined in (2.12), where $z^{\text{prev}}_{T,w}$ represents the bed occupancy at the end of the previous week. The integrality constraints on the number of surgeries are (2.13) (note that we do not need enforce similar constraints on $y_{k,t,w}$ or $z_{t,w}$).

We note that the model allows accommodating some specific policies, such as forcing the execution of a certain set U of surgeries across the model time horizon. To this aim, for each surgery $u \in U$ one can define a cluster K_u only containing surgery u and, letting $S(u)$ denote the sessions in which u can be performed, for each $u \in U$ one can add the constraint:

$$\sum_{s \in S(u)} x_{K_u,s} = 1. \tag{2.15}$$

2.5 Model validation

In this section we illustrate the approach we employed to validate the proposed process and model. To evaluate the long term effects of the mathematical model introduced in the previous sections, we perform a set of simulations of the proposed surgical planning process throughout a period of one year (in the following referred to as *simulation period*), and compare the results of these simulations with the real-life surgical process during the same period of time. The scenarios corresponding to each simulation are explained below.

At each iteration of each simulation, on the basis of the current status of the system, an instance of the mathematical optimization model is solved for the *planning horizon* of the model, which has been chosen as two weeks to provide a broader range of sessions for

accommodating patients, especially oncological ones. Then, the simulation iterates to the next planning horizon by updating all the data accordingly. The validation proceeds as follows.

Simulation input. The input to the simulation experiment consists of all relevant information on the surgical process collected throughout the one-year simulation period. This includes:

- *Operating theater*. Number of ORs for each operating block.
- *Master Surgical Schedule*. Specifies the surgical sessions taking place across one week, i.e., the specialty assigned to each OR in the various weekdays. We assume that the MSS does not change during the simulation period.
- *Waiting lists*. The snapshot of the waiting list $L(i)$ for each specialty i at the beginning of the simulation period is known. This includes, for each patient ℓ, the procedure, the initial flow time, and the corresponding score $\sigma(\ell)$.
- *Patients' data*. For each individual patient who has been operated throughout the simulation period, the *realized*, a-posteriori surgical time and length of stay (these data are known from hospital records). Note that these data are different from the a-priori estimates used in the mathematical models.

Clustering. To generate the clusters, the k-means clustering algorithm has been applied to the set of all surgeries performed in the hospital throughout a certain time span to identify a given number of different clusters. The clustering does not change across the simulation period. During the experiment, upon entering its respective waiting list, each patient is assigned a specific cluster. An exception is made for oncological patients, who require timely access to surgery. For each of these patients, a dedicated cluster (only containing that patient) is created and constraints (2.15) are used to force the selection of the surgery.

Generation of surgical plans at the cluster level. The simulation proceeds by two-week planning steps. Namely, given the data at a certain day t, the mathematical model described in Section 2.4 is solved, generating a two-week plan ranging from day t to $t + 14$. Recall that the optimization model specifies the plan at the cluster level, with no specification of individual surgeries.

Surgeons' decision. For each two-week planning period, the optimization model returns a number of surgeries from each cluster to be performed in each session. In our study, we need to represent *how* the surgeons decide to populate the surgical lists. While in reality this may depend on issues related to each patient's clinical state, in our simulation we had to resort to some general behavioral criterion. We considered that surgeons select the patients from the respective waiting lists *in non-increasing order of their score* $\sigma(\ell)$ at the planning time.

Data Update. Once a complete surgical plan is available for the current two-week period $[t, t + 14]$, the waiting lists are updated:

- all the surgeries planned in the period $[t, t + 14]$ are removed from the respective waiting lists;

- the scores of the surgeries which are still in the waiting lists are updated;
- newly arrived surgeries are added to the waiting lists;
- each new surgery is assigned to the nearest cluster on the basis of the expected surgical time and length of stay, and its score is computed;
- thereafter, the pool of candidate surgeries and cluster scores are updated on the basis of the new waiting lists.

Statistics collection. At the end of each simulation, statistics are collected concerning the following indices:

- *Throughput*. Total number of surgeries performed throughout the year.
- *Flow time*. Average patient waiting time (days) from decision time to the time the patient is operated. This figure is computed for all patients which are operated during the simulation year and it does not include the patients still in the waiting list at the end of the simulation year.
- *Room utilization*. We consider various figures related to the utilization of operating room resources. One figure is the average *overtime* over a two-week period, i.e., the total amount of time by which session duration was exceeded. Similarly, *undertime* indicates session time which has not been used. Another figure is the average *room occupation*, expressed as the total surgical time assigned to sessions (in hours).
- *Bed overflow*. Average number of bed-days exceeding the ward capacity during a two-week period. Again, to compute this figure the actual value of the post-surgical length of stay of each individual patient has been used.

These indices are used in comparing the performance of the proposed process with the current practice.

2.6 Computational experiments/case study

In this section we present the experiments we have performed to validate the process in a specific hospital, namely the Siena University Hospital (AOUS). It is one of the largest hospitals in Tuscany, with over 600 beds (covering both post-surgical and non-surgical cases) distributed across its 57 Complex Operational Units (UOC). Each year, it performs more than 17,000 surgical procedures (including emergencies) requiring hospitalization and 5,000 outpatient surgeries.

All surgical cases we considered are divided into 24 surgical specialties, and there are 7 post-surgical wards. Table 2.3 lists specialties along with the corresponding post-surgical ward. General surgery, Head/neck and Gynecology wards also include a Day Surgery section dedicated to same-day surgical procedures.

Surgical specialties	Wards
Anesthesia and pain therapy	General surgery
Bariatric surgery	General surgery
General surgery	General surgery
Hepatobiliopancreatic surgery	General surgery
Plastic surgery	General surgery
Urology	General surgery
Endoscopic urology	General surgery
Orthopedics	General surgery
Trauma-oriented orthopedics	General surgery
Vascular surgery	General surgery
Maxillofacial surgery	Head/neck
Thyroid surgery	Head/neck
Ophthalmology	Head/neck
Otorhinolaryngology	Head/neck
Neurosurgery	Neurology
Heart and great vessels surgery	Cardiology
Surgical treatment of heart failure	Cardiology
Thoracic surgery	Thoracic surgery
Oncologic breast surgery	Gynecology
Gynecology	Gynecology
Pediatric surgery	Pediatric surgery
Pediatric ophthalmology	Pediatric surgery
Pediatric otorhinolaryngology	Pediatric surgery
Diagnostic pediatrics	Pediatric surgery

Table 2.3: Mapping of surgical specialties to wards.

2.6.1 Design of experiments

The validation approach described in Section 2.5 has been applied in a number of distinct scenarios. For all scenarios, the simulation concerned the time span from 4 July 2022 to 2 July 2023. This includes 26 two-week periods. The common elements to each validation experiment are:

- for each surgical specialty, the snapshot of the waiting list at 1 July 2022;
- the real MSS adopted by the hospital during each of the 52 weeks;
- the current definition of surgical clusters;
- for each surgical specialty, the arrival dates of each new surgery in the waiting list throughout the whole simulation horizon.

In the experiments we assumed that the wards are empty at the beginning of the simulation period. The effect of this assumption in the long run is negligible, considering the first two-week period as a warm-up period.

2.6.2 Clustering

The clustering phase has been applied to the set of all elective surgeries performed within the Siena University Hospital in the time span from 1 January 2018 to 30 June 2023, all of which required patient hospitalization (i.e., excluding outpatients but including day surgeries). Outliers were identified and removed to enhance the robustness of the analysis.

A central point in our approach is the determination of the number of clusters. To start with, we applied the k-means algorithm to the database of all elective surgical procedures, with two exceptions, namely day surgeries and oncological surgeries. All day surgeries have LOS=1, and as they typically do not require long surgical times, we only considered two clusters, called 0D and 1D in the following. Concerning oncological surgeries, we have mimicked the actual policy adopted by the hospital managers. Such a policy consists in pre-assigning oncological surgeries to available sessions of the same specialty within the two-week planning horizon.

The k-means clustering algorithm has been applied to the set of non-oncological elective ordinary surgeries, considering all values of K from 2 to 40. For each case, the average distance from the centroid (a measure of how homogeneous are the clusters) has been recorded. As already mentioned in Section 2.2, while the highest possible values of K are desirable for model robustness, a large value of K might exceedingly limit the surgeons' freedom of choice. As it can be observed in Figure 2.1, cluster homogeneity sharply decreases when K falls below 10. On the other hand, hospital managers pointed out that as many as 15 clusters are still acceptable by surgical teams. Hence, in order to generate meaningful but distinct scenarios, we considered the two values $K = 10$ and $K = 15$. In both cases, we separately considered two additional clusters corresponding to day surgeries, for both of which the length of stay is 1.

The descriptions of the clusters obtained for $K = 15$ and $K = 10$ are given in Table 2.4 and Table 2.5, respectively, where for each cluster we report the average surgical time and LOS, along with the range of values. Note that, the average $LOS(k)$ of a cluster k is, in general, a fractional value. However, the occupancy of the beds in the wards is only allowed for integer days. Therefore, the data is rounded to the closest integer in the optimization model.

2.6.3 Objective functions

In our experiments we considered three different values of α in the objective function (2.2):

- $\alpha = 1$, in this case we seek to maximize the overall OR utilization expressed in hours;
- $\alpha = 0$, in this case we seek to maximize the total score of selected clusters;
- $\alpha = 0.02$, in view of the numerical values of the two objectives, this roughly corresponds to giving the same weight to the two objectives.

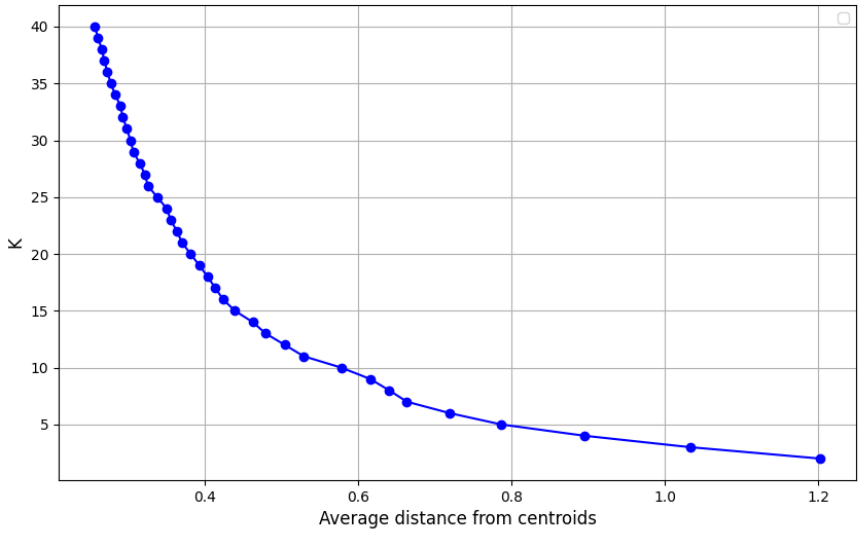

Figure 2.1: Cluster comparison based on mean intra-cluster distance.

k	$ST(k)$ (hrs)	range	$LOS(k)$ (days)	range
0	1.1	(0.1, 1.5)	1.15	(1, 2)
1	1.51	(0.5, 2)	3.35	(3, 5)
2	1.57	(0.5, 3)	6.83	(6, 7)
3	1.71	(1, 2.5)	2.15	(1.5, 2.5)
4	2.06	(1.5, 2.5)	1.16	(1, 1.5)
5	2.24	(1, 3)	4.69	(4, 6)
6	2.86	(2, 3.5)	3.33	(2, 4)
7	2.93	(2, 3.5)	6.73	(6, 7)
8	3.09	(2.5, 4.5)	1.66	(1, 2)
9	4.17	(3, 5.5)	4.62	(4, 6)
10	4.6	(3.5, 6)	2.49	(1, 3.5)
11	4.64	(3, 5.5)	6.82	(6, 7)
12	6.35	(5, 7)	6.77	(5, 7)
13	6.92	(5.5, 11)	3.65	(1, 4.5)
14	8.72	(7.5, 11)	6.75	(5, 7)
0D	0.69	(0.1, 1)	1.00	(1, 1)
1D	1.73	(1, 5)	1.00	(1, 1)

Table 2.4: Clusters when $K = 15$.

k	$ST(k)$ (hrs)	range	$LOS(k)$ (days)	range
0	1.25	(0.5, 2)	1.3	(1, 2)
1	1.54	(0.5, 2.5)	3.34	(2,5, 5)
2	1.73	(0.5, 3)	6.79	(6, 7)
3	2.51	(2, 4.5)	1.18	(1, 2)
4	2.63	(1, 4.5)	4.6	(4, 6)
5	2.69	(1.5, 4)	2.53	(2, 3.5)
6	3.29	(2.5, 5.5)	6.72	(5, 7)
7	5.02	(4, 10)	3.05	(1, 5)
8	5.54	(4.5, 7.5)	6.59	(5, 7)
9	8.15	(7, 11)	6.52	(4, 7)
0D	0.69	(0.1, 1)	1.00	(1, 1)
1D	1.73	(1, 5)	1.00	(1, 1)

Table 2.5: Clusters when $K = 10$.

2.6.4 Scenarios

As outlined in Section 2.5, we performed various one-year simulations, combining the following elements:

- Objective functions ($\alpha \in \{1, 0, 0.02\}$)
- Number of clusters ($K = \{10, 15\}$)

For the sake of clarity, we refer to a scenario also with the pair $(\alpha, |K|)$. The six scenarios have been compared with the real data, as reconstructed through the surgical records of the one-year time span under consideration. In this way, we obtained both an assessment of the overall approach vs. the current approach, as well as an indication of which choices are most profitable in the application of the model.

For each scenario, we evaluate the figures concerning the overall evolution of the surgical blocks, as described in Section 2.5, namely the number of surgical procedures performed (throughout the whole year), the average waiting time of selected surgeries, for each priority class, the average number of occupied beds over a two-week period, the average number of bed days exceeding the capacity across a two-week period, average room occupation, overtime and undertime.

In what follows we analyze the simulation on the various scenarios, and compare it with real-life performance data.

2.6.5 Results

We next present the results of the validation phase in each of the 6 scenarios, and in the real-world case.

Throughput

Table 2.6 shows the results concerning the number of surgeries performed in the various scenarios. The columns break down the results distinguishing different surgical categories, whereas each row refers to a single Scenario. Note that, in the results we present also "Real" scenario which reports the real results observed in the Siena University Hospital.

Scenario $(\alpha, \lvert K \rvert)$	ordinary	day-surgery	oncological	Total
1 (1,10)	5050	2505	1847	9402
2 (0,10)	4879	2910	1847	9636
3 (.02,10)	4879	2845	1847	9571
4 (1,15)	5050	2472	1847	9369
5 (0,15)	4950	2854	1847	9651
6 (.02,15)	4923	2856	1847	9626
Real	4999	2678	1847	9524

Table 2.6: Throughput.

From Table 2.6 we observe that when considering the total number of procedures carried out during the simulated period, most of the Scenarios are able to improve over the Real case (1.17%) which roughly corresponds to 2 additional procedures each week. Among the different proposed Scenarios the best performing ones in terms of throughput appear to be $(0, 10)$ and $(0, 15)$ highlighting as the total score maximization results in a larger number of surgeries. The smallest throughput is achieved with the OR occupancy objective. However, this objective is the only objective maximizing ordinary surgeries over day-surgeries. When considering the Oncological patients, we observe as all the Scenarios perform exactly the same number of surgeries, which is not surprising since Oncological surgeries are represented as a constraint in the mathematical model therefore they are guaranteed to be performed.

Flow time

Table 2.7 displays the values of the average flow time (expressed in days) of selected surgeries for the four classes A, B, C, D.

Scenario $(\alpha, \lvert K \rvert)$	A	B	C	D
1 (1,10)	23.42	25.38	43.73	71.03
2 (0,10)	21.56	21.57	34.65	61.78
3 (.02,10)	21.76	22.54	36.88	65.17
4 (1,15)	23.68	25.10	43.72	70.28
5 (0,15)	22.03	22.97	35.89	57.68
6 (.02,15)	21.61	22.58	36.41	60.17
Real	50.81	81.59	95.59	110.52

Table 2.7: Average flow time.

When considering the average flow time (Table 2.7) we observe a striking difference between the values of the Real case versus the simulated scenarios. The reduction in the average waiting time of the patients is even more than 50% for class A surgeries. These values are influenced by the fact that in the simulation, surgeries are prioritized based on scores. This may not fully reflect real-world practice, as surgeons' behavior also depends on factors related to the patients' specific clinical conditions and possibly on individual surgeons' selection schemes. Among the different scenarios, we observe again that the objective function is more impactful than the adopted number of clusters. As expected, the scenarios with objective 1 are consistently outperformed by the others. Scenarios with $\alpha = 0$ and $\alpha = 0.02$ yield comparable results for classes A and B, whereas $\alpha = 0$ seems to perform best for classes C and D. We observe that the effect of the number K of clusters is not particularly relevant.

Bed occupancy

Tables 2.8 and 2.9 show results related to bed occupancy in post-surgical wards. For each ward we report the capacity, and for each scenario the average number of occupied beds as well as the average number of bed/days exceeding capacity during a two-week period.

Wards	cap.	1(1,10)		2(0,10)		3(.02,10)	
		occ	ovfl	occ	ovfl	occ	ovfl
General surgery	45	23.84	0.00	22.95	0.00	23.15	0.00
Head/neck	10	5.62	0.05	5.58	0.05	5.47	0.03
Neurology	28	5.46	0.00	5.45	0.00	5.45	0.00
Cardiology	16	3.28	0.00	3.24	0.00	3.31	0.00
Thoracic surgery	8	4.20	0.00	4.19	0.00	4.19	0.00
Gynecology	8	5.19	0.07	4.65	0.02	4.63	0.02
Pediatric surgery	13	2.98	0.00	3.06	0.00	3.04	0.00
General surgery (DS)	8	4.04	0.00	4.33	0.00	4.23	0.00
Head/neck (DS)	4	1.63	0.00	1.66	0.00	1.66	0.00
Gynecology (DS)	6	0.69	0.00	1.26	0.00	1.26	0.00

Table 2.8: Bed occupancy (I).

When considering bed occupancy, we observe that the average over-occupation of the bed is small. Only in the Real scenario we observe an average overflow of 0.44 bed/night over a two-week planning period, occurring in the Gynecology ward. In practice, this means that every month bed occupancy exceeds the capacity of the Gynecology ward for 1 day. In simulated scenarios, this value never exceeds 0.07 and in most cases an overflow is never observed during the whole simulation period. When comparing the different scenarios we observe small differences. The most effective scenario is $(0, 15)$, although the differences are minimal. We can conclude that in the Siena University Hospital, the size of the wards is not underestimated.

Wards	cap.	4(1,15)		5(0,15)		6(.02,15)		Real	
		occ	ovfl	occ	ovfl	occ	ovfl	occ	ovfl
General surgery	45	24.26	0.00	24.14	0.00	23.9	0.00	17.25	0.00
Head/neck	10	5.58	0.02	5.53	0.01	5.54	0.02	4.30	0.04
Neurology	28	5.45	0.00	5.43	0.00	5.44	0.00	5.26	0.00
Cardiology	16	2.86	0.00	3.05	0.00	2.49	0.00	1.44	0.00
Thoracic surgery	8	4.19	0.00	4.19	0.00	4.19	0.00	3.05	0.00
Gynecology	8	5.22	0.02	4.7	0.01	4.75	0.01	5.36	0.44
Pediatric surgery	13	3.01	0.00	3.07	0.00	3.07	0.00	2.20	0.00
General surgery (DS)	8	3.95	0.00	4.2	0.00	4.25	0.00	1.02	0.00
Head/neck (DS)	4	1.62	0.00	1.66	0.00	1.67	0.00	0.28	0.00
Gynecology (DS)	6	0.64	0.00	1.25	0.00	1.22	0.00	0.30	0.00

Table 2.9: Bed occupancy (II).

OR utilization

Tables 2.10 shows the results related to operating room occupation, overtime and undertime across all surgical specialties. We report the average number of hours of occupation, overtime and undertime over a two-week period.

Scenario $(\alpha, \lvert K \rvert)$	occupation	overtime	undertime
1 (1,10)	878.09	5.06	52.52
2 (0,10)	869.48	5.01	60.52
3 (.02,10)	870.61	3.60	57.93
4 (1,15)	870.94	3.94	58.59
5 (0,15)	873.60	4.90	56.84
6 (.02,15)	868.42	3.63	60.26
Real	826.18	25.11	124.91

Table 2.10: Occupation, overtime and undertime.

On average, Scenario (1,10) exhibits the best performance in terms of operating room occupancy, as it is one of the scenarios that exclusively aims to maximize room utilization. It also results in the lowest undertime values, making it the most efficient overall. Conversely, Scenario (.02,10) performs best with respect to overtime, achieving the lowest values among all scenarios. The Real scenario, by contrast, displays a significantly different pattern: average occupancy is lower, while both overtime and undertime are higher. The quantity $\textit{occupation} + \textit{undertime} - \textit{overtime}$ remains approximately constant across all scenarios, as it corresponds to the average amount of time allocated to the various specialties, in all scenarios. This suggests that the overall system time balance is preserved, while scenarios differ in the way the workload is distributed among regular, under- and over-utilization. These findings may suggest some challenges in adhering to the MSS under real-world conditions.

2.7 Conclusions and Future Research

The computational experiments demonstrate that the proposed decision support framework can significantly improve hospital performance across multiple dimensions compared to current practices. In terms of surgical throughput, most scenarios achieved meaningful improvements, with increases corresponding to approximately two additional procedures per week. Scenarios prioritizing total score maximization exhibited the highest throughput, while those focusing on operating room occupancy maximization favored ordinary surgeries over day-surgeries. As expected, oncological surgeries were consistently performed across all scenarios due to their representation as hard constraints in the model. Regarding patient flow time, the framework achieved substantial reductions in average waiting times. These results reflect the score-based prioritization mechanism embedded in the model. However, it should be noted that real-world surgical selection also depends on factors related to individual patients' clinical conditions and surgeons' specific decision-making patterns, which may not be fully captured by the scoring system. Among the tested configurations, the choice of objective function proved more influential than the number of clusters employed. Analysis of bed occupancy in post-surgical wards revealed that capacity constraints are generally well-managed under the proposed framework, with minimal overflow observed across scenarios. This suggests that ward capacity is adequately dimensioned in the tested environment, and that the optimization model effectively respects downstream resource constraints. Concerning operating room utilization, scenarios explicitly targeting room occupancy maximization achieved the highest utilization rates and lowest undertime, while scenarios incorporating weighted objectives performed best in minimizing overtime. The framework maintained a consistent overall time balance across scenarios, with variations primarily reflecting different distributions between regular hours, undertime, and overtime. Observed differences between simulated scenarios and current practices may indicate challenges in adhering to the Master Surgical Schedule under real-world operational conditions, suggesting potential areas for process improvement. Overall, the results indicate that the proposed approach can enhance operational efficiency while maintaining quality of care, with the specific choice of objective function and clustering strategy offering flexibility to align with institutional priorities and operational contexts.

The proposed decision support framework offers several actionable insights for hospital managers and healthcare administrators. First, it provides a structured approach to balance centralized planning efficiency with the autonomy traditionally valued by surgical teams. By organizing patients into clinically meaningful clusters, the framework maintains surgeons' flexibility in patient selection while ensuring alignment with institutional priorities and resource constraints. Second, the model enables hospital management to systematically evaluate trade-offs between competing objectives (such as maximizing throughput, minimizing patient waiting times, or optimizing resource utilization), allowing them to tailor scheduling policies to their specific operational context and strategic goals. This flexibility is particularly valuable given the heterogeneity of healthcare settings and varying stakeholder priorities. Third, the framework facilitates proactive capacity planning by providing visibility into downstream resource requirements, such as post-surgical bed occupancy. This forward-looking perspective helps prevent bottlenecks and supports more

coordinated use of hospital resources across departments. Finally, the implementation demonstrates that data-driven scheduling approaches need not replace clinical judgment but can complement it by providing decision support that respects both operational efficiency and clinical considerations. This integration of quantitative optimization with practical flexibility represents a pragmatic pathway toward improved surgical planning that is more likely to gain acceptance among clinical staff.

To further assess the effectiveness of the cluster concept, future research could explore an experiment without clusters, thereby mimicking a purely centralized decision-making process in which managers select individual patients solely on the basis of system performance indices. Additionally, to better capture the reality (in which surgeons may select patients based on specific individual information regarding their current clinical condition), an alternative approach could involve randomly selecting a patient from among the first $n(i,k)$ patients in the respective waiting list from cluster k. Restricting the choice to these top-ranked patients would implicitly ensure that patients' priority scores are not disregarded.

Chapter 3

An integer linear programming model for the allocation of health services in Community Houses

This chapter presents a study focused on the allocation of resources within Community Houses (CHs). It examines how resources can be effectively distributed across a territory to meet the needs of residents. The analysis aims to provide insights into optimizing resource management, enhancing operational efficiency, and improving the overall quality of care provided within these community settings.

The chapter is structured as follows. Section 3.1 reviews relevant literature. Section 3.2 presents the modeling approach, followed by a detailed description of the mathematical formulation in Section 3.3. The case study application, along with results and managerial insights, is discussed in Section 3.4. Concluding remarks are provided in Section 3.5.

CHs are social-healthcare facilities, established by the Ministerial Decree (DM) 77/2022 to act as a point of reference for the citizens. They embody an integrated and multidisciplinary organizational care model in which multi-professional teams collaborate to deliver comprehensive services. The Decree characterizes CHs as "physical and easily identifiable locations where citizens can access health, social, and social assistance services." A principal objective of CHs is to alleviate the burden on hospitals and other healthcare facilities by providing timely and accessible care in closer proximity to patients' homes. This decentralization of healthcare services is designed to enhance the overall patient experience and improve health outcomes across the reference territory.

With the aim of ensuring widespread and equitable access to care, particularly in internal and rural areas, two organizational models have been defined for CHs, namely *hub CHs* and *spoke CHs*. Hub CHs are the main reference structures, offering a comprehensive suite of services, whereas spoke CHs provide a more limited range of services tailored to local needs.

Telemedicine plays a crucial role in optimizing healthcare service delivery within this framework. By transcending geographical barriers, telemedicine facilitates access to specialized care and enhances patient experience. It proves particularly valuable in chronic disease management, contributing to reduced healthcare costs, improved monitoring, and decreased demand for on-site staffing. In essence, telemedicine represents an effective and innovative approach with the potential to transform chronic disease management and promote equitable, patient-centered care while mitigating geographical and territorial disparities ([31], [32]).

This study focuses on a model evaluating the delivery of *outpatient specialist care* services with respect to the demand arising from a certain territorial area. Specifically, we consider the territory and facilities of a large Local Healthcare Unit, the South-East Tuscany Local Health Authority (Azienda Usl Toscana Sud Est, AUSL-TSE), which represents the largest healthcare authority in Tuscany. Its territory encompasses 99 municipalities across the provinces of Arezzo, Grosseto, and Siena, organized into 10 District Zones. The province of Arezzo comprises 36 municipalities distributed across 5 district zones (Casentino, Valtiberina, Valdichiana Aretina, Aretina, and Valdarno), while the province of Grosseto includes 28 municipalities organized into 2 zones (Colline dell'Albegna and Amiata Grossetana/Colline Metallifere/Grossetana). The province of Siena, encompassing the remaining 35 municipalities, is structured into 3 zones (Alta Val d'Elsa, Amiata senese/Val d'Orcia/Valdichiana senese, and Senese). We present a model that analyzes the alignment between demand for a designated set of outpatient services generated within this extensive and geographically diverse area and the overall capacity of healthcare structures. The objective is to develop a decision-support tool for personnel deployment across CHs, potentially identifying areas requiring enhanced service availability and revealing possible mismatches between demand and current service capacity across different territorial zones. Given the heterogeneous geographic configuration of this large territory, patients residing in different areas may experience varying levels of service access, with implications for equity and care quality. Our tool provides a quantitative evaluation of the overall social cost associated with a particular distribution of resources across CHs within the region.

Our model explicitly accounts for the opportunities offered by IT connectivity to fulfill the demand for services also through telemedicine and examines how these services can be adjusted to maximize benefits for the community. Moreover, it considers the presence of other public facilities in the area (e.g., hospitals) that provide outpatient services, including those potentially outside the jurisdiction of the territorial healthcare authority.

3.1 Literature review

CHs constitute an innovative component of the Italian healthcare system, developed in the context of a wider reorganization of local health services. As far as we are aware, the analysis regarding the allocation of resources to these facilities from an operations research perspective remains largely unexplored. [33] recently proposed an integrated decision support system for CHs planning, employing an integer linear programming (ILP) model that simultaneously addresses location and dimensioning decisions while accounting for accessibility and equity requirements.

The design of a CHs network naturally falls within the domain of facility location problems, which have been extensively studied in the literature with numerous applications documented in healthcare management over the past decades. Geographic accessibility plays a key role in the design of primary care systems, as the degree of penetration of territorial services is strongly correlated with geographic proximity. A comprehensive global study conducted by [34] mapped the time required to reach the closest healthcare facility from any point on Earth, revealing that while only approximately 9% of the

world's population cannot reach a healthcare facility within an hour when using motorized transportation, this proportion rises to approximately 43% for those who can only travel on foot. These findings underscore the critical importance of accessibility considerations in healthcare facility planning, particularly as inequitable access to healthcare services remains one of the most persistent challenges confronted by countries worldwide.

The spatial optimization of healthcare resources has emerged as a critical tool for addressing these accessibility challenges. [35] applied a two-step optimization approach to investigate the spatial allocation of newly added tertiary general healthcare resources in Chengdu, China, demonstrating how such methodologies can support regional health planning decisions. Their approach first optimized hospital locations to maximize population coverage, then assigned capacity to each facility to achieve equitable access across different residential locations. The results suggested that optimized spatial allocation could theoretically enhance efficiency through a 5% increase in population coverage and a 15% increase in the weighted median value of spatial accessibility, while simultaneously improving equity through a 27% decrease in the weighted standard deviation of spatial access. This work illustrates how rational allocation of newly added healthcare resources becomes critical when unevenly distributed resources pose substantial challenges to achieving cross-regional efficiency and equity.

Given the centrality of accessibility, the constraint of assignment to the closest facility constitutes a fundamental requirement in various healthcare system design problems. [36] implemented location-allocation models to address uncertainty in long-term hospital network planning, imposing that demand be met by the closest facility capable of satisfying that specific type of service. In contrast, [37] examined the design of mobile healthcare networks for disaster response, comparing centralized allocation models with those employing closest assignment constraints. Their findings demonstrated that while such constraints reduce travel costs, total costs often increase due to resulting imbalances among facilities, highlighting the trade-offs inherent in different allocation strategies.

The relationship between distance and service utilization has been further explored in preventive healthcare contexts. [38] considered distance to be of primary importance for service participation in their optimization method for preventive healthcare facility location. Similar to the CHs problem, they incorporated minimum client thresholds required to maintain facility operations. However, in contrast to geography-based allocation approaches, they assumed voluntary location choice by patients and did not enforce closest assignment constraints, reflecting a different philosophical approach to patient-facility matching.

While several studies have addressed problems sharing similarities with CHs network design, few consider the complete set of key features simultaneously. [39] presented a hierarchical facility location model for the public sector combining capacity with closest and single assignment constraints, allowing for facility opening and closing decisions across different services. However, their model did not explicitly incorporate opening and dimensioning costs, which are crucial considerations in real-world implementation.

Beyond traditional facility location considerations, emerging technological solutions are reshaping how healthcare resources can be allocated and delivered. [40] examined resource allocation decisions for hospitals operating both offline and online channels through

asynchronous telemedicine, addressing the challenge of insufficient and unevenly distributed healthcare resources. Their queuing network model, validated through empirical data from a collaborative hospital, revealed that e-visits displaced offline first-visits and substitute for offline return-visits, with the substitution effect decreasing as treatment complexity increases. Their analysis demonstrated that indiscriminate expansion of online channels may be ineffective, and proposed three scenario-dependent allocation strategies based on departmental treatment complexity, highlighting the need for nuanced approaches to channel capacity planning and workload distribution.

The challenge of healthcare resource allocation in underserved regions extends beyond facility location to encompass workforce planning as a critical component. [41] explored comprehensive strategies for strengthening healthcare access and equity in rural and remote areas, highlighting how workforce shortages, uneven distribution, and poor retention of health professionals exacerbate disparities in care. Their integrated framework emphasized the importance of adaptable, multimodal approaches combining supportive policy environments with community integration. Resource allocation in this context addresses infrastructure gaps through telemedicine, mobile clinics, and digital health investments. This holistic perspective underscores that effective healthcare delivery in underserved regions requires the strategic alignment of workforce planning with physical and digital resource allocation.

Within the context of facility dimensioning, the estimation of personnel needs in healthcare settings traditionally relies on three main criteria: the availability of current and future personnel, the demand for health services based on visit frequency, waiting lists, and demographic trends, and the expressed health needs of the reference population through satisfaction surveys and feedback. A notable example of this approach is the supply and demand forecast model developed by [42] for nurses, midwives, and radiographers at a Lithuanian University Hospital. Their model projects annual demand based on hospital data and national statistics alongside available supply, considering both a medium scenario with 1% yearly growth and fixed nurse-to-doctor ratios, and a prospective scenario with higher growth rates and increasing nurse-to-doctor ratios reaching 2.6 by 2030.

Alternative methodological approaches to staffing determination have also been developed and applied in primary care settings. [43] applied the WISN (Workload Indicators of Staffing Need) methodology, developed by the World Health Organization, to determine nurse requirements for primary healthcare in two São Paulo districts. Their approach was based on available working time and workload for both direct and indirect care activities, providing a workload-driven perspective on staffing needs.

The relationship between staffing levels and quality of care provides important validation for these planning approaches. [44] conducted a cross-sectional study involving 458 patients across three hospital units in Slovenia, examining the relationship between nurse staffing patterns and patient satisfaction. The results showed positive correlations between patient satisfaction and the percentage of registered nurse requirements met, nursing care hours per patient day, and the proportion of registered nurses in the nursing team. These findings demonstrate that adequate staffing, as determined through systematic planning methods, directly impacts the quality of care delivery and patient outcomes, reinforcing the importance of integrating workforce planning considerations into broader healthcare

resource allocation frameworks.

3.2 The modeling approach

The aim of our model is to assess to what extent a certain capacity of services offered in CHs matches the demand arising from a certain territory, and possibly support decisions concerning the deployment of personnel across CHs. The elements which are considered in our model are the following.

- *Model granularity*. The 99 municipalities of the provinces of Arezzo, Grosseto and Siena characterize both the location of residence of the patients, defined as *origins* in the context of this study, as well as the location of CHs. Population needs and delivery points are therefore assessed at the municipality level.

- *Community Houses*. We consider a set of CHs, either existing or prospective, each having a certain geographic location. CHs can be hub or spokes and may have different sizes and range of services provided, hence different capacities. For each CH, *capacities* have been identified. We consider a capacity for each service type (e.g. maximum number of cardiological visits that can be provided in one year in a certain CH). These data define whether or not a service can be provided in a CH, and to what extent. Note that in municipalities with more than one CH, we consider only one CH, assigning it a capacity equal to the total capacity of all CHs in that municipality. This is because our model accounts for distances between municipalities, but not within the same municipality.

- *Service types and services*. A *service type* is an activity provided to a patient, for instance ECG. The execution of a given service type on a patient is a *service*. A service requires a certain number of units of personnel and is characterized by a certain expected duration. For example, executing an Electrocardiogram on a patient requires 15 minutes and involves one nurse and one cardiologist, who are therefore busy for this amount of time. The service types we consider belong to the specialties of cardiology, diabetology and pneumology. For each service, the demand from the territories of residence of the patients over a one-year time horizon was taken into consideration. In particular, we considered the data from 1 January 2023 to 31 December 2023.

- *Resources*. Services are provided by professionals, grouped into *resource groups*. In our study, we consider four resource groups, namely cardiologists, diabetologists, pneumologists and nurses. For each CH and each resource group, an overall amount of hours is specified that can be devoted to service provision in that CH. Note that this amount does not include possible other activities which are carried out during the resource working hours but that are not related to service provision (e.g. administrative accomplishments).

- *Social cost*. The model aims to show how personnel resources can be matched to population health needs minimizing a measure of social cost. Here we consider

the *distance* from origins to CH that patients need to cover in order to receive the services. The distance is then used in the model to address various issues.

(*i*) The *overall* distance traveled by patients corresponds to the *average* disutility faced a patient. It is therefore an index related to the overall care accessibility.

(*ii*) The *maximum* distance a patient has to travel can be used to express an objective of equity, which is higher if no patient is forced to travel a long distance to receive the service.

We underline that the main goal of the model is not to provide an *absolute* evaluation of social costs, but rather to estimate the *relative* impact that a certain decision in the distribution of personnel resources may have, with respect to a current configuration.

- *Televisit modeling*. In our model we explicitly account for the opportunities offered by telemedicine. While in principle a patient may receive a service from home (e.g., cardiological televisit), for our purposes we assume that a patient may receive such a service in any CH equipped with appropriate internet connection (possibly a spoke CH). This means that a patient resident in town o may receive a certain service in a CH c, but the personnel involved in service delivery may be actually located in a different CH c' (typically farther than c from the patient's premises), or even in a facility which is not under the administration of a certain healthcare authority (e.g., a general hospital). Note that the same service may involve personnel located in different facilities or CHs. For example, if the patient is in CH c, an ECG may require assistance through a nurse in c, but it is then reported by a doctor in c'. In this chapter we consider that for a certain service, only physicians may remotely deliver a service, i.e., nurses are always in the same CH as the patient.

Data collection involved the extraction of internal data from AUSL-TSE through collaboration with administrators and professionals in the health sector to ensure accuracy and relevance of the information collected. Regular meetings and consultations with the working group helped ensure that the data reflect the actual conditions and needs of the health system.

In addition to internal data, official data sets from the website of the Tuscany Region and from ISTAT were used, as detailed in Section 3.4.2. This data provided additional information on the distances between municipalities in the reference area, allowing for a deeper understanding of demand.

3.3 The mathematical model

The model proposed can be synthetically described as follows. Given a certain description of production capacity, and given a certain demand for services, the model computes an allocation of the individual services requested by the patients to different CHs, so that the total distance traveled is minimized while preserving a certain level of equity.

A key issue is the representation of demand and of production capacity. The demand is represented by the number of services required by the population of each municipality

in a given time span. We call *planning horizon* such a time span (one year in our study). The capacity is represented by the total amount of time that each resource group may devote to various service types, in the same time span.

In our study, the values characterizing demand and capacity are derived from historical data, as explained in Section 3.2. However, different approaches can be used to the specification of these values, depending on the scope of the model. For instance, one might restrict the analysis to specific segments of populations (e.g. chronic patients, or certain types of chronic patients only), and consider, on the one hand, forecasted data from epidemiological models, and on the other hand an estimate of the fraction of time the various resources devote to this class of patients. The model structure is independent from the way demand and capacity values are assessed.

We propose a mixed-integer linear programming model, in which the variables express the allocation of capacity (from various resource groups) to the demand which originates in various municipalities. It is a mixed-integer model because the number of services received by certain sets of patients are integer numbers, while other quantities (hours of each resource group) may be considered continuous.

In what follows, we introduce the notation used in the model, and thereafter we illustrate the model in detail.

Data

The input to the model consists of the following data:

O set of municipalities in the area considered

C set of community houses (CHs). Each CH is located in one municipality of O.

C_{hub} set of hub CHs

S set of service types

S_{tele} subset of service types that can be provided through telemedicine

G set of resource groups

S_g ($g \in G$) subset of service types provided by resource group g

t_{oc} distance between origin $o \in O$ and community house $c \in C$ [*kilometers*]

v_{so} demand of services of type $s \in S$ originated by patients resident in $o \in O$ during the planning horizon [*number*]

d_{sg} time workload on resource g of a service of type s [*hours*]

d'_{sg} time workload on resource g of a service of type s provided in telemedicine [*hours*]

C_{sc} maximum number of services of type s which can be provided in CH c (service capacity) during the planning horizon [*number*]. Note that $C_{sc} = 0$ if CH c does not provide service type s

H_{gc} maximum time burden of resources of type g in CH c (resource capacity) during the planning horizon [*hours*]

W_g maximum amount of overtime resource group $g \in G$ can perform throughout the whole area during the planning horizon [*hours*]

ρ_{tele} for each service type that can be delivered through telemedicine and within each CH, this is the maximum fraction of services that can be remotely delivered out of the services provided to patients in presence. This is typically a choice of local healthcare managers. Keeping a percentage of visits on-site ensures that patients with complex clinical conditions can be cared for in the best possible way

θ maximum distance a patient should travel to receive a service.

Variables

The decision variables of the model are the following:

x_{gc} hours allocated to resource group $g \in G$ in CH $c \in C$

y_{soc} number of services of type $s \in S$ provided in $c \in C$ to patients resident in $o \in O$

y'_{socd} number of services of type $s \in S_{tele}$ provided to patients resident in $o \in O$ in CH $c \in C$ and involving a resource in $d \in C_{hub}$, with $c \neq d$

w_{gc} number of overtime hours performed by resource group $g \in G$ in CH $c \in C$

u_{gc} binary variable which is equal to 1 if all the H_{gc} regular hours of resource $g \in G$ in CH $c \in C$ are allocated, and 0 otherwise.

Objective function

The objective function is the total distance traveled by patients during the planning horizon. It is obtained considering both the services provided in presence and those provided remotely.

$$min \sum_{o} \sum_{c \in C} (\sum_{s} t_{o,c} y_{s,o,c} + \sum_{s \in S_{tele}} \sum_{d \in C_{hub}} t_{o,c} y'_{s,o,c,d}) \tag{3.1}$$

Constraints

We next introduce and illustrate the constraints of the model.

Constraints (3.2) and (3.3) apply to physicians, i.e., g denotes either pneumologists, cardiologists or diabetologists. The constraint imposes that the total number of hours (regular and overtime) worked by resource group g in the hub CH c is sufficient to provide all services provided by personnel in c. Note that services are delivered either in person (first term of the right hand side) or remotely (second term). The difference between (3.2)

and (3.3) is that the physicians in spoke CHs do not provide remote services. Hence, when g denotes a group of physicians, the following constraints hold:

$$x_{g,c} + w_{g,c} \geq \sum_{o} \Big(\sum_{s \in S_g} d_{s,g} y_{s,o,c} + \sum_{s \in S_{tele}} \sum_{d \in C} d'_{s,g} y'_{s,o,d,c} \Big) \quad \forall c \in C_{hub} \tag{3.2}$$

$$x_{g,c} + w_{g,c} \geq \sum_{s \in S_g} \sum_{o} d_{s,g} y_{s,o,c} \quad \forall c \notin C_{hub} \tag{3.3}$$

If g denotes a group of nurses, since they only provide service in presence, the analogous constraint is:

$$x_{g,c} + w_{g,c} \geq \sum_{o} \Big(\sum_{s \in S_g} d_{s,g} y_{s,o,c} + \sum_{s \in S_{tele}} \sum_{d \in C_{hub}} d'_{s,g} y'_{s,o,c,d} \Big) \quad \forall c \in C \tag{3.4}$$

The following constraints specify that the total number of services of type s provided to patients resident in o coincides with the corresponding demand $v_{s,o}$. The two constraints refer to services which cannot or, respectively, can be provided remotely.

$$\sum_{c \in C} y_{s,o,c} = v_{s,o} \quad \forall o \in O, s \notin S_{tele} \tag{3.5}$$

$$\sum_{c \in C} \Big(y_{s,o,c} + \sum_{d \in C_{hub}} y'_{s,o,c,d} \Big) = v_{s,o} \quad \forall o \in O, s \in S_{tele} \tag{3.6}$$

For each service type that can be delivered through telemedicine, the overall number of services remotely provided cannot exceed a certain share of the on-site number. In our study, this share has been fixed to 90% by the local healthcare authority:

$$\sum_{o} \sum_{d \in C_{hub}} y'_{s,o,c,d} \leq \rho_{tele} \sum_{o} y_{s,o,c} \quad \forall s \in S_{tele}, c \in C \tag{3.7}$$

The total time that can be allocated to each resource group g in CH c is denoted by H_{gc}. This number is obviously related to the maximum number C_{sc} of services of each type s in which the resource is involved. This quantity is different on whether telemedicine should or should not be considered. In particular, if $c \in C_{hub}$, for each resource group g one has

$$H_{gc} = \sum_{s \in S_g} (d_{s,g} C_{s,c} + d'_{s,g} C_{s,c}),$$

while if $c \notin C_{hub}$,

$$H_{gc} = \sum_{s \in S_g} (d_{s,g} C_{s,c}).$$

Hence, we limit the total time that can be allocated to each resource group in each delivery location. This limitation is given by the maximum capacity that can be provided for each service in each CH and is a function of the available instrumental resources and available clinics.

$$x_{g,c} \leq H_{gc} \quad \forall g \in G, c \in C \tag{3.8}$$

The following constraints impose that overtime can only be used once the overall amount

of regular hours (H_{gc}) is completely depleted:

$$x_{g,c} \geq u_{g,c} H_{gc} \quad \forall g \in G, c \in C \tag{3.9}$$
$$w_{g,c} \leq u_{g,c} W_g \quad \forall g \in G, c \in C \tag{3.10}$$
$$\sum_c w_{g,c} \leq W_g \quad \forall g \in G \tag{3.11}$$

In order to enforce the constraint on the maximum distance that should be traveled by a patient, we simply set $y_{soc} = 0$ and $y'_{socd} = 0$ whenever $t_{oc} > \theta$. Finally, limitations on variables are as follows.

$$x_{g,c} \geq 0 \quad \forall g \in G, c \in C \tag{3.12}$$
$$y_{s,o,c} \in \mathbf{Z}^+ \quad \forall s \in S, o \in O, c \in C \tag{3.13}$$
$$y'_{s,o,c,d} \in \mathbf{Z}^+ \quad \forall s \in S_{tele}, o \in O, c \in C, d \in C_{hub} \tag{3.14}$$
$$w_{g,c} \geq 0 \quad \forall g \in G, c \in C \tag{3.15}$$
$$u_{g,c} \in \{0, 1\} \quad \forall g \in G, c \in C \tag{3.16}$$

In conclusion, the optimization problem consists in minimizing (3.1) subject to constraints (3.2) to (3.16).

3.4 Application of the model to the territory of AUSL-TSE

The section is organized as follows. Section 3.4.1 and 3.4.2 detail the selection and characterization of healthcare services and their demand included in our analysis. In Section 3.4.3 we present the design of computational experiments conducted to test the model. In Section 3.5 we discuss the results obtained and their implications for healthcare management practice.

3.4.1 Service selection

We focused our attention on a set of services which are particularly relevant for a large class of chronic patients. We consider cardiological, diabetological and pneumological visits, as reported in Table 3.1.

As stated before, each service requires the involvement of a physician (depending on their specialty) and a nurse for a specified duration. Certain services can also be performed via telemedicine.

3.4.2 Demand

According to ISTAT data for total residents as of January 1, 2023, the AUSL-TSE territory encompasses a population of 334,052 in the province of Arezzo, 216,633 in Grosseto, and 260,557 in Siena [45]. This extensive territory is characterized by numerous sparsely

Specialty	Service type	Duration	Telemed.
Cardiology	Cardiology visit	20	
	Cardiology follow-up visit	20	x
	Stress echocardiography (exercise test)	20	
	Stress echocardiography (pharmacological test)	20	
	Two-dimensional echocardiography	20	
	Color Doppler echocardiography	20	
	Color Doppler echocardiography (special conditions)	20	
	Resting echocardiography	20	
	Cardiology consultation (first visit)	30	
	Cardiology consultation (follow-up visits)	30	x
	Electrocardiogram (ECG)	20	x
Pneumology	Pulmonology visit	20	
	Pulmonology follow-up visit	20	x
	Basic spirometry	20	
	Complete spirometry	20	
	Complete spirometry (plethysmography)	30	
	Bronchodilator reversibility test	30	
	Pulmonology consultation	40	x
Diabetology	Diabetology visit	20	
	Diabetology follow-up visit	20	x
	Remote diabetology follow-up visit	20	x
	Diabetes education therapy	30	x
	Remote diabetic foot assessment	30	x

Table 3.1: Service types and corresponding specialty and duration (minutes).

populated areas, with a population density of approximately 70 inhabitants per km^2, significantly below the regional average of 160 inhabitants per km^2 [46]. The demographic profile of this territory presents particular challenges for healthcare service organization and delivery. Tuscany ranks among Italy's oldest regions, and the AUSL-TSE area exhibits an even more pronounced aging trend: more than one in four individuals (27%) is over 65 years of age, while 5% of the population is 85 or older [47]. This progressive population aging, combined with the territory's sparse settlement patterns, represents a fundamental consideration for organizing and evaluating territorial health service performance. The aging demographic is intrinsically linked to an increasing prevalence of chronic diseases, which constitute a growing burden on healthcare systems. According to the PASSI surveillance system [48], approximately 18% of individuals aged 18 to 69 surveyed between 2023 and 2024 reported being affected by at least one chronic condition, with prevalence in Tuscany estimated at 16.2%. The prevalence of chronic conditions increases markedly with age, affecting 29% of individuals aged 50–69. Furthermore, multimorbidity, defined as the coexistence of two or more chronic conditions, is observed in approximately 4% of 18–69-year-olds, rising to 8% among those aged 50–69. These trends indicate that an increasing share of healthcare resources will require allocation to chronic disease management.

Given this epidemiological context, our study adopted a two-phase analytical approach. We commenced with a comprehensive analysis of overall healthcare demand across the

AUSL-TSE territory, establishing a foundation for understanding general service utilization and resource requirements. Subsequently, recognizing the strategic importance of chronic disease management in this aging population, we focused specifically on chronic patients as a priority target group. This population segment is particularly critical for healthcare planning due to both its growing prevalence and the predictable nature of its care needs.

Chronic patients require continuous, long-term management encompassing monitoring, consultations, medication, and coordinated care across multiple specialties. Inadequate care can precipitate disease progression, complications, emergency department visits, and hospitalizations, thereby increasing both patient burden and system costs. Conversely, the predictable nature of chronic care enables accurate demand forecasting, proactive resource allocation, and preventive interventions, enhancing system efficiency while reducing reliance on emergency services, an outcome particularly valuable in a geographically dispersed territory such as AUSL-TSE.

Our analysis of chronic conditions concentrated on patients with hypertension (without diabetes or cardiovascular disease), diabetes mellitus, a history of myocardial infarction, chronic heart failure, chronic obstructive pulmonary disease with respiratory failure, and non-valvular atrial fibrillation. Data were obtained from the Agenzia Regionale di Sanità della Toscana (ARS) [49] to quantify healthcare requirements across these patient populations.

3.4.3 Examples of model usage

We utilized the model described in Section 3.3 to explore potential applications of the model for organizational purposes. The computational setup consisted of a Windows Server 2022 Standard (version 21H2) environment, operating on hardware equipped with 128 GB of RAM and a 12th Gen Intel(R) Core(TM) i9-12900 CPU running at 2.40 GHz. The mathematical formulation is implemented in Python 3.11 using the PyCharm 2022.3.2 IDE. Optimization tasks were handled by the Gurobi solver, version 10.0.0.

Comparative analysis: Current State versus Optimal Allocation

The primary objective of our first set of experiments was to conduct a comprehensive comparison between the current state of healthcare service delivery and the optimal configuration that would be achieved through the application of our mathematical optimization model. Specifically, we sought to evaluate how the actual distribution of patients across CHs in 2023, referred to as the *State-of-the-art* scenario, compares with the theoretically optimal patient assignment patterns generated by our model.

This comparison serves multiple purposes. First, it quantifies the potential improvement achievable through optimization-based resource allocation. Second, it identifies inefficiencies in the current system, particularly regarding patient travel distances. Third, it provides healthcare managers with concrete evidence of the benefits that could be realized through more systematic planning approaches.

Our analysis utilized comprehensive patient distribution data from 2023, which documented precisely how many patients from each municipality within the AUSL-TSE territory received services at each CH. This granular data enabled us to calculate the

total distance traveled by patients under the current allocation patterns, establishing a baseline against which optimized scenarios could be compared. Using this same dataset as input, we then applied our mathematical model under various configuration scenarios, each designed to reveal different aspects of the system's performance and potential for improvement.

The first optimization scenario assumed unlimited service capacity at each CH, that is, we removed all constraints on the variable C_{sc} that typically bounds the number of services each facility can provide. This scenario represents a theoretical ideal and serves a critical analytical purpose: it establishes a lower bound for the objective function, representing the absolute minimum total patient travel distance that could theoretically be achieved if capacity constraints were not a limiting factor.

This lower bound is valuable because it reveals the maximum theoretical improvement possible purely through optimal patient assignment, independent of capacity limitations. Any gap between this lower bound and more realistic scenarios directly reflects the impact of capacity constraints on system performance.

To enable meaningful comparison with the State-of-the-art, we conducted experiments using the same capacity constraints C_{sc} that were implicitly present in the 2023 data. Specifically, we assumed that the number of services actually performed at each CH in 2023 represented the maximum feasible capacity at that facility. This assumption treats all facilities as operating at saturation, which is reasonable given the difficulty of distinguishing whether a particular service volume resulted from capacity limitations or from patient preference patterns.

We refer to these as *actual capacities* to distinguish them from the theoretical unlimited capacity scenario. This configuration allows us to compare the model's optimal patient assignment, given the same capacity constraints faced in reality, with the actual patient distribution observed in 2023.

Within the actual capacities scenario, we explored the potential impact of telemedicine availability. We ran the model twice: once with telemedicine enabled (allowing services listed in S_{tele} to be delivered remotely) and once with telemedicine disabled (requiring all services to be delivered in person). This comparison quantifies the potential benefits of telemedicine integration in terms of reduced patient travel burden.

Table 3.2 presents the total patient travel distances calculated under each scenario for the entire AUSL-TSE population, corresponding to 234,226 services provided. The results reveal substantial variation across scenarios and provide several important insights.

Under the State-of-the-art configuration, patients collectively traveled 4,455,900 kilometers in 2023. This figure represents the actual burden imposed on the population under current allocation patterns. When optimization is applied with unlimited capacities, the total distance drops dramatically to 1,246,762 kilometers. This substantial decrease demonstrates the theoretical maximum improvement achievable through optimal patient assignment.

However, when we impose the actual capacity constraints from 2023, the optimized distance increases to 2,755,447 kilometers with telemedicine enabled, or 3,038,891 kilometers without telemedicine. While these figures are higher than the Unlimited capacities scenario, they still represent substantial improvements over the State-of-the-art: ap-

		Total distance (km)
	State-of-the-art	4,455,900
Current CHs	Unlimited capacities	1,246,762
	Actual capacities (telemedicine-enabled)	2,755,447
	Actual capacities (telemedicine-disabled)	3,038,891
Proposed CHs	Unlimited capacities	1,170,437
	Hybrid capacities (telemedicine-enabled)	1,495,530
	Hybrid capacities (telemedicine-disabled)	1,639,967

Table 3.2: Total distance traveled under different scenarios (234,226 services provided), entire population of AUSL-TSE.

proximately 38% reduction with telemedicine and 32% reduction without telemedicine. This indicates that even within existing capacity constraints, considerable improvements are achievable through more systematic patient assignment strategies. The comparison between telemedicine-enabled and telemedicine-disabled scenarios reveals that remote service delivery could reduce total travel distance by approximately 283,444 kilometers (about 9%) under actual capacity constraints. This quantifies the potential value of investing in telemedicine infrastructure. Figure 3.1 illustrates the distribution of resource hours required across different CHs under the Unlimited capacities versus Actual capacities scenarios. This visualization reveals not only differences in total resource requirements but also how the spatial distribution of resources changes under different constraints. Such insights are valuable for understanding where capacity bottlenecks are most problematic and where expansion of services would yield the greatest benefits.

Alternative CH Network Configuration: Proposed CHs

The previous analysis examined optimization within the network of facilities that actually provided services in 2023, i.e. *Current CHs*. However, not all of these locations are officially designated CHs since some are other facilities that happen to provide primary care services. Furthermore, the AUSL-TSE has identified a proposed network of official CHs that represents the intended future configuration of the primary care system.

To evaluate this proposed network, we conducted a parallel set of experiments using only the official CHs operated by AUSL-TSE, which we refer to as *Proposed CHs*. This alternative configuration includes some new locations that were not operational in 2023 and excludes some locations that provided services in 2023 but were not designated as official CHs. Hospitals continue to be included in this analysis due to their significant ongoing role in service delivery. While one might consider optimizing the location of CHs themselves, determining where facilities should ideally be placed across the territory, such decisions were predetermined in our context. The proposed CH network consists of existing buildings that have been adapted for healthcare use rather than newly constructed facilities specifically designed and located for optimal coverage. Therefore, our analysis takes the proposed CH locations as given and focuses on optimal resource allocation across this predetermined network.

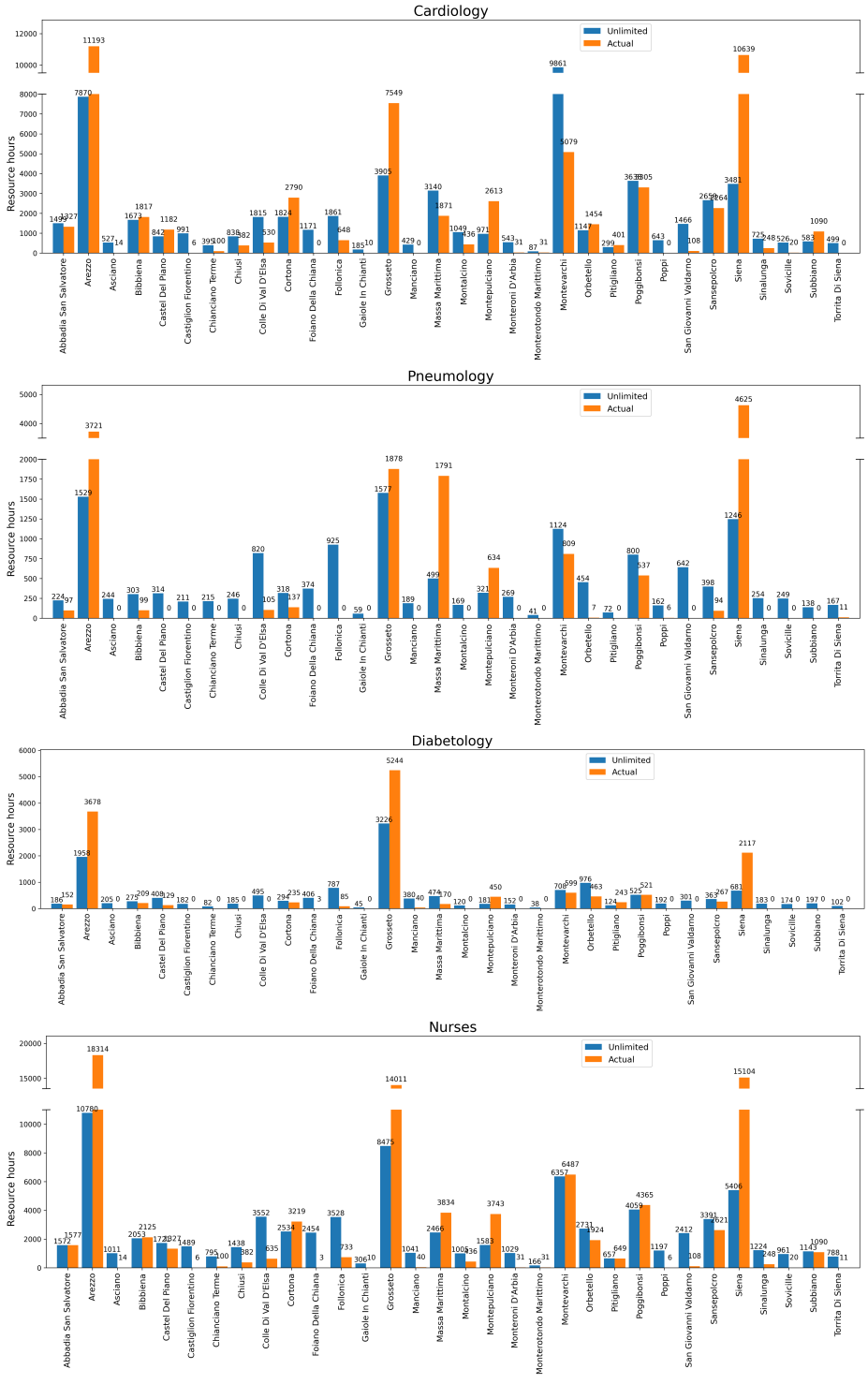

Figure 3.1: Distribution of resource hours in Unlimited and Actual capacities scenarios, with current CHs and enabling telemedicine.

For the Proposed CH network, we introduced a *hybrid capacity* approach that distinguishes between existing and new facilities. Specifically, existing CHs were assigned their actual capacities from 2023 (assuming they continue to operate at these levels), while newly proposed CHs were assigned unlimited capacities. This hybrid approach serves an important practical purpose: it allows the model to determine the optimal resource allocation for the new facilities. By leaving new CH capacities unconstrained, the model's solution reveals how many professional hours and which specific services should be provided at each new location to optimize overall system performance. This information is directly actionable for healthcare managers who must decide how to staff and equip the new facilities.

As shown in Table 3.2, the proposed CH network demonstrates improved performance compared to the current network across all scenarios. With unlimited capacities, the minimum achievable distance is 1,170,437 kilometers, slightly better than the 1,246,762 kilometers achieved with the current network. This improvement, though modest, indicates that the proposed network offers slightly better geographic coverage of the territory. More significantly, under the hybrid capacity scenario (actual capacities for existing CHs and unlimited for new ones), the total distance is 1,495,530 kilometers with telemedicine enabled and 1,639,967 kilometers without telemedicine. These figures represent substantial improvements over both the State-of-the-art (approximately 66% reduction) and the Actual capacities scenario with Current CHs (approximately 46% reduction with telemedicine). Figure 3.2 presents the distribution of resource hours across the Proposed CH network, comparing the Unlimited capacities scenario with the Hybrid capacities scenario. This visualization reveals how resources should be distributed across both existing and new facilities to achieve optimal performance. The differences between scenarios indicate where capacity constraints are most binding and where additional resources would provide the greatest marginal benefit.

Focus on Chronic Patients

Having established the general applicability and potential benefits of our model across the entire AUSL-TSE population, we now refine our analysis to focus specifically on chronic patients. They represent a particularly important population segment due to their predictable demand patterns, high service utilization rates, and the substantial healthcare system costs associated with inadequate management of their conditions. This refined focus allows us to provide more targeted and actionable insights for healthcare managers specifically concerned with optimizing care delivery for this critical patient population. By concentrating on chronic patients, we can develop resource allocation strategies that ensure the minimum documented demand of this group, which is of primary importance for healthcare system sustainability, is systematically and equitably met across the AUSL-TSE territory.

We applied the same analytical framework described above specifically to the chronic patient population, utilizing demand data for patients affected by the six chronic diseases listed in Section 3.4.2. This application generates optimized resource distribution strategies tailored to the specific needs and utilization patterns of chronic patients. We first

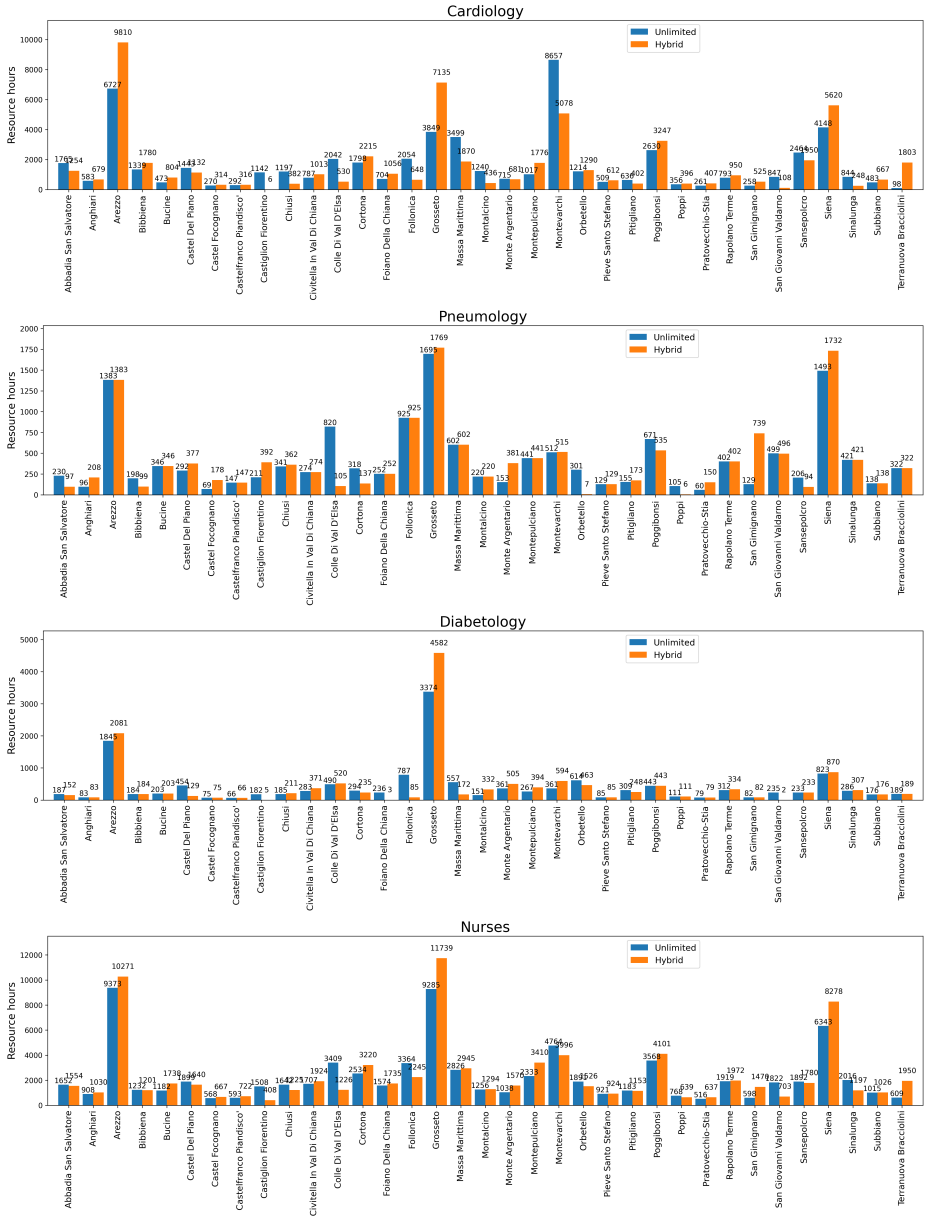

Figure 3.2: Distribution of resource hours in Unlimited and Hybrid capacities scenarios, with the CHs proposed and enabling telemedicine.

reconstructed the actual distribution of chronic patients across CHs as it occurred in 2023 (*State-of-the-art* scenario). It should be noted that this scenario considers the group of Current CHs, while subsequent scenarios are based on the Proposed CHs configuration. By quantifying the total patient travel burden under this configuration, we establish a reference point against which optimized scenarios can be compared.

We then applied our optimization model using the network of Proposed CHs, with and without telemedicine. As in previous analyses, we employed hybrid capacity constraints: existing CHs were assigned their actual 2023 capacities, while newly proposed CHs were given unlimited capacities, allowing the model to determine optimal resource allocation for these facilities.

When telemedicine is disabled, requiring all services to be delivered in person, the total travel distance increases to 1,357,732 kilometers, an increase of approximately 4.3%.

The results of this focused analysis provide healthcare managers with evidence-based guidance for allocating professional resources, determining service capacities, and structuring care delivery specifically for chronic disease management. By ensuring optimal resource distribution for this high-priority population, healthcare systems can simultaneously improve patient outcomes, enhance access to care, reduce unnecessary patient travel burden, and contain overall healthcare costs through prevention of acute exacerbations and hospitalizations. Table 3.2 shows the total distance traveled and Figure 3.3 presents the distribution of resource hours across the Proposed CH network.

		Total distance (km)
	State-of-the-art	3,144,457
Proposed CHs	Hybrid capacities (telemedicine-enabled)	1,301,856
	Hybrid capacities (telemedicine-disabled)	1,357,732

Table 3.3: Total distance traveled Hybrid capacities scenarios (109,109 services provided), chronic population.

The insights generated through these experiments demonstrate the practical utility of optimization-based approaches to healthcare resource allocation and establish a foundation for evidence-informed decision-making in primary care network design and management.

3.5 Conclusions

The application of the mathematical optimization model to the AUSL-TSE territory demonstrates significant potential for improving healthcare service delivery. Compared to the State-of-the-art scenario, optimization with actual capacity constraints reduces total distance travelled demonstrating that systematic resource assignment strategies can yield substantial benefits even within existing capacity limitations, representing significant gains in accessibility, environmental sustainability, and quality of life. The Proposed CHs network produces even more promising results. Using a hybrid capacity approach, the model achieves 66% reduction with telemedicine. These findings validate the AUSL-TSE's proposed configuration and provide specific guidance for resource allocation in new facilities. Telemedicine contributes meaningful improvements, reducing travel distance by 4-9% across scenarios.

The current analysis suggests directions for future research. The model assumes all facilities operate at saturation capacity and uses distance between municipalities as the sole accessibility metric, which may not fully reflect real-world barriers to access

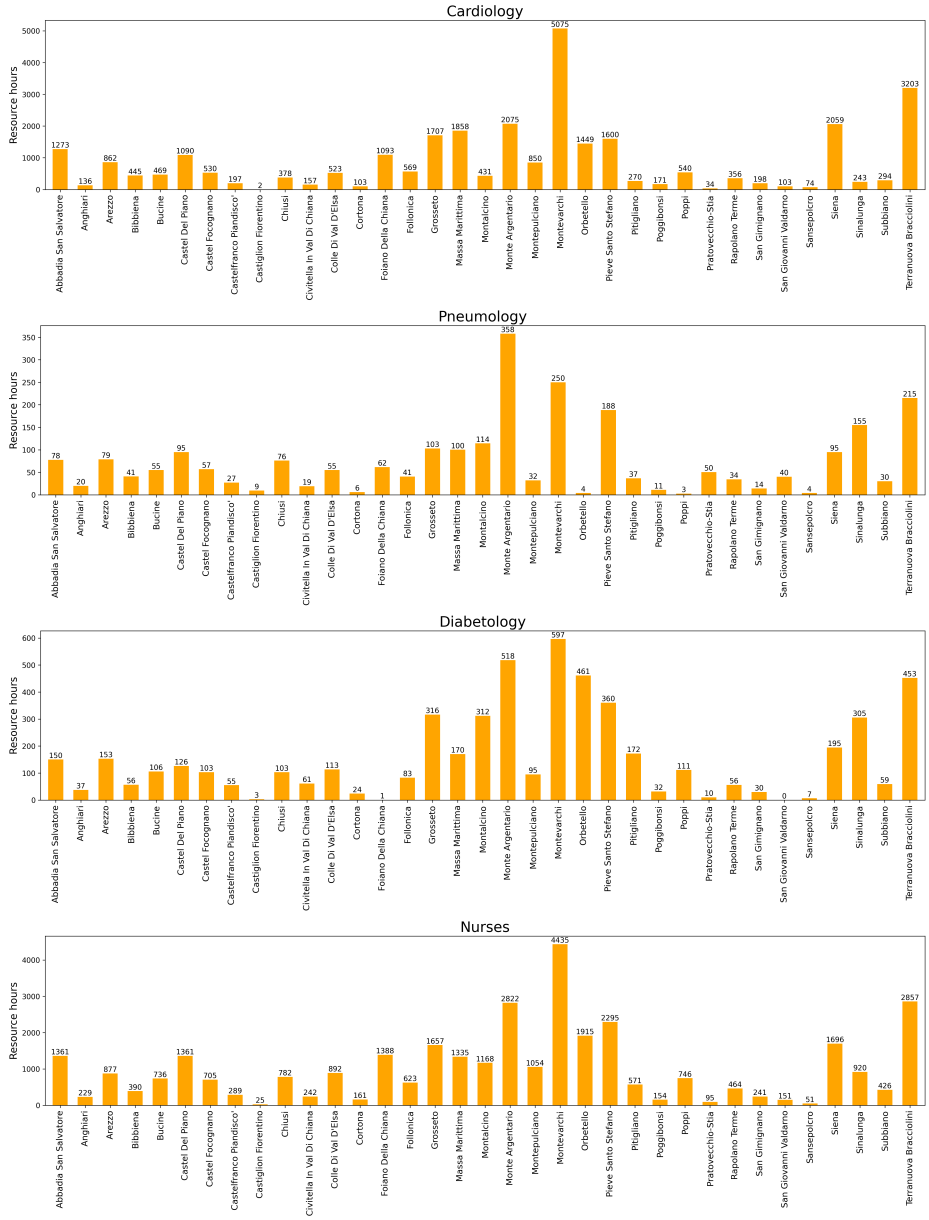

Figure 3.3: Distribution of resource hours in Hybrid capacities scenario considering chronic population.

such as transportation availability or travel time. A critical extension would involve introducing maximum distance constraints to ensure that no patient must repeatedly travel excessive distances for care. By progressively testing different distance thresholds, it could be possible to identify the minimum allowable distance that still permits all territorial demand to be satisfied within AUSL-TSE facilities. This approach balance

system-wide efficiency with principles of individual equity. It prevents situations in which optimization inadvertently creates significant burden for specific geographic areas, with solutions that are globally optimal but inequitable for particular groups. A multi-objective optimization framework could explicitly balance competing goals such as minimizing total distance, reducing maximum individual burden, and ensuring equitable distribution across municipalities. Such approaches would allow decision-makers to explore trade-offs between efficiency and equity according to policy priorities.

This work demonstrates that operations research techniques can generate substantial improvements in healthcare delivery efficiency and equity. The significant reduction in total patient travel distance represents significant gains in accessibility, environmental sustainability, and quality of life. However, practical implementation requires attention to operational constraints, patient preferences, and equity principles not fully captured by the current model. The proposed extensions aim to enhance realism and applicability while maintaining mathematical rigor. In a context of limited healthcare resources, such analytical tools provide valuable decision support. The ability to quantitatively evaluate alternatives, anticipate consequences, and identify configurations that best balance efficiency and equity represents an important step toward more rational, sustainable, and citizen-oriented territorial healthcare systems.

Part II

The impact of the number of preemptions in resource-constrained project scheduling problems with time-varying resources

Chapter 4

The impact of the number of preemptions in resource-constrained project scheduling problems with time-varying resources

In this chapter we present an integer linear programming (ILP) approach for the problem of minimizing project makespan and number of preemptions, and we show the results of a computational study on two large benchmarks. We address a number of issues, including the viability of our approach to derive Pareto optimal solutions, the insights about the benefits of preempting a few activities and how such benefit is related to structural problem properties. We also apply the approach to a real-life instance arising in the IT context.

The plan of the chapter is as follows. First we discuss relevant literature in this field (Section 4.1), and present our contribution. In Section 4.2 we formally introduce the problem and the notation. Section 4.3 presents the ILP formulations. Computational results and the case study are extensively presented in Sections 4.4 and 4.5 respectively. Finally, in Section 4.6 some conclusions are drawn.

Most of the literature on resource-constrained project scheduling problems (RCPSP) focuses on the minimization of the project makespan, assuming that, once started, an activity cannot be preempted. In real settings, it may be feasible or even necessary to preempt some activities, in order to comply with changing resource availability or to improve the makespan of the whole project. In any case, activity preemption is typically undesired, since it implies temporarily transferring resources from an activity which will require them again later on. Whenever a previously interrupted activity is resumed, it requires an effort to the project manager to re-align herself with the work to be done and it increases the risk of making mistakes. On the other hand, it can be useful to foresee how useful a certain number of preemptions can be in a typical context, also depending on the characteristics of resource availability. This leads us to consider the classical RCPSP from a bicriteria point of view, in which the two criteria are (i) the minimization of project makespan and (ii) the minimization of the total number of preemptions.

The study has been inspired by a real-life IT project management setting. A company produces devices for the fill-finishing process of injectable drugs. Due to the high degree of customization, each machine produced corresponds to a project. As in most IT projects, careful planning of human resources plays a key role, since different project activities may require skills owned by personnel from different departments of the firm. In what follows, we consider several types of human resources (hereafter simply referred to as

resources). One key feature of this IT setting is that human resources may be already partially engaged in previously scheduled projects, so that resource availability may not be constant over time, but the time intervals during which each unit of personnel may be engaged on the current project are known in advance. This is a fairly common issue in many fields of project management ([50], [51], [52]).

4.1 Literature review and present contribution

Within the huge literature on RCPSP and its numerous variants (see e.g. the recent survey by [53]), a significant subfield of research addresses the preemptive version of RCPSP. Various papers address the case in which there is no limit on the number of times any activity can be preempted. In one of the first papers on preemptive RCPSP, [54] present a branch and bound approach for the preemptive RCPSP, and apply it to a benchmark (instances with 7 - 51 jobs proposed by [55]). They report that only in 30 out of 110 instances there is some benefit, and even in these cases the benefit is very limited. However, if some form of resource variation is allowed over time, preemption helps in 74 instances, with a more significant makespan reduction in many cases. In this chapter we aim at performing a more in-depth analysis of the benefits which may arise from preemption as resource availability varies over time. [56] propose a branch-and-price algorithm and show that they are able to solve to optimality the preemptive version for a large number of benchmark instances.

Some models in the literature put restrictions on preemptions. [57] propose a fairly complex genetic algorithm to deal with a problem in which *each activity* can be preempted at most M times ($M = 0$, 1, 2 or 3 in the reported experiments). Similar assumptions and results are attained by [58], who also consider that each part of an activity must have a minimum duration. They propose a genetic algorithm, and quantify in about 2.5% the average makespan improvement over another genetic approach by [59], using the same number of explored schedules and allowed preemptions. [60] also consider a maximum number of allowed preemptions for each activity, and maximize an overall measure of robustness, while [61] allow at most one preemption per activity and propose a particle swarm optimization approach producing good solutions on benchmark instances.

Other contributions consider that whenever an activity is resumed, a setup time occurs ([62], [63]). A somewhat special type of preemption, referred to as *calendarization*, takes into account the fact that activities may have to be stopped and resumed because of calendar issues such as weekends or holidays, see e.g. [64]. Many other papers treat preemption in conjunction with other system features, e.g. multi-mode activity execution (e.g. [65]).

In this chapter we consider that activities can be preempted at any integer time point. This is not the most general setting – e.g., [66] and [67] address the case in which an activity can be preempted at any rational time instant. However, assuming that preemption only occurs at integer times is a fairly common assumption in RCPSP research, imposed to reduce complexity, in particular when addressing the problem by an ILP formulation approach (e.g. [54], [68]). However, while the previous literature addressing preemption restrictions focuses on a maximum number of preemptions *for each activity*, we consider

the total number of preemptions occurring *throughout the schedule*.

Another feature of the problems we consider is time-varying resource availability. This issue has been addressed by [52], who provides various applications with time-varying resources and proposes a genetic algorithm approach. He does not consider preemptions. Other papers jointly consider variable resource capacity and preemption, distinguishing between job interruption caused by changes in resource unavailability (*activity splitting*) and those related to precedence constraints. [69] analyze the effects of activity splitting, concluding that it may significantly reduce makespan, even without considering other preemptions. [70] perform a thorough study comparing the benefits of activity splitting and preemption, as well as the impact of various features of the precedence graph on these benefits. These two studies use an exact branch-and-bound algorithm, to solve instances with up to 13 and 16 activities respectively. They are not concerned with the *number* of activity splits or preemptions.

In this chapter we do not distinguish between preemptions due to changes in resource availability and general preemptions. Our viewpoint is that in both cases an activity has to be interrupted, with all the inconveniences that this entails. However, the actual *number* of times a project is preempted can indeed make a difference from the manager's standpoint. Hence, we want to investigate how beneficial a certain (possibly very limited) number of preemptions can be, and how this is related to some features of the problem, namely the availability and demand of resource(s). For this purpose, we use an adaptation of some summary measures introduced by [71] for the case of constant resource availability.

4.2 Problem definition and notation

A project is represented by an acyclic graph $G(V, E)$, in which the set of $|V| = n + 2$ nodes corresponds to activities (activities 0 and $n + 1$ are dummy) and each arc $(i, j) \in E$ indicates that activity j can only start after activity i is completed. There is a set K of renewable *resources*. Time is discretized in time slots, i.e., time slot t starts at time $t - 1$ and ends at time t. Each activity i requires a total (integer) number d_i of time slots (*activity duration*) to be performed and a_{ki} units of resource k ($k \in K$) during each time slot throughout its execution. During time slot t, there are n_{kt} units of resource k available, so the total usage of resource k in each time slot t must not exceed n_{kt} ($t = 1, \ldots, T$, $k \in K$).

Activities can be preempted, i.e., interrupted and resumed later. We assume that preemption can take place with no losses, i.e., regardless how many times an activity is preempted, the total amount of time during which an activity i is in execution must be equal to d_i. A *schedule* σ specifies the starting time and the duration of each portion of each activity execution. A schedule is feasible if all precedence and resource availability constraints hold, and all activities are entirely executed. In our experiments we assume that all activities can be preempted (an unlimited number of times), however the ILP models introduced later can be easily adjusted to the case in which certain activities can only be preempted a limited number of times (possibly zero), as in fact it occurs for some activities in the case study of Section 4.5.

Given a feasible schedule σ, in what follows we denote by $C_{max}(\sigma)$ the *makespan* of the

schedule and $Z(\sigma)$ its *total number of preemptions*. The project manager typically wants to minimize both of these performance measures, so we are interested in investigating *Pareto optimal schedules* (PO). Recall that if σ is PO, then no feasible schedule σ' exists such that $C_{max}(\sigma') \leq C_{max}(\sigma)$ and $Z(\sigma') \leq Z(\sigma)$ with at least one inequality being strict. The set of all PO schedules is also referred to as the *Pareto front*.

4.3 ILP formulations

Since we are dealing with a bicriteria problem, in order to find the whole Pareto front we adopt the ϵ-constraint approach (see [72]), i.e., we minimize one of the two objectives with a bound on the maximum value the other objective can attain. Suitably changing such a bound, we can generate all PO schedules. In this section we introduce the ILP formulations that will be used for this purpose.

We next introduce a formulation in which the objective is to minimize the number of preemptions with the bound that the project makespan C_{max} does not exceed a value Q. This can be viewed as a modification of the classical formulation by [73].

The time horizon is discretized in T time slots. For each pair (i,t) ($i = 1,\ldots,n$, $t = 1,\ldots,T$), we use binary variables $x_{it} = 1$ if activity i is in execution during time slot t. Moreover, we let the binary variable $z_{it} = 1$ if activity i is in execution in time slot t but not in $t+1$ ($t = 1,\ldots,T-1$). The problem can be addressed through the following formulation (denoted as MIN_PRMP).

$$\min \sum_{i=1}^{n} \sum_{t=1}^{T-1} z_{it} \tag{4.1}$$

$$\sum_{t=1}^{T} t x_{nt} \leq Q \tag{4.2}$$

$$z_{it} \geq x_{it} - x_{i,t+1} \quad \forall i \in V, \quad t = 1,\ldots,T-1 \tag{4.3}$$

$$\sum_{t=1}^{T} x_{it} = d_i \quad \forall i \in V \tag{4.4}$$

$$d_i x_{j,t+1} \leq \sum_{q=1}^{t} x_{iq} \quad \forall (i,j) \in E, \quad t = 1,\ldots,T-1 \tag{4.5}$$

$$\sum_{i=1}^{n} a_{ki} x_{it} \leq n_{kt} \quad \forall k \in K, \quad t = 1,\ldots,T \tag{4.6}$$

$$x_{it} \in \{0,1\} \qquad \forall i \in V, \quad t = 1,\ldots,T \tag{4.7}$$

$$z_{it} \in \{0,1\} \qquad \forall i \in V, \quad t = 1,\ldots,T-1 \tag{4.8}$$

The makespan constraint is modeled by (4.2). Constraints (4.3) set z_{it} variables to 1 exactly when $x_{it} = 1$ and $x_{i,t+1} = 0$, i.e., when the activity is either completed or preempted at time t. Activity duration is enforced by constraints (4.4). Precedence relations between activities are modeled by (4.5). In fact, if i must precede j ($(i,j) \in E$),

i must have been in execution for d_i periods before j can start. Constraints (4.6) enforce limited resource availability.

Variables z_{it} allow counting the number of preemptions. In fact, for a given feasible schedule, the actual number of preemptions is $\sum_{i=1}^{n}\sum_{t=1}^{T-1} z_{it} - n$, which explains the objective function (4.1). These variables can be discarded when computing the two extreme PO schedules, respectively the minimum-makespan nonpreemptive schedule and the minimum-makespan schedule with an unlimited number of preemptions. The former can be found using the following formulation which is close to the classical formulation by [73] (formulation NON_PRMP):

$$\min \sum_{t=1}^{T} t x_{nt} \tag{4.9}$$

$$d_i(x_{it} - x_{i,t+1}) - \sum_{q=\max\{1,t-d_i+1\}}^{t-1} x_{iq} \leq 1 \quad \forall i \in V, \quad t = 1,\dots,T-1 \tag{4.10}$$

$$(4.4), (4.5), (4.6)$$

$$x_{it} \in \{0,1\} \qquad \forall i \in V, \quad t = 1,\dots,T \tag{4.11}$$

Note that (4.9) equals the value of the makespan, while constraints (4.10) impose non-preemption. In fact, at time t one may have $x_{it} = 1$ and $x_{i,t+1} = 0$ only if activity i has been executed during the last d_i time slots.

The problem with an unlimited number of preemptions can be solved relaxing constraints (4.10) (formulation INF_PRMP):

$$\min \sum_{t=1}^{T} t x_{nt} \tag{4.12}$$

$$(4.4), (4.5), (4.6)$$

$$x_{it} \in \{0,1\} \qquad \forall i \in V, \quad t = 1,\dots,T \tag{4.13}$$

4.4 Computational experiments

The main purpose of our study is to assess how beneficial preemptions are to decrease the overall project duration. For this purpose, in this section we consider two public benchmarks, suitably modified to represent different scenarios in terms of resource availability. This will also allow us to test the effectiveness of the formulations in Section 4.3 in computing the Pareto front of the problem.

The experiments were conducted using a 12th Gen Intel(R) Core(TM) i9-12900 processor running at 2.40 GHz, with 128 GB of RAM. The operating system utilized was Windows Server 2022 Standard version 21H2. The formulations were implemented in Python 3.11 within PyCharm 2022.3.2 environment. Gurobi solver (version 10.0.0) was employed for optimization.

In what follows, we call *instance* an occurrence of RCPSP, defined by the activity graph, the duration and resource requirements of all the activities.

In our computational experiments we considered two RCPSP benchmarks containing medium-size instances, namely $n = 30$ for all the instances of both of them. These benchmarks are J30 (from [74] and available in the PSPLIB library[1]) and VNR[2], introduced by [71]. J30 is a collection of 480 instances, 4 resource types and each activity has at most 3 successors. This set of medium-size projects has been often employed in various papers for benchmark purposes (e.g. [75], [62]). VNR is a larger benchmark set, as it contains 1750 instances. In this set the number of resource types depends on the instance, ranging between 0 to 10 where '0' means that there are no resource restrictions. In our experiments, we excluded from VNR the 10 instances which do not require any resource, so the size of the benchmark becomes $480 + 1750 - 10 = 2220$.

The original J30 and VNR benchmarks are sets of instances for the nonpreemptive RCPSP in which renewable resources are constantly available over time. For our purposes, for each problem instance we consider several *resource scenarios* (simply referred to as *scenarios*), corresponding to different resource availability profiles over time. The scenarios are described in Section 4.4.1.

For each instance and each scenario, in order to compute the whole Pareto front, we must solve a number of *integer programs* MIN_PRMP (4.1)–(4.8), for different values of the bound Q on C_{max}. The first experiment consists in generating the whole Pareto front for each instance of J30 and VNR and for different resource availability scenarios, and suitably analyzing the results. The purpose is to figure out what is the typical makespan improvement due to preemption, and whether this gain is related to some structural parameter of the problem, in particular concerning resource requirements. In this respect we refer to a suitable adaptation of the parameter known as *resource strength* (introduced by [76] and adapted by [77]).

When resources are constantly available, the resource strength of a resource k is defined as

$$RS_k = \frac{n_k - r_k^{min}}{r_k^{max} - r_k^{min}}, \tag{4.14}$$

where r_k^{min} is the maximum amount of resource k demanded by some activity (hence, the minimum availability value for a feasible schedule to exist) and r_k^{max} the maximum demand for k in the earliest-start-schedule computed disregarding resource availability. The resource strength RS is the average value of RS_k over all resources. In our experiments we still define RS_k as in (4.14), but replacing n_k with $\bar{n}_k$, where the latter denotes the *average* availability of resource k over time. Notice that in this case, since resources vary over time, RS_k (and hence RS) may even attain negative values.

4.4.1 Resource availability scenarios

In the experiment we consider 11 different scenarios. These scenarios have been chosen so to mimic a phenomenon that can be found in real settings, in which some resources (typically,

[1] https://www.om-db.wi.tum.de/psplib/data.html
[2] available at https://www.projectmanagement.ugent.be/research/data

personnel) are engaged in different projects and therefore can be temporarily unavailable for a given project, as it occurs in the case study in Section 4.5. Therefore, we do not consider random resource variations, but rather we assume that resource capacity can be temporarily decreased, and investigate the impact of such variations on the makespan and computational issues (see Section 4.4.2).

For each instance of the two sets we considered a *baseline* scenario (or Scenario 0), in which each resource availability is constant over time and equal to the value it has in the original benchmark. The other scenarios are obtained from the baseline scenario defining a certain number of *variation intervals*, during which resource availability is temporarily decreased. A scenario is then characterized by three parameters, namely: *variation intensity*, i.e., the percentage decrease of resource availability with respect to the baseline during a variation interval, *variation lag*, i.e., the minimum time distance between two variation intervals, and *variation length*, i.e., the length of a variation interval.

To ensure that there always exists a feasible nonpreemptive schedule, we choose a lag which is not smaller than the longest activity, otherwise there might not be feasible solutions for the non-preemptive formulation.

In our experiment, each scenario applies to all resource types and is applied to all instances of J30 and VNR. The 10 scenarios with time-varying resources are the following, summarized in Table 4.1. Whenever reducing resource availability by a certain percentage results in a fractional value, it is rounded to the lower integer.

	variation intensity	variation lag	variation length
Scenario 1	20%	15	15
Scenario 2	20%	10	10
Scenario 3	20%	10	20
Scenario 4	20%	20	10
Scenario 5	20%	10 – 15	10 – 15
Scenario 6	40%	15	15
Scenario 7	40%	10	10
Scenario 8	40%	10	20
Scenario 9	40%	20	10
Scenario 10	40%	10 – 15	10 – 15

Table 4.1: Scenarios definition.

Scenario 1. Variation intensity is 20% (so, during a variation interval, resource availability is the 80% of the baseline), variation lag is 15 time slots and variation length is also 15 time slots. Hence, for each resource, its availability fluctuates between two values (the baseline and 80% of baseline) in a regular way.

Scenario 2. Also in this scenario variation intensity is 20%, but there are shorter and more frequent variation intervals. Variation lag is now 10 time units, as well as variation length.

Scenario 3. Variation intensity is 20%, variation lag is 10 time slots and variation

length is 20 time slots.

Scenario 4. Variation intensity is 20%, variation lag is 20 time slots and variation length is 10 time slots.

Scenario 5. This scenario can be seen as a combination of Scenarios 1 and 2. Variation intensity is still 20%, but lag and duration can be equal to 10 or 15, with an irregular pattern.

Scenarios 6, 7, 8, 9 and 10. These scenarios differ from Scenarios 1, 2, 3, 4 and 5, respectively, only in variation intensity, which is 40% instead of 20%.

4.4.2 Computational results

In each scenario and for each instance of J30 and VNR, in order to generate the whole Pareto front we have run several integer programs, one for each different value of Q.

We started by solving the nonpreemptive problem, running formulation NON_PRMP. For this purpose, in the baseline scenario, the value of T was fixed 10% higher than the value of the makespan publicly known on PSPLIB, to mimic a heuristic behaviour. In cases where the optimal value was not reported in the literature, a default value of 300 was assigned to T. Given that the value of the nonpreemptive makespan typically increases with reductions in resource availability, T was set to higher values for Scenarios 1–10. In particular, T was set to 300% of the optimal nonpreemptive makespan of the baseline in each scenario, except for Scenarios 3 and 8, where T was set to 450% (such a large increase allows to accomodate all the instances). However, as displayed in Table 4.2, the resulting average increase in the value of the nonpreemptive makespan compared to the baseline is actually much smaller. Notice that the largest increases occur for Scenarios 3 and 8, i.e. scenarios in which resource reduction occurs for the largest amount of time.

	Increase over baseline
Scenario 1	+39.71%
Scenario 2	+40.20%
Scenario 3	+71.75%
Scenario 4	+21.10%
Scenario 5	+38.08%
Scenario 6	+61.53%
Scenario 7	+61.71%
Scenario 8	+111.50%
Scenario 9	+34.81%
Scenario 10	+60.44%

Table 4.2: Average increase in the nonpreemptive makespan for Scenarios 1-10 compared to the baseline.

For each instance of each scenario, once the optimal nonpreemptive makespan $C^*_{max,NP}$ is computed, we have set T equal to $C^*_{max,NP}$ and we have run the formulation INF_PRMP, which allows computing the optimal preemptive makespan $C^*_{max,P}$. After these two extreme

makespan values are known, we have run several integer programs MIN_PRMP (setting $T = Q$), decreasing the bound Q on the makespan by 1 at a time, down to the value $C^*_{max,P}$. (Preliminary experiments showed that this procedure was computationally more efficient than alternating the solution of MILPs in which the roles of the objective function (4.1) and of the makespan constraint (4.2) are exchanged.) This allowed to reconstruct the whole Pareto set. In what follows we let Z_P denote the minimum number of preemptions such that the makespan equals $C^*_{max,P}$. In other words, the two extreme PO points of the Pareto front are $(0, C^*_{max,NP})$ and $(Z_P, C^*_{max,P})$.

For each ILP, we set a time limit of 120 seconds for the solution of each integer program. In fact, some preliminary experiments showed that ILPs which were not solved to optimality within two minutes would require a very long time to be solved. Throughout the whole experiment, a total of 85669 integer programs were run. Hereafter, we say that an instance is *solved* if we were able to compute the whole Pareto front, and *each* Pareto optimal schedule was found within 120 s.

Preliminary results suggested that the structural aspects of the activity graph do not seem to affect the hardness of an instance in a significant way, while resource availability does. In fact, the average value of RS for the MILPs solved within the time limit is 0.24, while it is only 0.01 for the other MILPs. On the whole, smaller values of resource strength (corresponding to scarce resources) significantly impact the possibility of reconstructing the whole Pareto front within the time limit imposed on each ILP.

The results for various scenarios are displayed in Figures 4.1 – 4.4 and in Tables 4.3 – 4.8.

The set of instances on which graphs and tables are based will be specified for each case. In particular, for each individual instance of RCPSP, the 11 different scenarios have been addressed. For each scenario, several MILPs have been solved. In our experiments, we only use the results from MILPs which were solved to optimality. More specifically: for the comparison of the Pareto fronts across scenarios (Figure 4.1) we only considered the instances for which the whole Pareto front has been found *in all scenarios*. In all other cases, separately for each scenario, we used all the instances for which the whole Pareto front was available *for that scenario*; the number of such instances is generally different across scenarios. In any case, for a given scenario, instances for which, at least in one case, the optimal solution of an ILP was not certified were discarded.

- *Size of the Pareto front.* Figure 4.1 shows the distribution of the size of the Pareto front in the various scenarios. This picture only refers to the instances for which the whole Pareto front was computed *in all scenarios* (600 instances). The distribution in the baseline scenario is very different from the other scenarios. We note that for 544 instances, in the baseline scenario there is a single PO schedule – i.e., preemption does not help. However, this number sharply decreases when there is a 20% resource reduction (Scenarios 1, 2, 3, 4, 5) and even more for a 40% reduction (Scenarios 6, 7, 8, 9, 10). The fact that the average size of the Pareto front is larger for more constrained scenarios suggests that multiple preemptions are more likely to be beneficial when a significant reduction in resource availability occurs. Also notice that Scenarios 3 and 4, as well as Scenarios 8 and 9, exhibit the lowest number

of Pareto optimal solutions among the scenarios with the same level of variation intensity.

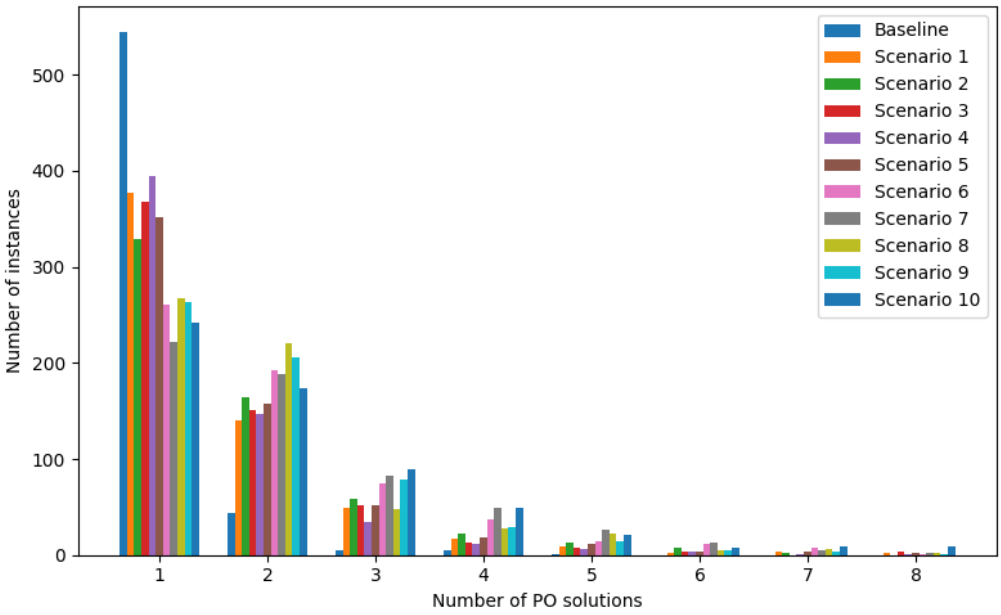

Figure 4.1: Size of the Pareto front in the various scenarios.

- *Gain for a given number of preemptions.* The scatter plots in Figure 4.2 illustrate the percentage gain (w.r.t. the nonpreemptive solution) achieved in each scenario by a certain number of preemptions. Here each point corresponds to a solution of the Pareto front. Recall that, for each scenario, all the instances for which all the Pareto front was obtained were considered. The plot shows how the points spread out for an increasingly large resource reduction, indicating a larger gain. The difference is especially apparent between the baseline scenario and the scenarios with 20% of reduction intensity, less so between 20% and 40%. Although there are not dramatic differences across scenarios, those corresponding to shorter lags (Scenarios 2, 3, 5, 7, 8, 10) typically entail larger gains.

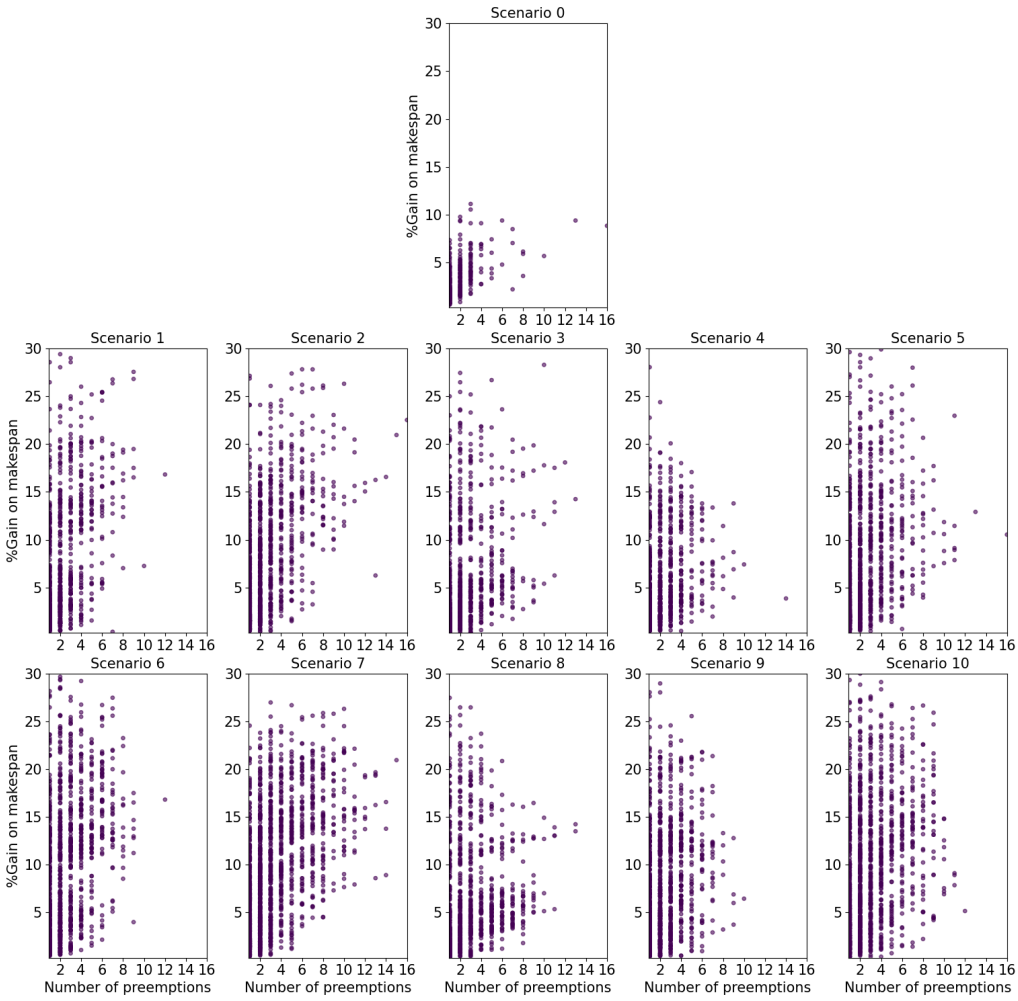

Figure 4.2: Number of preemptions and percentage gain on makespan.

- *Relationship with resource strength.* The scatter plots in Figure 4.3 relate RS to the gain achievable with an unrestricted number of preemptions. Here each point corresponds to an instance which was solved exactly. It is interesting to observe that while, in the baseline scenario, RS is a number between 0 and 1 (as implied by (4.14)), in more constrained scenarios RS expands to negative values, especially in Scenarios 6 – 10 (recall what observed about (4.14)). A low value of RS denotes closeness between the average amount of resource availability and the minimum amount needed for feasibility. Hence, as expected, on the average lower values of RS correspond to situations in which there is more convenience in preempting. Instances are more likely to benefit from preemption when resources are more binding, i.e., for smaller values of RS. In fact, the average value of RS for instances for which preemption does help (i.e., instances in which the size of the Pareto front is at least

2) is 0.04, while for instances in which preemption does not bring any benefit is 0.43. There is also a difference in the size of the Pareto front for different values of RS (Figure 4.4), namely for low values of RS there are more Pareto optimal solutions. This applies to all scenarios but the behavior is more apparent as resource availability decreases.

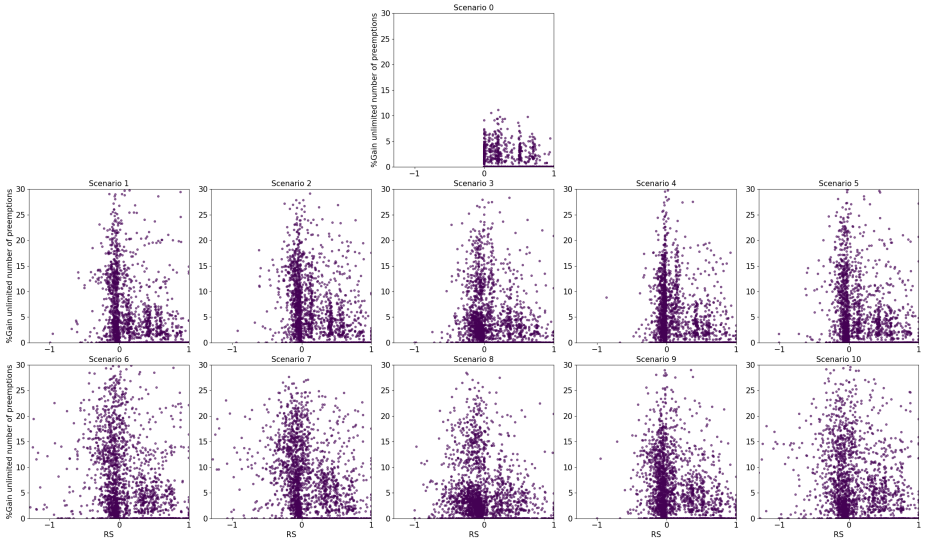

Figure 4.3: RS and percentage gain on nonpreemptive makespan with unlimited number of preemptions.

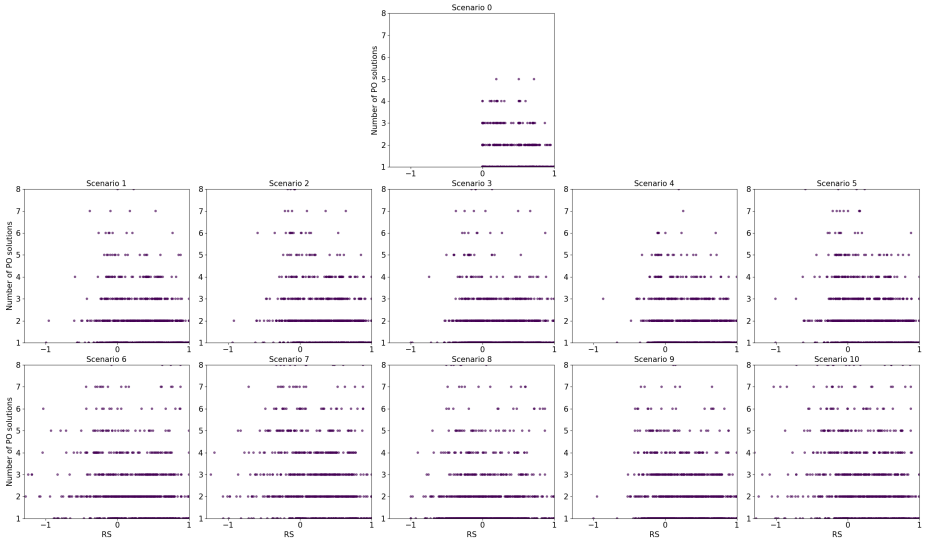

Figure 4.4: RS and size of the Pareto front.

We next perform a more careful analysis concerning the impact of preemptions on project makespan. In what follows, for a given scenario, let J_r be the set of instances for which a PO schedule with *at least* r preemptions exists. The results are reported in Tables 4.3 – 4.7. In these tables, for each scenario, column 1 shows the total number of instances solved. Columns 2 and 3 report the number r of preemptions, and, respectively, the size of J_r. Notice that J_r also includes the instances for which no PO schedule with *exactly* r preemptions exists, but $Z_P > r$ (recall that for a certain instance, Z_P is the largest number of preemptions bringing any benefit). Column 4 indicates the average makespan improvement of a schedule with at least r preemptions w.r.t. the corresponding nonpreemptive schedule, and column 5 expresses the same improvement in relative terms (%). The subsequent columns 6 and 7 express the *marginal* gain achieved by a preemption, i.e., given r, we consider the gain of a PO schedule with *exactly* r preemptions w.r.t. the PO schedule with the largest number of preemptions strictly less than r.

	# of preempt. r	# of instances $\|J_r\|$	Gain w.r.t. nonpreemptive schedule	%Gain w.r.t. nonpreemptive schedule	Gain w.r.t. preceding PO schedule	%Gain w.r.t. preceding PO schedule
Scenario 0	1	257	1.38	2.00	1.38	2.00
1162	2	110	1.79	2.97	0.82	1.36
	3	61	1.41	2.67	0.50	0.95
	4	26	2.04	4.18	0.54	0.99
	5	16	1.69	3.58	0.25	0.49
	6	11	1.73	3.90	0.18	0.50
	7	9	1.89	3.82	0.33	0.62
	8	6	2.17	4.07	1.17	1.62
	10	3	2.00	4.83	1.00	1.89
	13	2	3.00	9.10	1.50	4.69
	16	1	3.00	8.82	1.00	2.94

Table 4.3: Gains and marginal gains on makespan in the baseline scenario.

For example, consider Scenario 1 in Table 4.4. We see that for 151 instances a PO schedule with at least 3 preemptions exists. On average, having *at least* 3 preemptions allows reducing the makespan by 7.98% with respect to the nonpreemptive value $C_{max,NP}$, and the marginal contribution of the third preemption (still w.r.t. $C_{max,NP}$) is 3.21%.

- *Preemptions and variation intensity*. The utility of preempting some activity appears strictly related to intensity variation. We observe that preemption determines some gain in 22.1% of the instances (257/1162) in the baseline scenario (Table 4.3). Considering scenarios with 20% of intensity variation, we observe that this figure increases to 53.8% for Scenario 1, 63.8% for Scenario 2, 53.7% for Scenario 3, 49.7% for Scenario 4 and 54.5% for Scenario 5 (Tables 4.4 – 4.5). When considering scenarios with 40% variation it further grows to 72.2% for Scenario 6, 83.9% for Scenario 7, 64.1% for Scenario 8, 65.8% for Scenario 9 and 77.0% for Scenario 10 (Tables 4.6 – 4.7). These results suggest that besides of course variation intensity, the scenarios that most largely benefit from preemption are those with small lags

	# of preempt. r	# of instances $\lvert J_r \rvert$	Gain w.r.t. nonpreemptive schedule	%Gain w.r.t. nonpreemptive schedule	Gain w.r.t. preceding PO schedule	%Gain w.r.t. preceding PO schedule
Scenario 1	1	527	4.92	3.41	4.92	3.41
979	2	294	9.38	5.46	5.26	2.95
	3	151	15.98	7.98	6.75	3.21
	4	84	20.58	9.07	4.49	1.99
	5	52	26.23	10.69	4.88	1.86
	6	34	36.68	14.66	4.14	2.03
	7	16	36.88	15.19	1.13	0.92
	8	9	41.11	16.92	0.89	0.41
	9	5	52.00	21.58	1.20	0.50
	10	3	31.00	17.03	0.33	0.58
	11	2	47.50	24.40	3.00	2.36
	12	1	55.00	16.82	1.00	0.31
Scenario 2	1	577	4.68	3.53	4.68	3.53
904	2	338	9.54	5.67	5.28	3.05
	3	198	14.58	7.88	5.69	3.04
	4	108	18.44	9.14	4.19	2.02
	5	71	23.28	10.62	2.61	1.29
	6	46	30.26	13.25	2.50	1.12
	7	31	31.45	13.61	1.81	0.88
	8	16	42.75	16.12	5.75	1.87
	9	12	36.42	14.61	3.67	1.38
	10	7	42.86	17.47	1.29	0.50
	11	5	42.80	17.30	3.00	1.00
	12	4	42.25	18.27	0.50	0.15
	13	3	39.67	19.45	0.33	0.10
	14	3	40.00	19.55	0.33	0.10
	15	2	34.00	21.29	0.50	0.24
	16	1	25.00	22.52	1.00	0.90

Table 4.4: Gains and marginal gains on makespan in scenarios with 20% resource reduction and equal lag and length.

and durations (2 and 7), i.e., when resource availability changes frequently.

- *Unlimited preemptions.* Column 5 of Table 4.8 illustrates the average gain of the preemptive schedule (no limits on the number of preemptions) with respect to the nonpreemptive schedule. These values have been calculated by averaging the gains obtained for each instance allowing unlimited interruptions, on the total of the instances for which it is useful to preempt. In the baseline scenario, the average gain of a preemptive solution is only 3.21%, while it is considerably higher in all other scenarios. Comparing these values with columns 2, 3, 4 of Table 4.8, respectively reporting the average percentage gains achievable with at most 1, 2, 3 preemptions, we see that the difference among scenarios becomes more significant as the intensity

	# of preempt. r	# of instances $\lvert J_r \rvert$	Gain w.r.t. nonpreemptive schedule	%Gain w.r.t. nonpreemptive schedule	Gain w.r.t. preceding PO schedule	%Gain w.r.t. preceding PO schedule
Scenario 3	1	617	4.41	2.79	4.41	2.79
1149	2	353	7.95	4.33	5.01	2.62
	3	199	10.10	4.51	4.13	1.82
	4	132	10.50	4.30	3.27	1.37
	5	94	13.89	5.45	5.09	1.93
	6	58	13.53	5.45	3.98	1.62
	7	37	15.59	5.80	6.46	2.60
	8	16	16.75	6.21	2.44	1.12
	9	15	20.27	7.98	4.33	1.91
	10	8	29.50	12.37	10.00	4.32
	11	4	32.75	12.65	1.50	0.55
	12	2	37.50	16.00	0.50	0.28
	13	1	44.00	14.24	1.00	0.32
Scenario 4	1	570	3.75	2.99	3.75	2.99
1147	2	333	5.25	3.89	3.17	2.30
	3	202	8.15	5.17	4.12	2.57
	4	117	9.06	5.66	4.12	2.55
	5	60	9.87	5.84	4.42	2.51
	6	38	10.95	6.21	5.53	2.79
	7	19	6.47	5.30	2.42	2.20
	8	8	8.50	7.08	2.88	2.71
	9	5	10.80	7.90	1.40	1.70
	10	1	6.00	7.41	1.00	1.23
	14	1	3.85	2.00	3.85	1.00
Scenario 5	1	601	5.74	4.02	5.74	4.02
1103	2	370	10.78	6.21	5.23	2.85
	3	221	14.96	8.13	5.06	2.58
	4	147	19.48	9.44	5.16	2.20
	5	84	20.45	9.37	4.32	1.79
	6	54	22.70	10.20	3.87	1.64
	7	35	24.14	11.00	3.80	1.51
	8	23	25.91	10.75	2.17	0.78
	9	11	29.55	13.33	4.36	1.41
	10	8	34.00	16.24	0.63	0.20
	11	6	25.67	15.40	2.33	1.11
	13	2	3.50	10.40	0.50	1.61
	16	1	4.00	10.53	1.00	2.63

Table 4.5: Gains and marginal gains on makespan in scenarios with 20% resource reduction and different lag and length.

of variation increases. Notice that in the baseline scenario, the first preemption achieves 62% of the gain attainable with unlimited preemptions. This gain grows to

	# of preempt. r	# of instances $\lvert J_r \rvert$	Gain w.r.t. nonpreemptive schedule	%Gain w.r.t. nonpreemptive schedule	Gain w.r.t. preceding PO schedule	%Gain w.r.t. preceding PO schedule
Scenario 6	1	643	6.53	4.35	6.53	4.35
891	2	415	12.29	6.92	6.84	3.79
	3	245	18.96	8.98	8.64	3.81
	4	156	25.36	11.26	7.29	3.30
	5	96	30.50	12.20	4.48	1.74
	6	64	38.10	14.53	4.34	1.84
	7	35	40.28	14.98	1.50	0.86
	8	14	48.00	15.97	4.46	1.37
	9	8	39.89	13.76	2.78	1.06
	12	1	55.00	16.82	1.00	0.31
Scenario 7	1	694	5.50	3.98	5.50	3.98
827	2	492	10.29	6.60	5.01	3.05
	3	341	14.90	8.25	5.12	2.68
	4	218	18.80	9.51	3.68	2.06
	5	155	23.10	10.30	3.83	1.66
	6	120	27.88	11.72	3.04	1.34
	7	96	33.23	13.44	3.78	1.68
	8	69	39.77	14.83	3.57	1.37
	9	39	48.15	16.15	2.85	0.90
	10	31	49.71	16.62	3.29	1.03
	11	18	48.06	15.98	2.94	0.94
	12	9	48.56	16.61	0.56	0.17
	13	7	49.43	17.34	0.71	0.25
	14	4	45.50	15.96	0.75	0.23
	15	1	44.00	20.95	1.00	0.48

Table 4.6: Gains and marginal gains on makespan in scenarios with 40% resource reduction and equal lag and length.

80% considering also the second preemption and 87.5% with the third. As expected, when resources vary over time, since a larger number of preemptions is still beneficial, the share of the first preemption is comparatively lower (e.g. 50% in Scenario 1, 45% in Scenario 6), and similarly for subsequent preemptions.

- *Marginal utility of a preemption.* Let us now consider the marginal impact of preemptions, i.e., the benefit achieved by one preemption over the previous. As already observed, most often the *first* preemption leads to the most significant marginal improvement on makespan, compared to subsequent preemptions (if any). This outcome remains consistent across all scenarios, although the gain is more apparent as variation intensity increases. Although the trend is not strictly monotonic, most often subsequent preemptions display decreasing marginal returns, as typical of limited resource settings. Notice however that especially for large values of r we encounter some uneven results, e.g. for $r \geq 10$ in the baseline scenario. These

	# of preempt. r	# of instances $\lvert J_r \rvert$	Gain w.r.t. nonpreemptive schedule	%Gain w.r.t. nonpreemptive schedule	Gain w.r.t. preceding PO schedule	%Gain w.r.t. preceding PO schedule
Scenario 8	1	634	4.56	2.82	4.56	2.82
989	2	414	7.85	3.89	4.74	2.28
	3	275	10.00	4.61	4.37	2.09
	4	189	11.94	4.94	4.87	2.12
	5	137	12.23	4.64	3.76	1.52
	6	93	14.03	4.89	4.38	1.63
	7	63	16.21	5.30	4.11	1.31
	8	38	17.32	5.47	4.61	1.53
	9	19	21.53	6.71	4.47	1.22
	10	9	24.11	7.89	10.44	3.67
	11	5	35.80	11.71	15.80	5.47
	13	2	64.50	20.81	1.50	0.48
Scenario 9	1	740	4.91	3.75	4.91	3.75
1125	2	463	7.82	5.29	4.42	2.96
	3	292	9.86	6.00	4.15	2.59
	4	176	12.74	7.01	4.42	2.52
	5	107	13.16	6.69	3.25	1.79
	6	59	16.22	7.49	7.02	3.19
	7	29	20.54	9.25	10.65	4.48
	8	14	12.14	4.75	7.93	2.98
	9	5	11.00	5.88	1.40	1.58
	10	1	4.00	6.45	4.00	6.45
Scenario 10	1	657	6.72	4.66	6.72	4.66
853	2	451	12.34	7.50	5.81	3.41
	3	290	17.51	9.33	5.91	2.96
	4	180	21.39	10.50	3.40	1.67
	5	120	24.58	11.07	1.88	0.92
	6	89	29.74	12.28	2.35	0.96
	7	66	33.53	13.39	2.71	1.29
	8	40	36.08	13.99	2.83	1.20
	9	29	39.45	14.08	2.10	0.71
	10	12	35.00	13.35	0.83	0.34
	11	4	25.50	10.05	3.25	1.48
	12	1	16.00	5.16	1.00	0.32

Table 4.7: Gains and marginal gains on makespan in scenarios with 40% resource reduction and different lag and length.

occasionally large marginal gains are related to very few, very specific instances (as reported in column 3). In any case, such large marginal gains for large r appear to have limited statistical relevance with respect to the values attained for small values of r.

- *Preemption patterns.* While this feature is not reported in the tables, we have also carefully analyzed the preemption pattern of the various PO schedules. Considering all the PO schedules computed in all instances of each scenario, the number of PO schedules in which there is at least one activity which is preempted *more than once* ranges between 3% (in Scenario 5) and 25% (in Scenario 1). This suggests that if a constraint is enforced on the maximum number of preemptions allowed *for each single activity* (as in [57], [60], [78]), for most of the instances derived from J30 and VNR, setting this bound to 1 yields the same optimal solution as that attained through formulation INF_PRMP.

	%Gain 1 preemption	%Gain 2 preemptions	%Gain 3 preemptions	%Gain unlimited preemptions
Scenario 0	2.00	2.58	2.81	3.21
Scenario 1	3.41	5.06	5.98	6.75
Scenario 2	3.53	5.32	6.36	7.15
Scenario 3	2.79	4.29	4.88	5.74
Scenario 4	3.04	4.05	4.82	5.68
Scenario 5	4.02	5.77	6.72	7.82
Scenario 6	4.35	6.80	8.25	9.64
Scenario 7	3.98	6.14	7.46	9.20
Scenario 8	2.81	4.30	5.21	6.76
Scenario 9	3.74	5.34	6.20	7.43
Scenario 10	4.66	7.00	8.31	9.32

Table 4.8: Gains on makespan w.r.t. nonpreemptive schedule.

4.5 A case study: machinery design for the pharma industry

In this section we present the application of the bicriteria model to a real-life project arising in an information technology company.

Pharma Integration is a pharmaceutical machinery manufacturer based in Siena, Italy, specialized in the design of machinery for the fill-finishing process of injectable drugs. The product is highly sophisticated and requires careful customization since the technology is entirely based on robotics. Such technology eliminates human involvement within the process area, so that both operator safety and product quality are guaranteed, minimizing the risk of contamination. This grants the customers, which are pharmaceutical companies integrating the machinery in their production lines, to obtain secure and reliable products suitable for commercial distribution.

The machines produced have a modular design, which offers the advantage of easy customization, allowing customers to select their preferred configuration. Each module encloses groups consisting of robotic arms and cameras, all of them integrated into an

isolator, i.e., an aseptic room in which sensors and cameras drive the robotic arms in the vial filling process.

The project we consider in this experiment concerns the production of a single machinery unit. Each machine produced can be viewed as the customization of a standard product. Correspondingly, the production project follows a standard template, in which some activities depend on the specific customization of the machine being produced. In this study we consider a standard project which includes all the activities which are always present in any actual project. In fact, research and development activities required by design customization take place before production can start, and have a limited impact on the structure of the subsequent project. Hence, the benefits achieved in terms of makespan reduction on the standard project are typically applicable to any customer order. From here on, by *project* we refer to the standard project. Due to the long time span required by the project, the time unit adopted for planning purposes is the working week, i.e., 5 days, and hence this is also the minimum duration of a task. Using the working week as time unit contributes to schedule robustness, preventing small delays or programming errors from having disruptive effects on project duration. Moreover, using the week as time unit is quite common and even recommended in relatively long IT projects, in which, as pointed out by [79], setting too small a time unit would be impractical for planning purposes.

4.5.1 Resources

The project activities require 4 resource types, corresponding to 4 distinct company departments (business units), namely Mechatronics Process Engineering (MPE), Mechatronics Automation Engineering (MAE), Validation & Documentation Engineering (VDE) and Project Management (PM). MPE and MAE develop and install, respectively, mechanical/robotics solutions and software solutions, on the basis of the customer's specifications. They are also in charge of preparing the related documentation and testing the systems implemented. VDE edits and reviews the technical documentation. It also tests the protocols, both internally and at the customer's site, and compiles the instruction manual for the machine. PM is responsible for project execution. This department is in charge of managing and tracking the planning, and it is the main interface with the customer. Starting from the analysis of the technical specifications requested by the customer, PM translates them into input for the company technical departments.

Resources are renewable and one unit corresponds to one company employee of the respective department. Resource availability varies over time, as it is determined by staff commitments to other pre-scheduled projects, vacation days and company closures. The availability of resources for the next 100 weeks is shown in detail in Table 4.9. For each department, the number of available resources varies between 0 and 2.

Most of the project activities require one or more staff units from one department (the activity owner), but some activities require the simultaneous cooperation of different departments. Other activities are outsourced and do not require company resources.

t	n_{MAE}	n_{MPE}	n_{VDE}	n_{PM}
1	0	0	1	1
2	1	0	1	1
3	1	0	1	1
4	1	0	1	1
5	1	1	0	1
6	1	1	0	1
7	1	1	0	1
8	1	1	0	1
9	1	1	2	1
10	0	1	2	1
11	0	1	2	1
12	0	1	2	1
13	1	1	1	1
14	1	2	1	1
15	1	2	1	1
16	1	2	1	1
17	1	2	1	1
18	1	2	1	1
19	1	2	1	1
20	2	2	2	1
21	2	1	2	1
22	2	1	2	1
23	2	1	2	0
24	2	1	2	0
25	2	1	2	0
26	2	1	0	0
27	2	1	0	1
28	2	1	0	1
29	2	1	0	1
30	2	1	0	1
31	1	0	0	1
32	1	0	1	1
33	1	0	1	1
34	1	0	1	1
35	1	0	1	1
36	1	1	1	1
37	1	1	1	1
38	1	1	1	1
39	2	1	1	1
40	2	1	1	1
41	2	1	0	1
42	2	1	0	0
43	2	2	0	0
44	2	2	2	0
45	2	2	2	0
46	1	2	2	1
47	1	2	2	1
48	1	2	2	1
49	1	2	2	1
50	1	2	1	1
51	0	2	1	1
52	0	1	1	1
53	0	1	1	1
54	0	1	1	1
55	0	0	1	1
56	0	1	0	1
57	2	1	1	0
58	2	0	1	0
59	2	0	1	1
60	2	1	1	1
61	2	0	1	1
62	2	0	0	1
63	2	1	1	0
64	2	0	1	0
65	2	0	1	1
66	2	1	1	1
67	0	1	1	1
68	2	0	0	1
69	1	0	0	1
70	1	1	0	1
71	2	1	1	1
72	0	1	1	1
73	1	0	0	1
74	1	1	1	1
75	2	0	0	1
76	1	1	1	1
77	1	1	1	1
78	0	1	1	1
79	1	1	1	1
80	2	1	1	1
81	1	1	1	1
82	1	1	1	1
83	1	1	1	1
84	2	1	1	1
85	2	2	2	0
86	1	2	2	0
87	1	0	2	1
88	1	2	2	1
89	1	2	1	1
90	1	2	1	1
91	1	1	1	1
92	2	1	1	0
93	2	1	1	0
94	2	1	1	0
95	2	1	0	0
96	2	1	1	1
97	2	0	1	1
98	2	1	1	1
99	1	1	1	1
100	1	1	1	1

Table 4.9: Resource departments availability.

4.5.2 Activities

The project instance consists of 80 activities (with activity 0 and 79 being dummy), each having up to three predecessors. Figure 4.5 shows the Activity-on-Node graph of the project. The complete list of activities is reported in Tables 4.10 - 4.11. The planning time horizon that the project is expected not to exceed is $T = 100$ working weeks. This limit is based on previous experience on similar projects.

The project starts with the kickoff meetings (internal and with the customer) to define

i	*Activity*	*d*	a_{MAE}	a_{MPE}	a_{VDE}	a_{PM}	*Succ*	*interr*
0	Start	0	0	0	0	0	1	
1	Kickoff meetings (internal and with the customer)	2	0	0	0	1	2, 6, 10, 14	x
2	DENSO configuration - EDITING	1	1	0	0	1	3	
3	DENSO configuration - REVIEWING	1	0	1	0	0	4	
4	Layout with facility interfaces - EDITING	1	0	1	1	0	5	
5	Layout with facility interfaces - REVIEWING	1	1	0	0	0	8, 12, 16	
6	3D layout - EDITING	1	0	1	0	0	7	
7	3D layout - REVIEWING	1	1	0	1	0	8	
8	Piping and instrumentation diagram - EDITING	2	0	0	1	0	9	x
9	Piping and instrumentation diagram - REVIEWING	1	1	1	0	0	24, 28, 30, 32	
10	Mechanical specification - EDITING	2	0	0	1	0	11	x
11	Mechanical specification - REVIEWING	1	0	1	0	0	18, 20, 22, 24, 28, 30, 32, 44	
12	Functional specification - EDITING	2	0	0	1	0	13	x
13	Functional specification - REVIEWING	1	1	0	0	0	18, 20, 22, 24, 28, 30, 32	
14	Sensors list - EDITING	1	0	0	1	0	15	
15	Sensors list - REVIEWING	1	1	1	0	0	20	
16	HW & SW design specification - EDITING	2	1	0	0	0	17	x
17	HW & SW design specification - REVIEWING	1	0	1	0	0	26	
18	Basement - Procurement	20	0	0	0	0	19	
19	Basement - Acceptance, control, labeling	1	0	1	0	0	56	
20	Main Groups & Robots - Procurement	30	0	0	0	0	21	
21	Main Groups & Robots - Acceptance, control, labeling	2	0	1	1	0	57	x
22	Isolator - Procurement	40	0	0	0	0	23	
23	Isolator - Acceptance, control, labeling	1	0	1	0	0	59	
24	Electrical and pneumatic diagram - EDITING	4	1	0	0	0	25	x
25	Electrical and pneumatic diagram - REVIEWING	1	1	0	0	0	26	
26	I_O list - EDITING	1	1	0	0	0	27	
27	I_O list - REVIEWING	1	1	0	0	0	46	
28	Message matrix - EDITING	1	1	0	0	0	29	
29	Message matrix - REVIEWING	1	0	0	1	0	48	
30	Software interface signal list - EDITING	1	1	0	0	0	31	
31	Software interface signal list - REVIEWING	1	1	0	0	0	36	
32	Process flow chart - EDITING	2	0	0	1	0	33	x
33	Process flow chart - REVIEWING	1	1	0	0	1	34, 46 48	
34	Vision system specification - EDITING	2	1	0	0	0	35	x
35	Vision system specification - REVIEWING	1	1	0	1	0	36	
36	SW detail design specification - EDITING	1	1	0	0	0	37	
37	SW detail design specification - REVIEWING	1	1	0	0	0	38	
38	SCADA specification - EDITING	2	1	0	0	0	39	x
39	SCADA specification - REVIEWING	1	1	0	1	0	40	
40	Process sequence matrix - EDITING	2	1	0	0	0	41	x

Table 4.10: Activities of the standard project (Part I).

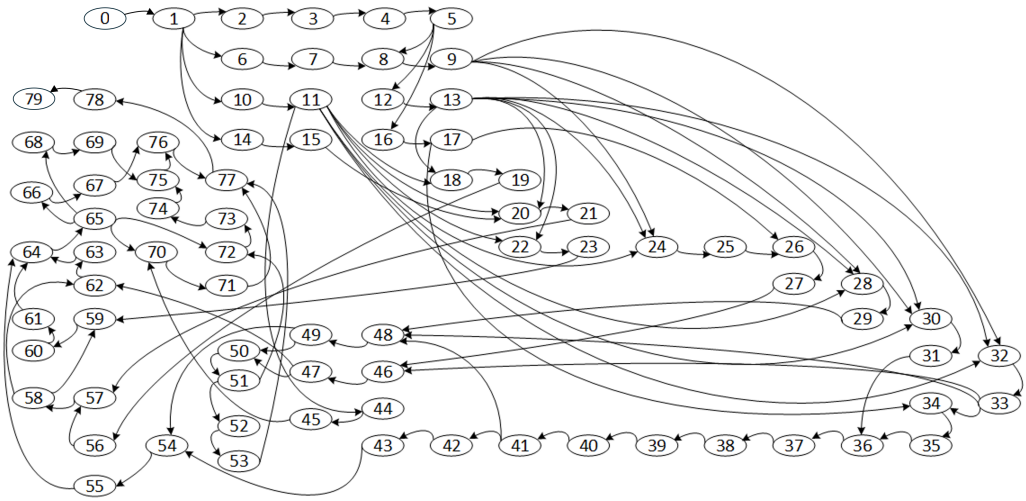

Figure 4.5: Standard project Activity-on-Node diagram.

all the product characteristics and functionalities. In activity 1 of Table 4.10, also the customer documentation review is included.

The pharmaceutical field is strongly characterized by regulations which have to be fulfilled through the release of several documents. These documents describe how the machine operates, its technical specifications, machine layout and customer production plant layouts. In Table 4.10 the corresponding activities are numbered from 2 to 17. There are also more detailed documents such as lists of parameters, signals and spare parts, electrical and process flow diagrams, software specifications (from 24 to 45). Moreover, the machine is delivered with operational manuals including the maintenance manual and the assembly instructions (from 66 to 71). Each activity of documentation editing is followed by a review generally made by a resource from a different department with respect to the editor.

The activities of machinery procurement are the longest activities in the project. However, they do not require internal resources, since component production is committed to external suppliers. Following each procurement activity (activities 18, 20, 22), there are additional tasks requiring company resources (activities 19, 21, 23). When parts arrive in the warehouse, these inspection activities are carried out to ensure that the order received is accurate.

Thereafter, machine parts are assembled (activities 56 and 57) and cabling, electrical and vision system checks are implemented (activities from 58 to 61).

Activities from 46 to 53 concern hardware and software protocols reports editing and reviewing. These reports are filled in upon the completion of internal tests, factory acceptance test (FAT) and site acceptance test (SAT) to track tests' results. As machine operation is simulated in a virtual room, virtual tests (activities 54 and 55) can happen regardless of the arrival of machinery components. Instead, tests of the hardware and its integration with the software (activities 62 through 65) require the assembled machine.

i	*Activity*	*d*	a_{MAE}	a_{MPE}	a_{VDE}	a_{PM}	*Succ*	*interr*
41	Process sequence matrix - REVIEWING	1	1	0	0	0	42, 48	
42	Interface signal list - EDITING	1	1	0	0	0	43	
43	Interface signal list - REVIEWING	1	1	0	0	0	54	
44	Spare parts list - EDITING	1	0	1	0	0	45	
45	Spare parts list - REVIEWING	1	0	0	0	1	70	
46	HW internal test protocol_report - EDITING	2	0	0	1	0	47	x
47	HW internal test protocol_report - REVIEWING	1	0	1	1	0	50, 62	
48	SW integration test protocol_report - EDITING	2	0	0	1	0	49	x
49	SW integration test protocol_report - REVIEWING	1	1	0	1	0	50, 54	
50	FAT protocol_report - EDITING	2	0	0	1	0	51	x
51	FAT protocol_report - REVIEWING	1	1	1	1	0	52	
52	SAT protocol_report - EDITING	2	0	0	1	0	53	x
53	SAT protocol_report - REVIEWING	1	1	1	1	0	77	
54	SW integration test (virtual) - Execution	2	0	0	1	0	55	x
55	SW integration test (virtual) - Review/Approval	1	0	0	1	0	64	
56	Basement predisposition for assembly	1	0	1	0	0	57	
57	Groups assembly in basement	2	0	2	0	0	58	x
58	General line cabling and electrical check	2	1	0	0	0	59, 62	x
59	Isolator integration with basement	4	0	1	0	0	60	x
60	Positioning adjustments and vision system checks	2	1	0	0	0	61	x
61	Preliminary test run	2	1	1	0	0	64	x
62	HW internal test - Execution	2	0	0	1	0	63	x
63	HW internal test - Review/Approval	1	0	0	1	0	64	
64	HW & SW integration test - Execution	2	0	0	1	0	65	x
65	HW & SW integration test - Review/Approval	1	0	0	1	0	66, 68, 70, 72	
66	Operating manual - EDITING	2	0	0	1	0	67	x
67	Operating manual - REVIEWING	1	1	1	1	0	76	
68	Transport and assembly instructions - EDITING	2	0	0	1	0	69	x
69	Transport and assembly instructions - REVIEWING	1	0	1	1	1	75	
70	Maintenance manual - EDITING	2	0	0	1	0	71	x
71	Maintenance manual - REVIEWING	1	1	1	1	0	77	
72	Meeting with customer and FAT execution	1	0	1	1	1	73	
73	Crating operations	1	0	1	0	0	74	
74	Shipment to customer	3	0	0	0	0	75	
75	Supervision for move-in and connection to media	1	0	1	0	0	76	
76	Start-up/Commissioning	3	0	0	1	0	77	
77	Meeting with customer and SAT execution	1	0	1	1	0	78	
78	Project final meeting	1	0	0	0	1	79	
79	End	0	0	0	0	0		

Table 4.11: Activities of the standard project (Part II).

FAT (activity 72) helps verify that the equipment fulfills its intended purpose and ensures that customer purchase order specifications have been met. This activity takes place on Pharma Integration's premises at the presence of the customer.

After the execution of FAT, the machine is packed and shipped to the customer plant (activities 73 and 74). When the machine is delivered, it is installed and started at the customer plant (activities 75 and 76). Then, SAT is executed (activity 77) to confirm that the performance resulting from the FAT is repeated after the systems are installed on site,

ensuring that nothing has been damaged during shipment and installation.

The project ends with activity 78 which consists in a closing meeting to evaluate the results and ideas for possible future improvement.

4.5.3 Application of the model

In order to apply the model to the project described in this section, it has to be established whether each activity can or cannot be interrupted.

Activities 18, 20, 22 (concerning components procurement) and 74 (concerning machinery shipment) are outsourced to external companies, so there is no point in preempting them, even without explicitly imposing a non-preemption constraint (4.10).

The activity of machinery start up (activity 76) cannot be interrupted because it is carried out at the customer's site and consequently involves a business trip of company staff, so we add to the ILP model a non-preemption constraint (4.10). Interruptible activities (i.e., activities of duration larger than 1 which are technically feasible to preempt) are marked in the last column of Tables 4.10-4.11.

We note that activities 76 and 77 are both carried out by the same resource (VDE staff) at the customer's site, so it is desired that they be carried out in the minimum time. For this reason, we impose that 76 cannot be interrupted and that 77 starts as soon as 76 is finished. This could have been obtained through the no-wait constraints

$$x_{j,t+1} \geq z_{it}$$

with $i = 76$ and $j = 77$, for each t, which suffice since activity i cannot be interrupted. However, even omitting these constraints, the no-wait requirement turned out to be fulfilled in all Pareto optimal solutions. A generic formulation of the no-wait condition between the end of i and the start of j, which also applies if activity i is allowed to be interrupted, would be

$$x_{j,t+1} \geq z_{it} + (1/d_i) \sum_{\tau=1}^{t} x_{i\tau} - (2 - 1/d_i) \quad t = 1, \ldots, T-1$$

Table 4.12 shows that in this instance, the Pareto front consists of four solutions, corresponding to 0, 1, 2 and 3 preemptions respectively. Despite the relatively large number of activities, the model has been solved in few seconds of computation. We note here that some specific features of the case study (short tasks, long paths on the activity graph) may contribute to the efficient solution of the ILPs.

# of preemptions	Makespan (weeks)	Gain on makespan w.r.t. sol with 0 preemptions (weeks)	Runtime (s)
0	100	-	0.76
1	96	4	2.74
2	90	10	7.23
3	89	11	4.17

Table 4.12: Pareto front of case study solutions.

The results show that without preemptions the project makespan is 100 weeks, and that the possibility to interrupt activities presents a significant opportunity to decrease the overall project duration. The largest *marginal* gain (6 weeks) is achieved by the second preemption. Allowing three preemptions achieves a total gain of 11 weeks on makespan, i.e., 11%. Figure 4.6 shows the Gantt chart of the solution with 3 preemptions. The preempted activities are 59 (twice) and 64. They are interrupted due to variations in resource availability.

Finally, we observe that many project activities have unit duration, and are therefore treated as non-interruptible by the model. Even if, adopting a shorter time unit (e.g., day), the duration of some activities might be expressed in greater detail, and hence further gains might be obtained, the resulting schedule would be inherently less robust, especially on a long time span.

4.6 Conclusions

In this chapter we analyzed the tradeoff between makespan and number of preemptions in resource-constrained project scheduling problem under variable resource availability. Using a time-indexed ILP formulation, we derived the Pareto set for a large number of medium-size instances from two benchmark sets, and a real-life instance from an IT development project.

The experiments showed that a limited overall number of preemptions can typically achieve most of the gain attained with an unrestricted number of preemptions, and that marginal gains on makespan due to preemptions are typically decreasing, though there are several exceptions (such as the case study). On the whole, the gains are higher if resource variation intensity is larger while variation frequency plays a minor role, and most often the first preemption accounts for nearly 50% of the gain w.r.t. unlimited preemptions. In view of the fact that in several benchmark instances even extreme Pareto optimal solutions display a small number of preemptions, we observe that the schedules obtained setting a limit on the maximum number of preemptions for each activity typically have the same makespan of the solution with an unlimited number of preemptions. We noticed that as the intensity of variation increases, the size of the Pareto front also increases, i.e. with scarcer resources, more interruptions are necessary to achieve the maximum benefit obtainable on the makespan. This behavior is also reflected in the RS indicator: lower values of RS correspond to conditions in which it is more convenient to interrupt, there are more Pareto optimal solutions and the gain is higher.

Further research may address several issues, including:

- New formulations for RCPSP with limited number of preemptions. We performed some tentative experiments with an event-based approach, for the baseline scenario, obtained by modifying the formulation by [75]. Indeed, out of all the J30 benchmark instances, only in few instances such formulation returned better results (in terms of computation time) than MIN_PRMP. These instances typically have long activities. This is why we ran our experiments using formulations MIN_PRMP, NON_PRMP and INF_PRMP. The extension of the event-based concept to scenarios with variable

resource availability is not obvious, and it deserves further analysis.

- Larger-size instances may require different solution approaches. These may include computationally efficient heuristic or metaheuristic approaches, as well as constraint programming approaches.
- The bicriteria problem can be investigated in more general RCPSP settings, such as multi-mode, multi-skill, presence of setups or constraints on the time instants in which preemption can occur. Different models are needed to deal with such more general cases.
- Another direction for future research is to focus on preemption modeling. While in the RCPSP literature the usual assumption is that preemptions have no impact on the overall duration of an activity, some authors (e.g., [80]) do consider a forgetting effect (during the preemption period) which may lead to an increased length of the remaining part of the activity. Note that other issues, such as preemption-related costs, could easily be taken into account by our model (simply multiplying z_{it} by a cost factor depending on i in the objective function (4.1)).
- On the theoretical side, bounds can be investigated concerning the maximum relative gain that can be attained through a given number of preemptions. We note that, as shown by [67], $n(n-1)/2$ preemptions are always sufficient to attain the same minimum makespan value when an unlimited number of preemptions is allowed.
- Finally, this study suggested that despite the growing literature on preemptive RCPSP, a widely accepted reference framework for such problems is still missing. In fact, in real-life situations, various preemptive settings may occur, for example in which a limit exists on the maximum number of preemptions for each activity, on the total number of preemptions, on the variation patterns for resource capacities etc. Solution approaches tailored for various configurations of the above settings (possibly motivated by specific application contexts) might be the object of future research.

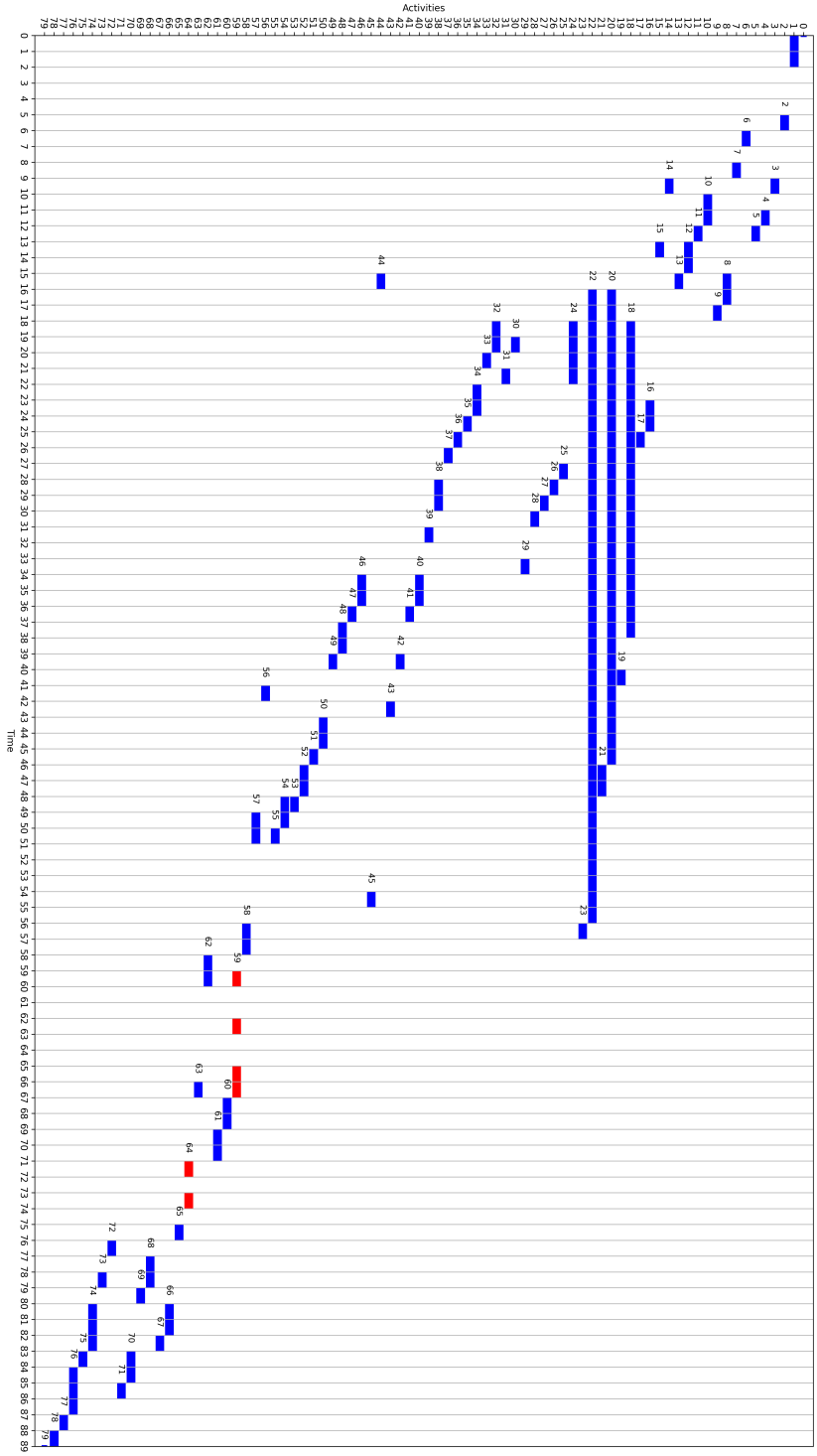

Figure 4.6: Gantt diagram of the solution with three preemptions.

Part III

Scheduling jobs on machines subject to unrecoverable breakdowns

Introduction to scheduling problems subject to unrecoverable breakdowns

This part of the thesis addresses scheduling problems that arise in the context of machines subject to failures. Scheduling problems on machines subject to unrecoverable breakdowns represent an important class of stochastic scheduling models where machines may fail permanently during job execution. In several models, failures imply the temporary unavailability of the machine and they occur at unpredicted times, though a probabilistic description of the failure process is usually available [81]. These problems arise naturally in various practical contexts where equipment failure, resource depletion, or catastrophic events can terminate operations irreversibly. Unlike traditional scheduling models that assume deterministic processing environments or recoverable failures, unrecoverable breakdown models capture the inherent uncertainty and irreversibility present in many real-world applications.

In many models, after the machine recovers from failure, the job that was being processed can either continue where it left off (preempt-resume models), or it has to be restarted from scratch (preempt-repeat models) [82], [83]. In these cases, the problem is to optimize a classical scheduling-related objective, such as the weighted completion time or some tardiness-related measure, etc. In this part, we address a different situation, namely when machines are subject to *unrecoverable interruptions* [84]. By "interruption", we mean that, while processing a job, a processing resource may become unavailable for whatever reason (e.g., a real machine failure or the machine being preempted by some higher-priority user). In any case, when an interruption occurs, the current job and all subsequently scheduled jobs on the machine are lost. In such contexts, we consider that each job has a certain value if successfully carried out, and the problem becomes that of deciding the schedule of the jobs so as to maximize the expected amount of this value. Clearly, such a decision must take into account the specificity of the failure process of the machine(s).

Scheduling with unrecoverable breakdowns applies to settings where failures during sequential operations permanently halt all subsequent activities. For example, in the case of project scheduling, uncontrollable events may force the project to close down [85], or in diagnostic problems, the tests are stopped when the results of the previous

tests show that the overall system is not properly working [86]. Within the scheduling field, unrecoverable interruptions are considered in cloud computing, and strategies are enacted to hedge against host permanent failures [87]. [88] analyzed the problem of sharing the workload among various computers subject to unrecoverable interruptions, using strategies such as job replication or checkpointing. Enlarging the view beyond the scheduling domain, there are other situations in which unrecoverable events may bring activities to a halt. For example, in diagnostic testing, physicians must sequence tests to maximize information while accounting for adverse events that could terminate the examination. In manufacturing contexts, high-defect jobs increase equipment breakdown probability, necessitating careful ordering when repairs are prohibitively expensive or infeasible.

In the following, we examine two distinct models of breakdown behavior. Chapter 5 considers settings where failure probability depends on job characteristics, that is, different jobs impose different breakdown risks on the machine. Chapter 6 analyzes settings where the machine's breakdown process follows a linear risk function, with failure probability increasing over time regardless of the job being processed.

Chapter 5

The Unreliable Job Selection and Sequencing Problem

In this chapter we study a stochastic single-machine scheduling problem, denoted the Unreliable Job Selection and Sequencing Problem (UJSSP). Given a set of jobs, a subset must be selected for processing on a single machine that is subject to failure. Each job incurs a cost if selected and yields a reward upon successful completion. A job is completed successfully only if the machine does not fail before or during its execution, with job-specific probabilities of success. The objective is to determine an optimal subset and sequence of jobs to maximize the expected net profit.

In Section 5.1, we provide an overview of related work and discuss practical applications of the problem. In Section 5.2, we establish the complexity of UJSSP, showing that the problem is NP-hard in the general case. In Section 5.3 we present a compact mixed-integer linear programming (MILP) formulation of UJSSP, and in Section 5.4 we provide an upper bound for the optimal value of UJSSP which can be efficiently computed. In Section 5.5 some conclusions are drawn.

We are given a set $J = \{1, \dots, n\}$ of jobs which can be performed on a single machine. Each job has a cost $c_j \geq 0$ which has to be paid if the job is selected, and a reward $r_j \geq 0$ which is achieved if the job is successfully carried out. The machine can fail during the execution of a job. When this happens the job is lost and no further job can be carried out. The probability that the machine fails during the execution of job j is given by $1 - \pi_j$, where $\pi_j \in [0, 1]$ is called the *success probability*. If a certain subset S of jobs is selected for processing, let σ denote a sequence of these $|S|$ jobs, and let $\sigma(k)$ represent the k-th job in the sequence σ. The probability of reaching and successfully carrying out the k-th job in the sequence is given by

$$\prod_{i=1}^{k} \pi_{\sigma(i)},$$

therefore the expected reward $R(S, \sigma)$ from selecting S and sequencing its jobs according to σ is

$$R(S, \sigma) = \sum_{k=1}^{|S|} r_{\sigma(k)} \prod_{i=1}^{k} \pi_{\sigma(i)}. \tag{5.1}$$

If we let $c(S) = \sum_{j \in S} c_j$ denote the total cost of selecting S, the *expected net profit* of selecting S and sequencing the jobs according to σ is

$$\tilde{z}(S,\sigma) = \sum_{k=1}^{|S|} r_{\sigma(k)} \prod_{i=1}^{k} \pi_{\sigma(i)} - c(S). \tag{5.2}$$

We study the problem of selecting a subset $S \subseteq J$ of jobs and finding a sequence σ so that the expected net profit $\tilde{z}(S,\sigma)$ is maximized. We call this problem the *Unreliable Job Selection and Sequencing Problem* (UJSSP).

When there are no costs ($c_j = 0$ for all $j = 1,\ldots,n$), it is obviously profitable to select all jobs, and only the sequencing problem is left. In this case, we refer to the problem as the *Unreliable Job Sequencing Problem* (UJSP). This problem has been independently considered by various authors under different names and application contexts; see Kadane [89], Stadje [90], Hellerstein and Stonebraker [91], and Agnetis et al. [92]. UJSP is also equivalent to the basic n-out-of-n test sequencing problem, from which a considerable literature on testing problems originated (see for instance [93] or [94]). When there are no costs, the optimal sequence is obtained by simply scheduling the jobs by non-increasing values of the following index ([89], [93]):

$$Z_j = \frac{\pi_j r_j}{1-\pi_j}. \tag{5.3}$$

Consequently, in UJSSP, once a subset S is selected, expression (5.2) is maximized by sequencing the jobs in S according to a schedule dictated by (5.3). Letting σ_Z denote such sequence, in the following we let $R(S) = R(S,\sigma_Z)$ and $z(S) = \tilde{z}(S,\sigma_Z)$ we can restate UJSSP as the problem of finding S^* such that:

$$z(S^*) = \max_{S\subseteq J} \{R(S) - c(S)\}. \tag{5.4}$$

For the sake of simplicity, unless specified otherwise, we assume from now on that the n jobs are numbered by non-increasing values of Z_j, and we rewrite $R(S)$ as

$$R(S) = \sum_{k\in S} r_k \prod_{i\le k, i\in S} \pi_i. \tag{5.5}$$

5.1 Related work and applications

Research on scheduling under machine breakdowns has a long history in the scheduling literature. [95] proposed a fairly general model in which the scheduling process is viewed as a Markov decision process, with both job and machine breakdown durations being stochastic, and the aim is to define policies to decide which jobs should be carried forward at any time. In this as well as other subsequent models, breakdowns are assumed to last for a certain (possibly random) duration, after which the machine resumes operation, either restarting the current job from scratch or from the point it was when preemption occurred [96]. In these cases, models are considered in which classical scheduling objectives are pursued, such as total (weighted) flow time or total tardiness. The focus is on devising optimal strategies even in the presence of disruptions, such as showing that the SPT rule is still optimal when minimizing total flow time in various post-disruption management

situations ([82], [97]), or that WSPT is not, when minimizing total weighted flow time in the face of stochastic breakdowns [98].

While Kadane [89] had addressed a very similar problem in the context of a quiz game, one of the first papers to deal with unrecoverable machine breakdowns was the one by Stadje [90], which is very relevant also for our problem. In his model, a given number of jobs needs to be selected out of a set of jobs for processing on a single machine, each achieving a certain (discounted) reward if successfully completed. The machine breaks down with a certain job-specific probability and cannot be restored. Stadje found out that the problem of maximizing the expected reward is solved by the greedy approach, and observed that it does not work in the more general case of jobs having different costs. The problem without costs (UJSP) has been generalized to a context with parallel machines (Agnetis et al., [92]), and when job replication is allowed on multiple machines to hedge against breakdowns (Agnetis et al., [99]). All these generalizations result in NP-hard problems, even for $m = 2$ machines.

UJSSP can be shown to be a special case of a problem called *Simultaneous Selection Problem* (SSP), introduced by Olszewski and Vohra [100]. To this aim, we first need to briefly recall some basic concepts of submodular optimization. Consider a set T of items, each having utility u_i, such that $u_1 \geq u_2 \geq \cdots \geq u_n$, and a submodular non-decreasing set function $f(\cdot)$. Then consider the following *submodular optimization problem* (SOP):

$$g(T) = \max \sum_{i \in T} u_i x_i \tag{5.6a}$$

$$\sum_{i \in S} x_i \leq f(S) \;\; \forall S \subseteq T \tag{5.6b}$$

$$x_i \geq 0, i \in T$$

SOP can be interpreted as the maximum utility that one can obtain by assigning weights x_i to the elements of T, given that for any subset S of T the sum of the weights does not exceed the submodular function $f(S)$. As $f(\cdot)$ is submodular, (5.6b) is indeed a polymatroid and it is well known that $g(T)$ has the following optimal solution:

$$\begin{aligned} x_1^* &= f(\{1\}) \\ x_2^* &= f(\{1,2\}) - f(\{1\}) \\ x_3^* &= f(\{1,2,3\}) - f(\{1,2\}) \\ &\dots \\ x_n^* &= f(\{1,2,3,\dots,n\}) - f(\{1,2,3,\dots,n-1\}). \end{aligned} \tag{5.7}$$

It is easy to check that UJSP (i.e., UJSSP without costs) is a special case of SOP. In fact, given an instance of UJSP, in (5.6a) let $u_j = Z_j$ and $f(S) = 1 - \prod_{j \in S} \pi_j$. From (5.7), we get that the optimal solution is:

$$\begin{aligned} x_1^* &= 1 - \pi_1 \\ x_2^* &= (1 - \pi_1\pi_2) - (1 - \pi_1) = \pi_1(1 - \pi_2) \end{aligned} \tag{5.8}$$

$$
\begin{aligned}
x_3^* &= (1-\pi_1\pi_2\pi_3) - (1-\pi_1\pi_2) = \pi_1\pi_2(1-\pi_3) \\
&\dots \\
x_n^* &= (1-\pi_1\pi_2\pi_3\dots\pi_n) - (1-\pi_1\pi_2) = \pi_1\pi_2\dots\pi_{n-1}(1-\pi_n)
\end{aligned}
$$

and hence (5.6a) becomes

$$
\begin{aligned}
g(T) = \sum_{i\in T} u_i x_i^* &= \frac{\pi_1 r_1}{1-\pi_1}(1-\pi_1) \\
&+ \frac{\pi_2 r_2}{1-\pi_2}\pi_1(1-\pi_2) \\
&+ \frac{\pi_3 r_3}{1-\pi_3}\pi_1\pi_2(1-\pi_3) \\
&+ \dots \\
&+ \frac{\pi_n r_n}{1-\pi_n}\pi_1\pi_2\dots\pi_{n-1}(1-\pi_n)
\end{aligned} \tag{5.9}
$$

which is identical to (5.1) (for $|S| = n$).

Let us now add the costs to the picture. Given a ground set N, and a cost c_j for each variable $j = 1, \dots, |N|$, for each $T \subseteq N$, let $z(T) = g(T) - c(T)$, where $g(T)$ is defined by (5.6a) and $c(T) = \sum_{j\in T} c_j$. The *Simultaneous Selection Problem* (SSP) [100] consists in selecting a subset $T^* \subseteq N$ such that

$$
T^* = \arg\max_{T\subseteq N} \{z(T)\}\,. \tag{5.10}
$$

From what above, it is apparent that UJSSP is indeed a special case of SSP.

A *feasible* solution to SSP can be found through the *greedy algorithm* (see Algorithm 1). Starting from an empty set, the algorithm iteratively adds the item that maximizes the increase of the objective function of SSP, as long as there is at least one such item. The solution obtained is not optimal in general, but it can be if $f(\cdot)$ fulfills specific properties [100].

Algorithm 1 Greedy algorithm for SSP.

Order the items by nonincreasing utilities u_i; $S_0 \leftarrow \{\emptyset\}$; $i \leftarrow 0$; **while** *($i < n$ and) there is at least one item $j \notin S_i$ such that $z(S_i \cup \{j\}) > z(S_i)$* **do**-_

Select the job $k \notin S_i$ such that

$$
z(S_i \cup \{k\}) = \max_{j\notin S_i}\{z(S_i \cup \{j\})\};
$$

$S_{i+1} \leftarrow S_i \cup \{k\}$ $i \leftarrow i+1$ **return** S_i.

Other problems have been shown to be special cases of SSP, including the following *college selection problem* (CSP) introduced by Chade and Smith [101]. In this problem a student must select a subset S from a set N of colleges ($|N| = n'$) to apply to. For each college j there is a utility w_j, a probability α_j of being accepted, and an application cost c'_j. The student applies to a number of colleges, and will pick one having the largest

utility among those that admit her. If we number the colleges by non-increasing utilities, the probability that college j is the one the student will actually go to when S is selected, is given by

$$\alpha_j \prod_{i<j, i\in S} (1-\alpha_i),$$

and CSP consists in selecting a subset S of colleges so as to maximize the expected utility, i.e., letting $c'(S) = \sum_{j\in S} c'_j$:

$$\max_{S\subseteq N} \left\{ \sum_{j\in S} w_j \alpha_j \prod_{i<j, i\in S} (1-\alpha_i) - c'(S) \right\}. \tag{5.11}$$

It was shown by Chade and Smith [101] that if all costs c'_j are identical, the greedy algorithm solves the problem.

Now, UJSSP and CSP can be transformed into one another. In fact, given an instance of CSP, consider an instance of UJSSP with $n = n'$ jobs, letting $r_j = w_j\alpha_j/(1-\alpha_j)$, $\pi_j = (1-\alpha_j)$ and $c_j = c'_j$. In this case (5.2) and (5.11) coincide, and

$$Z_j = \frac{\pi_j r_j}{1-\pi_j} = w_j,$$

so indeed the ordering based on (5.3) of UJSSP coincides with the ordering by non-increasing utilities in CSP. Vice versa, one can transform an instance of UJSSP into an equivalent instance of CSP by letting $w_j = r_j\pi_j/(1-\pi_j)$, $\alpha_j = 1-\pi_j$, and $c'_j = c_j$. Hence, the general complexity results we derive for UJSSP will also apply to CSP.

5.2 Complexity

Before discussing the complexity of UJSSP in the general case, we rule out two special cases.

5.2.1 Identical costs

Consider the special case of UJSSP in which all jobs have the same cost, i.e., $c_j = c$ for all $j = 1, \ldots, n$. A result by Stadje [90] implies that in this case UJSSP can be solved by the greedy algorithm. Indeed, in view of the aforementioned equivalence between UJSSP and CSP, we note that the same result was independently found by Chade and Smith [101], as they showed that when, in CSP, the application cost is the same for all colleges, the greedy strategy is optimal. We can therefore conclude that if $c_j = c$ for all j, the problem can be solved in $O(n^2)$, as at each step we simply need to "try out" each of the remaining jobs and select one yielding the largest marginal expected gain. (Notice that examples can be provided [101] in which the optimal set S^* does not contain all the *first* $|S^*|$ jobs when ordered by nonincreasing Z_j.)

5.2.2 Identical probabilities

In this section we consider the special case of UJSSP in which $\pi_j = \pi$ for all $j = 1, \dots, n$. First of all, notice that in this case, letting $r_{[k]}$ denote the reward of the job in position k, and denoting with S the set of selected jobs, the expected reward can be written as:

$$R(S) = \sum_{k=1}^{|S|} r_{[k]} \pi^k. \tag{5.12}$$

As a consequence, if a job j is assigned to position k, its contribution to the net expected reward is given by

$$\max\{0, r_j \pi^k - c_j\}, \tag{5.13}$$

and for a fixed number H of selected jobs, the optimal solution can be found by solving an assignment problem in which the reward q_{jk} from assigning j to position k is given by (5.13) if $k \leq H$, and 0 otherwise. Iterating for all values of H, we find the optimal solution of UJSSP.

However, a stronger result can be established viewing UJSSP as a special case of SSP. To this aim, we need one last notion. A submodular function is *downward recursive* if, for any two disjoint subsets U and V such that $\min_{j \in U}\{u_j\} \geq \max_{j \in V}\{u_j\}$, it holds

$$f(U \cup V) = f(U) + f(V)\rho(U),$$

where $\rho(\cdot)$ is a multiplicative function (i.e., $\rho(U \cup V) = \rho(U)\rho(V)$) whose values belong to $[0, 1)$.

The main result given in [100] is the following.

Theorem 1. *Olszewski and Vohra, 2016 [100]. Given an instance of SSP, if:*

(i) $f(S)$ is downward recursive, and

(ii) $f(S)$ only depends on the cardinality of S

then SSP is optimally solved by the greedy algorithm.

We next use this result to prove that UJSSP with identical probabilities can be solved by the greedy algorithm.

Theorem 2. *If all the jobs have the same success probability π, UJSSP is solved by the greedy algorithm.*

Proof. We only need to show that the two conditions (i) and (ii) of Theorem 1 hold.

(i) From (5.6a)-(5.6b), UJSSP can be formulated as SSP, with $f(S) = 1 - \prod_{j \in S} \pi_j$. Consider the following function $\rho(S) = \prod_{j \in S} \pi_j$. Clearly, $\rho(S)$ is a multiplicative function, and given disjoint sets U and V:

$$f(U \cup V) = 1 - \prod_{i \in U \cup V} \pi_i = (1 - \prod_{i \in U} \pi_i) + (\prod_{i \in U} \pi_i)(1 - \prod_{i \in V} \pi_i) = f(U) + \rho(U) f(V),$$

hence f is downward recursive.

(ii) In the special case of UJSSP in which all the jobs have the same success probability π, for any subset S one has

$$f(S) = 1 - \pi^{|S|},$$

hence $f(S)$ only depends on the cardinality of S. □

5.2.3 The general case

SSP is NP-hard for a general submodular function f, even when $f(S)$ is of coverage-type (Feige and Vondrák [102]). However, we are not aware of results establishing the complexity of UJSSP. In the following, we address this issue by showing that UJSSP is in general NP-hard.

UJSSP*. Given a set J of n jobs, and for each job j a reward $r_j \in \mathbb{R}$, a success probability $\pi_j \in [0,1]$ and a cost $c_j \in \mathbb{R}$, and a value $X \in \mathbb{R}$, is there a subset $S \subseteq J$ of jobs and a sequence σ so that the expected net profit $\tilde{z}(S,\sigma) \geq X$?

In the proof we use the following problem, which is strongly NP-hard [103].

PRODUCT PARTITION. Given a set N of n positive integers $a_1, \ldots, a_n$, is there a subset $S \subset N$ such that

$$\prod_{j \in S} a_j = \prod_{j \in N \setminus S} a_j?$$

Note that in the instance of PRODUCT PARTITION we can assume $a_j \geq 2$ for all j.

Theorem 3. *UJSSP* is NP-hard.*

Proof. Given an instance of PRODUCT PARTITION, we define an instance of UJSSP* as follows, where we let $W = \prod_{j=1}^{n} a_j$. There are n jobs, such that:

$$\pi_j := \frac{1}{a_j} \tag{5.14}$$

$$r_j := \sqrt{W}\left(\frac{1-\pi_j}{\pi_j}\right) = \sqrt{W}(a_j - 1) \tag{5.15}$$

$$c_j := \ln a_j. \tag{5.16}$$

$$X := \sqrt{W} - 1 - \ln \sqrt{W}. \tag{5.17}$$

We want to establish whether there is a subset of jobs that ensures a net expected reward of at least X.

From (5.14)–(5.16), we observe that for any job j, one has $Z_j = \pi_j r_j/(1-\pi_j) = \sqrt{W}$, so the value of Z_j is the same for all jobs. Hence, given any subset S, the value of $R(S,\sigma)$ is the same for any σ, given by:

$$\pi_{\sigma(1)} r_{\sigma(1)} + \pi_{\sigma(1)}\pi_{\sigma(2)} r_{\sigma(2)} + \cdots + \pi_{\sigma(1)}\pi_{\sigma(2)} \cdots \pi_{\sigma(|S|)} r_{\sigma(|S|)}$$

which, from (5.15) and after some algebra, yields (assume that $\prod_{j \in S} \pi_j = 1$ if $S = \emptyset$):

$$R(S) = \sqrt{W}(1 - \prod_{j \in S} \pi_j).$$

As a consequence, we can write the net expected profit as

$$z(S) = \sqrt{W}(1 - \prod_{j \in S} \pi_j) - \sum_{j \in S} c_j. \tag{5.18}$$

So, from (5.17) and (5.18) we want to establish whether there is a subset S of jobs such that

$$\sum_{j\in S} c_j + \sqrt{W}\prod_{j\in S}\pi_j \leq 1 + \ln\sqrt{W}. \tag{5.19}$$

For any S, from (5.14) and (5.16) we can write the left-hand-side of (5.19) as

$$\ln\prod_{j\in S} a_j + \sqrt{W}\frac{1}{\prod_{j\in S} a_j}. \tag{5.20}$$

Now, the function

$$f(x) = \ln x + \sqrt{W}\frac{1}{x}$$

has a minimum for $x = \sqrt{W}$, and it is given by $1 + \ln\sqrt{W}$. Hence, (5.19) holds (at equality) if and only if there is a partition of the n integers into two sets, each having product equal to $\sqrt{W}$.

We are left with showing that the reduction is polynomial. Denoting by $a_{\max}$ the largest integer appearing in the instance of PRODUCT PARTITION, the input size of such an instance is $O(n\log_2 a_{\max})$. The only issue is that the values of rewards and costs in the instance of UJSSP can be irrationals. Of course, precise encoding of an irrational would require an infinite number of bits, but we are only concerned with a precision level which allows distinguishing two different values of $\ln\prod_{j\in S} a_j$ for any S. As $\prod_{j\in S} a_j \leq W$, we only need enough bits to distinguish between $\ln W$ and $\ln(W+1)$. Since $\ln(W+1) - \ln W \geq \frac{1}{W} - \frac{1}{W^2} > \frac{1}{W^2}$ (for any $W \geq 3$), we need at most $2\lceil\log_2 W\rceil$ bits (corresponding to decimal digits) to distinguish between $\ln W$ and $\ln(W+1)$. On the other hand, as the smallest probability will be larger than $1/W$, each probability requires at most $\log_2 W = O(n\log_2 a_{\max})$ bits each to be encoded, and the rewards at most $\log_2 a_{\max} + 1/2\log_2 W = O(n\log_2 a_{\max})$ bits, as rewards are integer and each reward is smaller than $\sqrt{W}a_{max}$. Finally, each cost can be encoded by means of $\log_2(\ln a_{max}) = O(\log_2 a_{max})$ bits for the integer part and, as recalled, $2\lceil\log_2 W\rceil$ bits for the decimal part. Hence, as there are n jobs, the total number of bits required to encode the instance of UJSSP is $O(n^2\log_2 a_{\max})$, which is polynomial in the input size of the instance of PRODUCT PARTITION. □

Notice that even if PRODUCT PARTITION is strongly NP-hard, we only prove the ordinary NP-hardness of UJSSP, since in our proof we use the value $\sqrt{W}$, where W is the product of all numbers of PRODUCT PARTITION, and such a product is not bounded by a polynomial in the maximum integer of PRODUCT PARTITION. In fact, the input of an instance of PRODUCT PARTITION does not include the number $\sqrt{W}$.

We note here that the result in Theorem 3 implies that the Simultaneous Selection Problem [100] is NP-complete even when the submodular function $f(S)$ is downward recursive.

As the cases with identical costs or identical probabilities are solved by the greedy algorithm, one may be led to investigate if this is the case also when all rewards are identical. It turns out that the greedy algorithm may not provide the optimal solution in this case.

Example 1. *Consider $n = 3$ and the data in Table 5.1. In this example, all jobs have unit reward (hence the optimal sequence is by decreasing probabilities). At the first step, the greedy algorithm would select job 2, since $z(2) = \pi_2 - c_2 = 0.1$ while $z(1) = \pi_1 - c_1 = 0.09$ and $z(3) = \pi_3 - c_3 = 0.09$. At the second step, the greedy algorithm adds job 1, as $z(1,2) = \pi_1 + \pi_1\pi_2 - c_1 - c_2 = 0.103 > z(2,3) = \pi_2 + \pi_2\pi_3 - c_2 - c_3 = 0.1029$, and we see that $\{1,2\}$ is indeed better than $\{2\}$. Subsequently adding also job 3 yields $z(1,2,3) = \pi_1 + \pi_1\pi_2 + \pi_1\pi_2\pi_3 - c_1 - c_2 - c_3 = 0.0476 < 0.103$, so the greedy algorithm returns the set $\{1,2\}$. However, the optimal solution is $\{1,3\}$, as $z(1,3) = \pi_1 + \pi_1\pi_3 - c_1 - c_3 = 0.113$.*

The complexity of UJSSP in the special case of identical rewards is open. Also, we can give some partial result for the special case in which the product $\pi_j r_j$ is identical for all jobs j. In fact, the following lemma holds.

j	π_j	c_j	r_j
1	0.9	0.81	1
2	0.87	0.77	1
3	0.67	0.58	1

Table 5.1: Data for Example 1.

Lemma 1. *Let J be a set of n jobs such that for all $j \in J$: $r_j\pi_j = k$, for some $k > 0$. A subset $S = \{j_1, \ldots, j_{|S|}\} \subseteq J$, can only be optimal if:*

1. *The job with the lowest cost is included in S, and*
2. *For every $i \in \{1, \ldots, |S| - 1\}$, the job $j = \arg\min_{j > j_i} c_j$ is also included in S.*

Proof. The expected profit of executing S in order is:

$$z(S) = \sum_{i=1}^{|S|} \left(k \cdot \prod_{l=1}^{i-1} \pi_{j_l} \right) - \sum_{i=1}^{|S|} c_{j_i}.$$

Suppose $j_{\min} = \arg\min_{j \in J} c_j$ is not in S. Let $S' = (S \setminus \{j_{|S|}\}) \cup \{j_{\min}\}$. Then:

$$z(S') \geq z(S) + (c_{j_{|S|}} - c_{j_{\min}}) > z(S),$$

since replacing the last job in the sequence by $j_{\min}$ and keeping the order the same does not influence the revenue component and increases the expected profit with $c_{j_{|S|}} - c_{j_{\min}} > 0$, which contradicts optimality of S. Similarly, for any $i \in \{1, \ldots, |S| - 1\}$, let $j = \arg\min_{j > j_i} c_j$. If $j \notin S$, define S' by replacing the last element of S with j. The profit again increases due to the cost reduction, contradicting optimality of S. □

This lemma implies that in this special case, the greedy algorithm is optimal if $|S^*| = 2$ or if $n \leq 3$. However, for $n = 4$ we can already find an example for which the greedy algorithm does not find the optimal solution.

Example 2. *Consider $n = 4$ and the data in Table 5.2. In this example, all jobs have a reward times probability which equals 80 (hence the optimal sequence is by decreasing*

probabilities). At the first step, the greedy algorithm would select job 3 since it has the lowest cost: $z(3) = 80 - 8 = 72$. *At the second step, the greedy algorithm adds job 2, as* $z(2,3) = 80 + 0.4 \cdot 80 - 24 - 8 = 80 > z(1,3) = 80 + 0.8 \cdot 80 - 57 - 8 = 79 = z(3,4) = 80 + 0.2 \cdot 80 - 8 - 9 = 79 > z(3)$. *In a third step, the greedy algorithm adds job 1 because* $z(1,2,3) = 80+0.8\cdot80+0.8\cdot0.4\cdot80-57-24-8 = 80.6 > z(2,3) > z(2,3,4) = 80+0.4\cdot80+0.4\cdot0.2\cdot80-57-8-9 = 77.4$. *The greedy algorithm stops here as adding job 4 to* $\{1,2,3\}$ *does not lead to an improvement. However, in the optimal solution we select* $\{1,3,4\}$, *leading to an objective function value of* $z(1,3,4) = 80+0.8\cdot80+0.8\cdot0.2\cdot80-57-8-9 = 82.8$.

j	π_j	c_j	r_j
1	0.8	57	100
2	0.4	24	200
3	0.2	8	400
4	0.1	9	800

Table 5.2: Data for Example 2.

5.3 A MILP formulation

Exploiting the ordering induced by (5.3), it is possible to give a very compact MILP formulation for UJSSP. In fact, numbering the jobs by non-increasing values of $Z_j = \pi_j r_j/(1-\pi_j)$, we can exploit the Z-ordering to avoid using disjunctive constraints that are typical of many compact formulations for scheduling problems.

$$\max \sum_{j=1}^{n} (r_j P_j - c_j x_j) \tag{5.21a}$$

$$P_j \le \pi_j x_j \qquad \forall j \in \{1,\dots,n\} \tag{5.21b}$$

$$P_j \le \pi_j (P_i + 1 - x_i) \qquad \forall i,j \in \{1,\dots,n\} : i < j \tag{5.21c}$$

$$x_j \in \{0,1\} \qquad \forall j \in \{1,\dots,n\} \tag{5.21d}$$

$$P_j \ge 0 \qquad \forall\, j \in \{1,\dots,n\} \tag{5.21e}$$

In this formulation, $x_j = 1$ if job j is selected. If $x_j = 1$, constraints (5.21b) and (5.21c) ensure that P_j is the product of the probabilities of the jobs selected up to j (included), whereas, if and only if job j is not selected, $P_j = 0$. Note that all the constraints (5.21c) containing x_i become non-binding if $x_i = 0$.

5.4 An upper bound

In what follows, we develop an upper bound for the problem. Recall that the jobs are numbered in non-increasing order of Z_j. We denote by $\pi_{1:l}^{[s]}$ the s-th largest probability in $\{\pi_1, \pi_2, \dots, \pi_l\}$, $s = 1, \dots, l$ (and we let $\pi_{1:l}^{[0]} = 1$). Now consider a job j, and suppose that

it is scheduled in position $k \leq j$. An upper bound on its contribution to the objective function is given by

$$\max\{0, \left(\Pi_{s=0}^{k-1}\pi_{1:j-1}^{[s]}\right)\pi_j r_j - c_j\}. \tag{5.22}$$

We now define the following quantities:

$$q_{jk} = \begin{cases} \max\{0, \left(\Pi_{s=0}^{k-1}\pi_{1:j-1}^{[s]}\right)\pi_j r_j - c_j\} & \text{if } k \leq j \\ 0 & \text{if } k > j \end{cases}$$

An upper bound can be derived by solving the following assignment problem, in which $x_{jk} = 1$ if job j is assigned to position k:

$$\max \quad \sum_{j=1}^{n}\sum_{k=1}^{n} q_{jk}x_{jk} \tag{5.23a}$$

$$\sum_{k=1}^{n} x_{jk} = 1 \qquad \forall\, j = 1, \ldots, n \tag{5.23b}$$

$$\sum_{j=1}^{n} x_{jk} = 1 \qquad \forall\, k = 1, \ldots, n \tag{5.23c}$$

$$x_{jk} \geq 0 \qquad \forall\, j = 1, \ldots, n, k = 1, \ldots, n \tag{5.23d}$$

This upper bound can be slightly refined. It can happen that a job j is assigned to a position k, while a job $j' > j$ is assigned to position $k' < k$. We can make sure the Z-order is maintained by adding constraints (5.24a) or constraints (5.24b), (5.24c), (5.24d), and (5.24e).

$$x_{jk} + x_{j'k'} \leq 1 \qquad \forall\, j, j', k, k' \in \{1, \ldots, n\} : j < j', k' < k \tag{5.24a}$$

$$\sum_{j'<j}\sum_{k'>k} x_{j'k'} \leq M_{1jk} \cdot \delta_{1jk} \qquad \forall j \in \{2, \ldots, n\}, k \in \{1, \ldots, n-1\} \tag{5.24b}$$

$$\sum_{j'\geq j}\sum_{k'\leq k} x_{j'k'} \leq M_{2jk} \cdot \delta_{2jk} \qquad \forall j \in \{2, \ldots, n\}, k \in \{1, \ldots, n-1\} \tag{5.24c}$$

$$\delta_{1jk} + \delta_{2jk} \leq 1 \qquad \forall j \in \{2, \ldots, n\}, k \in \{1, \ldots, n-1\} \tag{5.24d}$$

$$\delta_{1jk}, \delta_{2jk} \in \{0, 1\} \qquad \forall j \in \{2, \ldots, n\}, k \in \{1, \ldots, n-1\} \tag{5.24e}$$

with $M_{1jk} = \min\{n - k, j\}$ and $M_{2jk} = \min\{k, n - j\}$.

5.5 Conclusions

In this chapter, we have introduced the *Unreliable Job Selection and Sequencing Problem* (UJSSP), a stochastic scheduling problem in which a subset of jobs must be selected and sequenced on a single machine subject to permanent failure. The objective is to maximize the expected net profit, accounting for job-specific rewards, costs, and probabilities of success. We have shown that UJSSP generalizes the *n-out-of-n Test Sequencing Problem*, is equivalent to the *College Selection Problem*, and constitutes a special case of the

Simultaneous Selection Problem. These connections have enabled us to identify two polynomially solvable cases, when all job costs or all success probabilities are identical, where a greedy algorithm yields an optimal solution. Furthermore, we have established the NP-hardness of the general UJSSP through a reduction from the strongly NP-hard *Product Partition Problem*. The chapter presented is part of a wider study [104] in which also several exact solution approaches are proposed, including a dynamic programming algorithm and two novel stepwise exact algorithms.

Several directions for future research remain open. The computational complexity of the general UJSSP with possibly fractional costs and rewards has not been fully characterized: it remains unclear whether the problem is weakly or strongly NP-hard. Additionally, the approximability of the greedy algorithm in the general setting is unknown, though a related result exists for the somewhat similar Product Knapsack problem, where [105] established a constant-factor approximation guarantee. The complexity of UJSSP with uniform rewards also remains to be established.

Chapter 6

Scheduling Jobs on Unreliable Machines Subject to Linear Risk

This chapter addresses a new class of scheduling problems in the context of machines subject to unrecoverable interruptions. Each job has a certain processing time and a reward which is attained if the job is successfully completed. For the failure process, we consider the linear-risk model, according to which the probability of machine failure is uniform across a certain time horizon. We analyze both the situation in which the set of jobs is given, and that in which jobs must be selected from a pool of jobs, at a certain selection cost. We characterize the complexity of various problems, showing both hardness results and polynomial algorithms.

The layout of this chapter is as follows. In Section 6.1, we review the relevant literature on these kinds of problems. In Section 6.2, the various complexity results are presented. Section 6.3 reports the results of a set of computational experiments. Finally, in Section 6.4, some conclusions are drawn.

This chapter addresses the following scenario. Consider a job set $J = \{1, \ldots, n\}$ and m identical machines. Each job requires a certain processing time p_j and, if it is successfully carried out, a reward r_j is gained. The machines are subject to unrecoverable interruptions.

In this context, we address two different problem types:

(i) Given J, the problem is to assign the jobs to the m machines and sequence them on each machine in order to maximize the *expected reward*. We called this problem the *unrecoverable interruption machine-scheduling problem*, and denote it by UIMSP(m).

(ii) Given J, for each job, a *selection cost* c_j is further specified for selecting it. The problem is then to decide which subset $S \subseteq J$ of jobs should be selected, and how the selected jobs should be assigned and sequenced on the machines in order to maximize the *net expected reward*, i.e., the expected reward of the selected jobs minus their selection cost. The problem is then called the *unrecoverable interruption machine-scheduling problem with job selection*, denoted by UIMSPS(m). The cardinality of S may or may not be fixed, giving rise to two variants of the problem. If it is imposed that $|S| = k$, the problem is denoted by UIMSPS$_k$(m).

Note that c_j represents the cost to secure job j, e.g., the cost of raw materials or other fixed costs. It is important to underline that, even if a selected job is not carried out because of the interruption, the cost c_j is borne anyhow.

For UIMSP (and UIMSPS) to be fully defined, one must specify the machine failure process. In this chapter, we addressed the case in which the failure process follows a *linear risk model*, i.e., the probability of interruption is proportional to the amount of time elapsed since the beginning of the schedule (see, e.g., [88]). In our chapter, we considered that a *time horizon* of length T is given, i.e., the jobs must be completed within a certain time horizon T, as the machine is not available afterwards. Besides representing all situations with a constant failure rate, the linear risk assumption can conveniently represent situations in which access to the processing resource is given at budget cost, but it is not granted for the whole time horizon, as some higher-priority users may claim the resource at any time [88]. Without any information on when the resource can be withdrawn, we assume that the interruption probability is uniformly distributed across the time horizon, hence yielding the linear risk model.

In this chapter, we address various problems:

- UIMSP(1). In this case, we show that the problem can be solved in polynomial time by a simple priority rule.

- UIMSPS(1). We show that UIMSPS$_k$(1) is, in general, NP-hard, but some special cases are polynomially solvable. We derived a particularly efficient algorithm for the case in which jobs have a unit-processing time ($p_j = 1$).

- UIMSP(m). We show that the problem is, in general, NP-hard, but it can be solved in polynomial time for the special case in which all jobs have the same reward ($r_j = r$) or the same processing times ($p_j = p$).

6.1 Literature Review

Scheduling models in which machines may fail constitute a meaningful subfield in the area of machine scheduling. Unlike usual scheduling settings, the manager's objectives are not related to classical objective functions such as tardiness or flow time-related measures, but rather to maximizing the expected profit. The models addressed in this chapter have applications in both manufacturing [92] and computer systems [88].

As widely described in Chapter 5, a stream of works exists in which failure probabilities are associated with the jobs: for each job j, a probability π_j is given that the machine will fail while performing that job (Unreliable Job Sequencing Problem, UJSP). The problem is to sequence the jobs in order to maximize the expected reward. Under a different name, this problem was introduced by Stadje [90], who was the first to show that a simple priority rule solves UJSP without further constraints. For $m = 2$, UJSP becomes NP-hard [92], and a list-scheduling algorithm on m machines produces a solution that is 0.8535-approximate [106]. Notice that, if the machine failure process is exponentially distributed, the probability that a job is successfully carried out does not depend on the amount of time elapsed, but only on the job's processing time. In this case, if p_j denotes the processing time of job j, then $\pi_j = e^{-\lambda p_j}$. In other words, when machine failures are exponentially distributed, UJSP and UIMSP coincide. This special case of UJSP is dealt

with in [107]. We are not aware of papers that have addressed UIMSP with linear risks, but in [104], the need for investigating this type of failure model is pointed out.

The selection variant of UJSP (UJSSP) has received less attention in the literature; however, it has been shown [90] that the problem of selecting the jobs in order to maximize the net expected reward can be solved by a greedy algorithm when all the jobs have the same selection cost. [100] further characterized special cases of selection problems (including UJSP), which can be solved by a greedy approach. However, in the previous papers, the complexity of the general UJSP with selection costs was left open. We are not aware of papers that have addressed UIMSP with job selection (i.e., UIMSPS). Our study, using a linear risk model, is the first contribution in this sense.

6.2 Methodology

In this section, we address the complexity of various cases of UIMSP and UIMSPS when failures follow a linear risk model. Under this model, a time horizon T is specified, and the failure probability is *uniformly distributed* across T. Hence, the probability that a machine is still working at time $t < T$, which can be called q_t, is given by

$$q_t = 1 - \frac{t}{T}. \tag{6.1}$$

When $m \geq 2$, machines may fail independently, and the time horizon T is the same for each machine.

In the following, if $T \geq \sum_i p_i$, we say that T is *nonbinding*; otherwise, we say that T is *binding*.

6.2.1 UIMSP(1)

Let us start from the simplest problem, i.e., there is a single machine and there are no selection costs. In what follows, we consider the case when T is nonbinding.

Given a schedule σ, let $\sigma(i)$ denote the job in the i-th position. So, the first scheduled job is $\sigma(1)$, of length $p_{\sigma(1)}$ and reward $r_{\sigma(1)}$. Due to (6.1), the expected reward stemming from the execution of the first scheduled job is given by

$$r_{\sigma(1)} \left(1 - \frac{p_{\sigma(1)}}{T}\right).$$

The reward of $\sigma(2)$ is attained if the machine does not fail within time $p_{\sigma(1)} + p_{\sigma(2)}$; hence, its contribution to the expected reward is

$$r_{\sigma(2)} \left(1 - \frac{p_{\sigma(1)} + p_{\sigma(2)}}{T}\right),$$

and so on. Given a schedule $\sigma = \{\sigma(1), \sigma(2), \ldots, \sigma(n)\}$, the expected reward $ER(\sigma)$ of the schedule σ is therefore given by

$$ER(\sigma) = \sum_{i=1}^{n} r_{\sigma(i)} \left(1 - \frac{\sum_{k=1}^{i} p_{\sigma(k)}}{T}\right). \tag{6.2}$$

Example 3. *Consider the example in Table 6.1, where* $T = 10$. *Figure 6.1(a) shows the probability* q_t *that the machine is still working at time* t. *Starting from 1,* q_t *linearly decreases, becoming 0 at time* $T = 10$. *Figure 6.1(b) illustrates a feasible schedule* $\{2, 3, 1\}$. *Under this schedule, the probability of successfully carrying out job 2 is* $(1 - p_2/T) = 0.6$*; hence, the expected reward from scheduling job 2 is* $r_2(1 - p_2/T) = 80 \cdot 0.6 = 48$. *The probability of successfully carrying out job 3 is* $(1 - (p_2 + p_3)/T) = 0.3$, *and therefore, job 3 contributes by* $r_3(1 - (p_2 + p_3)/T) = 55 \cdot 0.3 = 16.5$. *A similar computation for job 1 yields* $r_1(1 - (p_2 + p_3 + p_1)/T) = 50 \cdot 0.1 = 5$*; hence, the total expected reward of this schedule is* $z = 69.5$.

i	1	2	3
p_i	2	4	3
r_i	50	80	55
p_i/r_i	0.04	0.05	0.0545

Table 6.1: Data for Example 3.

We next show that UIMSP(1) can be efficiently solved when T is nonbinding. From (6.2), we have

$$ER(\sigma) = \sum_{i=1}^{n} r_{\sigma(i)} - \frac{1}{T} \sum_{i=1}^{n} r_{\sigma(i)} \left(\sum_{h=1}^{i} p_{\sigma(h)} \right). \tag{6.3}$$

Since the first term of the above expression is a constant, the expected reward $ER(\sigma)$ is maximized when

$$\frac{1}{T} \sum_{i=1}^{n} r_{\sigma(i)} \left(\sum_{h=1}^{i} p_{\sigma(h)} \right) \tag{6.4}$$

is minimized. Now consider the classical scheduling problem $1||\sum w_j C_j$, consisting of minimizing the total weighted completion time of n jobs, with each having a processing time p_j and a weight w_j. If we let $w_j := r_j/T$, observing that $p_{\sigma(1)} + \ldots, p_{\sigma(i)}$ is the completion time of the job in the i-th position, (6.4) coincides with the value of the total weighted completion time of a given schedule. As a consequence, (6.4) is minimized by sequencing the jobs according to Smith's rule [108], i.e., by nondecreasing order of the ratios

$$\frac{p_j}{r_j}. \tag{6.5}$$

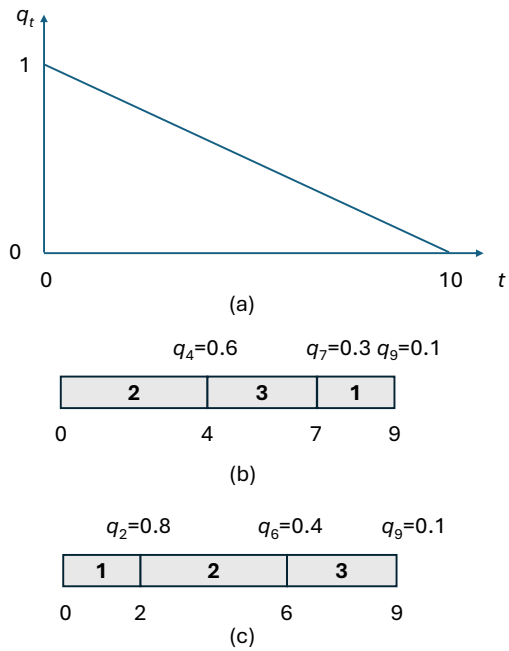

Figure 6.1: The linear risk function (a) and two feasible schedules (b) and (c) for Example 3.

Theorem 4. *When $m = 1$, UIMSP under linear risk is solved in $O(n \log n)$.*

Note that the ordering rule is fairly intuitive, in that it gives precedence to jobs with a small processing time and a large reward.

Example 4. *By sequencing the jobs of Example 3 by increasing ratios p_j/r_j, one obtains the schedule in Figure 6.1(c), i.e., schedule $\{1, 2, 3\}$. This is the optimal schedule, and its value is given by* $50(1 - 2/10) + 80(1 - (2 + 4)/10) + 55(1 - (2 + 4 + 3)/10) = 77.5$.

Finally, we observe that, if T is binding (i.e.,$T < \sum_j p_j$), the nature of the problem changes significantly, since no jobs can complete after T. Thus, the problem becomes deciding which jobs should be completed before T, and UIMSP(1) becomes equivalent to UIMSPS(1) with no costs. The complexity of UIMSP(1) in this case is still open.

6.2.2 UIMSPS(1) and UIMSPS$_k$(1)

In this section, we address the problem with selection costs; i.e., each job j also has a cost c_j that is paid if the job is selected for processing. Notice that, once a subset $S \subseteq J$ of jobs is selected to be performed within T, the optimal sequence is obtained by ordering the jobs according to (6.5). Hence, UIMSPS(1) essentially consists of finding S. In what follows, we denote by $z(S)$ the value of the net expected reward when S is selected, i.e., from (6.2):

$$z(S) = ER(S) - \sum_{j \in S} c_j \tag{6.6}$$

We next propose a dynamic programming algorithm for solving UIMSPS(1). In what follows, we assume that the jobs are numbered by *nondecreasing values of the ratio* (6.5). For a given positive integer $B \leq T$, let $\mathcal{P}(j, B)$ denote the subproblem restricted to jobs $\{1, 2, \ldots, j\}$, under the condition that the last selected job completes exactly at time B. We denote by $F(j, B)$ the value of the optimal solution of $\mathcal{P}(j, B)$. It is possible to express $F(j, B)$ by means of the following recursive formula:

$$F(j, B) = \max\{F(j-1, B); F(j-1, B-p_j) + r_j(1 - \frac{B}{T}) - c_j\} \tag{6.7}$$

The first term corresponds to the case in which j is *not* selected in the optimal solution of $\mathcal{P}(j, B)$. The second term corresponds to selecting j, in which case job j is appended at the end of the optimal solution of $\mathcal{P}(j, B - p_j)$, and the contribution of job j is accounted for. The value of the optimal solution of UIMSPS(1) is given by

$$z^* = \max_{0 \leq B \leq T} \{F(n, B)\}. \tag{6.8}$$

The correctness of the algorithm is derived from the following considerations. Suppose that, for some (j, B), in the optimal solution of $\mathcal{P}(j, B)$, there is some idle time of length $\tilde{I}$ between two consecutive jobs (which is possible, to meet the constraint that the last job precisely completes at B). By eliminating such an idle time, we obtain a schedule that is feasible for the subproblem $\mathcal{P}(j, B - \tilde{I})$ and has a strictly larger net expected reward. Hence, as we consider all values of B in (6.8), the makespan $B^* \leq T$ of the optimal solution to UIMSPS(1) is also included.

Formula (6.7) must be suitably initialized, as follows:

$$F(j, 0) = 0 \quad \text{for all } j = 1, \cdots, n,$$

$$F(0, B) = 0 \quad \text{for all } B = 0, \cdots, T,$$

$$F(j, B) = -\infty \quad \text{for all } B < 0,$$

$$F(1, B) = \begin{cases} \max\{0, r_1(1 - p_1/T) - c_1\} & \text{if } p_1 \leq B \\ -\infty & \text{if } p_1 > B \end{cases}$$

In conclusion, the following result holds.

Theorem 5. *UIMSPS(1) can be solved in $O(nT)$ time.*

Proof. Formula (6.7) has to be computed for each value of i and B. As each $F(i, B)$ is simply obtained by comparing two terms, it can be computed in constant time, and the thesis follows. □

In Section 6.3, we report the results of some computational experiments about the viability of the above dynamic programming approach. Notice that, although the complexity of UIMSPS(1) is open, the result of Theorem 5 rules out the strong NP-hardness of UIMSPS(1).

Concerning UIMSPS(1), as its complexity is open, a relevant issue is to figure out whether it could be solved by a greedy approach. A greedy algorithm in this context

simply consists of iteratively adding to the set of selected jobs a new job that maximizes the improvement of the objective function (see Algorithm 2). The algorithm stops when no other jobs can be added to the current set, either because the time horizon would be violated or because any additions would only decrease the value of the objective function. In this algorithm, $P(S)$ denotes the total processing time of a set S of jobs.

Algorithm 2 Greedy algorithm for UIMSPS(1)

Input: job set J, values p_j, r_j, c_j $(j = 1, \ldots, n)$ and T
Output: a feasible set S of selected jobs
$S = \{\emptyset\}$; $improve := true$;
while $improve = true$ **do**
  $max =: z(S)$; **for** $j \notin S$ **do**
    Compute $z(S \cup \{j\})$ by (6.6);
    if $z(S \cup \{j\}) > max$ **then**
    h:=j, max:=$z(S \cup \{j\})$
  if $max > z(S)$ **then**
  $S := S \cup \{h\}$ **else**
    $improve := false$
return S.

The following example shows that the greedy algorithm may not return the optimal solution.

Example 5. *Consider an instance of UIMSPS(1) with the data in Table 6.2, and the time horizon is* $T = 83$. *Jobs are numbered according to* (6.5). *The greedy algorithm starts by considering singletons, as follows:* $z(\{1\}) = r_1(1 - p_1/T) - c_1 = 5366.26$, $z(\{2\}) = r_2(1-p_2/T)-c_2 = 5483.97$, *and* $z(\{3\}) = r_3(1-p_3/T)-c_3 = 5101.2$. *Among these options, the best is* $z(\{2\})$, *so job 2 is added to* S. *Next, we observe that all sets with two jobs have a total length smaller than* T. *Hence, the next step of the greedy algorithm compares sets* $\{1, 2\}$ *and* $\{2, 3\}$, *yielding* $z(\{1, 2\}) = r_1(1 - p_1/T) + r_2(1 - (p_1 + p_2)/T - c_1 - c_2 = 5493.61$ *and* $z(\{2, 3\}) = r_2(1-p_2/T)+r_3(1-(p_2+p_3)/T-c_2-c_3 = 5404.45$. *As* $z(\{1, 2\}) > z(\{2, 3\})$, *job 1 is added to* S. *Since* $p_1 + p_2 + p_3 > T$, *the set* $\{1, 2, 3\}$ *cannot be considered, and in conclusion, the greedy algorithm returns set* $S = \{1, 2\}$. *However, it is easy to check that the optimal solution is* $S^* = \{1, 3\}$, *as* $z(\{1, 3\}) = r_1(1-p_1/T)+r_3(1-(p_1+p_3)/T-c_1-c_3 = 5768.67 > z(\{1, 2\})$.

j	1	2	3
p_j	39	43	39
r_j	10,500	11,400	10,000
c_j	200	10	200
p_j/r_j	0.00371	0.00377	0.0039

Table 6.2: Data for Example 5.

We next address UIMSPS$_k$, i.e., the case in which $|S| = k$. To establish the complexity of the problem, we first express UIMSPS$_k(1)$ in decision form:

UIMSPS$_k(1)$_DEC. *A set J of n' jobs is given. Each job has a processing time p_i, a reward r_i, and a cost c_i, $i = 1, \ldots, n'$. Given an integer T, and the linear risk function*

$$1 - \frac{t}{T},$$

is there a subset S of k jobs such that, when selected and scheduled on the machine, the net expected reward is at least R?

In the proof, we use the following well-known NP-complete problem:

EQUAL-SIZE PARTITION. *Given a set of positive integers, $N = \{a_1, a_2, \ldots, a_n\}$ (n even), such that $\sum_{i=1}^{n} a_i = A$, is there an equal-size partition, i.e., a partition $(S, \bar{S})$ of N, such that $|S| = |\bar{S}| = n/2$ and*

$$\sum_{i \in S} a_i = \sum_{i \in \bar{S}} a_i = \frac{A}{2}?$$

Theorem 6. UIMSPS$_k(1)$_DEC *is NP-complete.*

Proof. Clearly, UIMSPS$_k(1)$_DEC is in NP. Consider an instance of an EQUAL-SIZE PARTITION with n integers, in which we denote as a_{min} the smallest integer in N. We define an instance of UIMSPS(1)_DEC as follows. There are $n' := n$ jobs, corresponding to the integers of the EQUAL-SIZE PARTITION, and we let $k := n/2$. The processing times and the rewards of the jobs are defined as

$$r_i := p_i := A^2 + a_i, i = 1, \ldots, n. \tag{6.9}$$

Note that, since $p_i/r_i = 1$ for all jobs, the order in which the selected jobs are sequenced is immaterial. The time horizon T is defined as

$$T := \frac{\sum_{i=1}^{n} p_i}{2} = \frac{nA^2 + A}{2}. \tag{6.10}$$

For all $i = 1, \ldots, n$, let

$$c_i := K - \frac{p_i^2}{2T}, \tag{6.11}$$

where K is a suitable integer defined later on. We want to establish if there is a subset S of $n/2$ jobs yielding a net expected profit of at least

$$R := \frac{T}{2} - \frac{n}{2}K. \tag{6.12}$$

Consider a subset S of jobs. For the sake of simplicity, let us number the selected jobs with $1, 2, \ldots, n/2$. The corresponding value z of the net expected reward is, from (6.2),

$$z(S) = \sum_{j=1}^{n/2} p_j \left(1 - \frac{\sum_{i=1}^{j} p_i}{T}\right) - \sum_{j=1}^{n/2} c_j = \sum_{j=1}^{n/2} (p_j - c_j) - \frac{1}{T}\left(\sum_{j=1}^{n/2} \sum_{i=1}^{j} p_i p_j\right) = \tag{6.13}$$

$$= \sum_{j=1}^{n/2}(p_j - c_j) - \frac{1}{T}\left(\sum_{j=1}^{n/2} p_j^2 + \sum_{i<j} p_i p_j\right). \tag{6.14}$$

Since, given $n/2$ integers $\{p_1, \ldots, p_{n/2}\}$,

$$\sum_{i<j} p_i p_j = \frac{(\sum_{i=1}^{n/2} p_i)^2 - \sum_{i=1}^{n/2} p_i^2}{2},$$

we can rewrite (6.14) as

$$z(S) = \sum_{j=1}^{n/2}(p_j - c_j) - \frac{1}{T}\left(\sum_{j=1}^{n/2} p_j^2 + \frac{1}{2}((\sum_{j=1}^{n/2} p_j)^2 - \sum_{j=1}^{n/2} p_j^2)\right) \tag{6.15}$$

$$= \sum_{j=1}^{n/2}(p_j - c_j) - \frac{1}{2T}\left((\sum_{j=1}^{n/2} p_j)^2 + \sum_{j=1}^{n/2} p_j^2\right) \tag{6.16}$$

and, from (6.11),

$$= \sum_{j=1}^{n/2}(p_j - K + \frac{p_j^2}{2T}) - \frac{1}{2T}\left((\sum_{j=1}^{n/2} p_j)^2 + \sum_{j=1}^{n/2} p_j^2\right) \tag{6.17}$$

$$= \sum_{j=1}^{n/2} p_j - \frac{n}{2}K - \frac{1}{2T}\left(\sum_{j=1}^{n/2} p_j\right)^2. \tag{6.18}$$

Now observe that the term $-Kn/2$ is a constant, as it does not depend on which jobs are selected. Hence, the problem is to select $n/2$ jobs so that the target value R specified in (6.12) is met. Considering the function

$$f(x) = x\left(1 - \frac{x}{2T}\right),$$

we can rewrite the net expected reward (6.18) as

$$z(S) = f(\sum_{j=1}^{n/2} p_j) - \frac{n}{2}K. \tag{6.19}$$

We observe that $f(x)$ is maximized by $x = T$, and the maximum value is $f(T) = T/2$. This means that, if there is a set S of $n/2$ jobs such that

$$\sum_{j \in S} p_j = T,$$

then (6.18) is the maximum, attaining the value

$$\frac{T}{2} - \frac{n}{2}K = R.$$

Therefore, we can conclude that a selection S of $n/2$ jobs achieving the net expected profit of R exists if, and only if, there are $n/2$ jobs such that

$$\sum_{j\in S} p_j = \frac{nA^2}{2} + \sum_{j\in S} a_j = T = \frac{nA^2 + A}{2},$$

i.e., if, and only if,

$$\sum_{j\in S} a_j = \frac{A}{2},$$

which holds if, and only if, the instance of UIMSPS$_k(1)$_DEC is a yes-instance.

We are only left with showing that a suitable value of K exists. Since the costs c_j must be positive, from (6.11), it must hold that, for all $j = 1, \ldots, n$,

$$K > \frac{p_j^2}{2T}. \tag{6.20}$$

On the other hand, it must also hold that $c_i < p_i$ for all i; hence,

$$c_j = K - \frac{p_j^2}{2T} < p_j. \tag{6.21}$$

Consider the following:

$$K := \frac{(A^2 + A)^2}{2T}.$$

Since $p_j^2 = (A^2 + a_j)^2 < (A^2 + A)^2$, (6.20) holds for all $j = 1, \ldots, n$. Moreover, recalling the definition of T (6.10), we can write

$$K = \frac{A^4 + A^2 + 2A^3}{2T} = \frac{A^4}{2T} + \frac{A^2 + 2A^3}{nA^2 + A} < \frac{A^4}{2T} + \frac{1}{n} + \frac{2A^2}{nA + 1} < \frac{A^4}{2T} + \frac{1}{n} + \frac{2A}{n};$$

since, obviously, $1/n + 2A/n < A^2$, and from (6.9), $A^2 + a_{\min} \leq p_j$ for all j; thus, one has

$$K < \frac{A^4}{2T} + A^2 < \frac{(A^2 + a_{min})^2}{2T} + (A^2 + a_{min}) \leq \frac{p_j^2}{2T} + p_j,$$

Also, (6.21) is fulfilled for all $j = 1, \ldots, n$. Finally, as $K < 4A^4$, we observe that the number of bits necessary to encode K does not exceed $\log(4A^4) = 16 \log A$, hence resulting in a polynomial in the input size of the EQUAL-SIZE PARTITION. □

UIMSPS$_k$(1) with Identical Rewards

We next consider the special case of UIMSPS$_k$ in which all jobs have the same reward, i.e., $r_j = r$ for all $j = 1, \ldots, n$. In this case, the selected jobs are simply sequenced by nondecreasing processing times (SPT rule). We show that this problem can be solved in polynomial time.

Given a set S, let us denote the k selected jobs as $\sigma(1), \sigma(2), \ldots, \sigma(k)$, and suppose they are numbered by nondecreasing processing times. Then, the value of the objective function is

$$z(S) = k - \frac{r}{T}\left(kp_{\sigma(1)} + (k-1)p_{\sigma(2)} + \cdots + (k-h+1)p_{\sigma(h)} + \cdots + p_{\sigma(k)}\right) - \sum_{i=1}^{k} c_{\sigma(i)}$$

$$= rk - \frac{r}{T}\sum_{i=1}^{k}(k-i+1)p_{\sigma(i)} - \sum_{i=1}^{k} c_{\sigma(i)}.$$

As k is fixed, the objective of maximizing $z(S)$ is therefore equivalent to that of minimizing the following function $\bar{z}(S)$:

$$\bar{z}(S) = \frac{r}{T}\sum_{i=1}^{k}(k-i+1)p_{\sigma(i)} + \sum_{i=1}^{k} c_{\sigma(i)}. \tag{6.22}$$

From this expression, if a job j is selected and assigned to position h, its contribution to $\bar{z}(S)$ is given by

$$\frac{r}{T}(k-h+1)p_j + c_j. \tag{6.23}$$

Suppose now that the jobs are numbered by nondecreasing processing times. UIMSPS$_k$ is equivalent to the following assignment problem, in which $x_{jh} = 1$ if job j is assigned to position h:

$$\min \quad \sum_{j=1}^{n}\sum_{h=1}^{k} q_{jh}x_{jh} \tag{6.24a}$$

$$\sum_{j=1}^{n} x_{jh} = 1 \qquad \forall\, h = 1, \ldots, k \tag{6.24b}$$

$$\sum_{h=1}^{k} x_{jh} \leq 1 \qquad \forall\, j = 1, \ldots, n \tag{6.24c}$$

$$x_{jh} \geq 0 \qquad \forall\, j = 1, \ldots, n \quad h = 1, \ldots, k \tag{6.24d}$$

where

$$q_{jh} = \begin{cases} \frac{r}{T}(k-h+1)p_j + c_j & \text{if} \quad h \leq j \\ +\infty & \text{if} \quad h > j \end{cases} \tag{6.25}$$

The correctness of Formulation (6.24) is derived from the fact that exactly k jobs are selected, as implied by the k assignment constraints (6.24b). Given an optimal solution $\{x^*\}$ of (6.24a)–(6.24d), the optimal set S_k^* of selected jobs is given by

$$S_k^* = \{j | x_{jh}^* = 1 \text{ for some } h\}.$$

Notice that the second case in (6.25) is derived from the observation that we can assume that a job j will not occupy a position h larger than j. Suppose, in fact, that $j \in S_k^*$ and j is assigned position $h > j$. Hence, some other job $u \in S_k^*$ such that $p_u \geq p_j$

is assigned a position $\ell < h$. By switching positions between jobs j and u, it is easy to check that $\bar{z}(S)$ decreases in the amount of $(h-\ell)(p_u - p_j)$, which is always nonnegative. We can therefore conclude the following:

Theorem 7. *UIMSPS$_k$(1), when $r_j = r$, can be solved in $O(n^3)$.*

If we solve UIMSPS$_k$(1) for all values $k = 1, \ldots, n$, we obtain n optimal subsets, $S_1^*, S_2^*, \ldots, S_n^*$. Clearly, the optimal solution to UIMSPS(1) (without a cardinality constraint on S) is

$$S^* = \arg \max_{1 \le k \le n} \{z(S_k^*)\},$$

so we obtain the following result.

Theorem 8. *UIMSPS(1), when $r_j = r$, can be solved in $O(n^4)$.*

UIMSPS$_k$(1) with Unit-Processing Times

Let us now consider another special case, namely when $p_j = 1$ for all j, and we must select k jobs. Since, in this case, the jobs have identical processing times, from (6.5), the selected jobs will be sequenced by nonincreasing rewards. Moreover, we obviously assume that $k \le T$, as otherwise, the problem would be infeasible.

As before, consider a set S of k selected jobs, denoted as $\sigma(1), \sigma(2), \ldots, \sigma(k)$, and suppose they are numbered by nonincreasing rewards. The value of the objective function is

$$z(S) = r_{\sigma(1)} \left(1 - \frac{1}{T}\right) + r_{\sigma(2)} \left(1 - \frac{2}{T}\right) + \cdots + r_{\sigma(k)} \left(1 - \frac{k}{T}\right).$$

Hence, if job i is selected and assigned to position j, its contribution to the objective function is

$$r_i \left(1 - \frac{j}{T}\right) - c_i. \tag{6.26}$$

Therefore, this special case of UIMSPS$_k$ can also be solved through an assignment problem. However, in this case, it is easy to see that the problem can be solved more efficiently, taking advantage of the following property.

Theorem 9. *Given an instance of UIMSPS in which all jobs have the same processing time $p_j = 1$, let S_{k-1} be the optimal set of UIMSPS$_{k-1}$. There exists an optimal set S_k for UIMSPS$_k$ that includes S_{k-1} ($k = 2, \ldots, n$).*

Proof. The proof is by induction on k. Consider $k = 1$, i.e., set S_1. In this case, the selected job h is such that

$$r_h \left(1 - \frac{1}{T}\right) - c_h = \max_{i \in N} \left\{ r_i \left(1 - \frac{1}{T}\right) - c_i \right\}. \tag{6.27}$$

Now suppose that the set S_2 does not include h, but it includes jobs i and j, so that

$$z(S_2) = r_i \left(1 - \frac{1}{T}\right) + r_j \left(1 - \frac{2}{T}\right) - c_i - c_j.$$

If, in S_2, we replace i with h, we obtain set S_2' such that

$$z(S_2') = r_h \left(1 - \frac{1}{T}\right) + r_j \left(1 - \frac{2}{T}\right) - c_h - c_j.$$

From (6.27), $z(S_2') \leq z(S_2)$; hence, S_2' is optimal and the thesis holds.

Now let us consider a generic value of k, and from the inductive hypothesis, the thesis holds for all of the following problems: UIMSPS$_1$, UIMSPS$_2$, ..., up to UIMSPS$_{k-1}$.

Let σ_k denote the optimal schedule for S_k, and, in such a schedule, let q be the *rightmost* job in σ_k such that $q \in S_k$ and $q \notin S_{k-1}$; additionally, consider the set $S_k \setminus \{q\}$. If $S_k \setminus \{q\} = S_{k-1}$, we are finished. Otherwise, let u be the position occupied by q in the optimal schedule σ_k for S_k. Since q is the rightmost "new" job, only jobs of S_{k-1} appear at the right of q (possibly none). Now, we can view σ_k as obtained by inserting q in position u in the schedule, which can be called σ'_{k-1}, for the set $S_k \setminus \{q\}$. R denotes the total reward of the $k-u+1$ jobs occupying the positions $u, u+1, \ldots, k$ in σ'_{k-1}. When we insert q in position u in σ'_{k-1}, each of the last $k-u+1$ jobs moves rightwards by one position, so their total contribution to the expected reward decreases by R/T. So, when inserting q in σ'_{k-1}, the expected *marginal* reward is

$$z(S_k) - z(S_k \setminus \{q\}) = r_q \left(1 - \frac{u}{T}\right) - \frac{R}{T} - c_q. \tag{6.28}$$

Now consider the optimal schedule σ_{k-1} for S_{k-1}, and a new schedule $\tilde{\sigma}$ obtained from σ_{k-1} by adding q in position u in σ_{k-1}. Since we are inserting q in position u, its contribution to the increase in the expected reward is $r_q(1 - \frac{u}{T})$, as in (6.28). Now, letting R' denote the total reward of the jobs occupying the last $k-u+1$ positions in σ_{k-1}, as each of these jobs moves rightward by one position, going from σ_{k-1} to $\tilde{\sigma}$, their contribution to the total reward decreases by R'/T. The key observation is that R' is certainly not greater than the total reward R of the jobs *belonging to* S_{k-1}, which occupy the last $k-u+1$ positions in σ'_{k-1}. This is because the jobs occupying the last $k-u+1$ positions in σ_{k-1} are precisely the $k-u+1$ jobs with the smallest rewards in S_{k-1}. As a consequence, when inserting q in σ_{k-1}, the expected marginal reward is

$$z(S_{k-1} \cup \{q\}) - z(S_{k-1}) = r_q(1 - \frac{u}{T}) - \frac{R'}{T} - c_q. \tag{6.29}$$

Since: $R' \leq R$, the marginal expected reward from adding q to S_{k-1} (given by (6.29)) is not smaller than that attained from adding q to $S_k \setminus \{q\}$ (given by (6.28)). But since, from the inductive hypothesis, S_{k-1} is optimal for UIMSPS$_{k-1}$, $z(S_{k-1}) \geq z(S_k \setminus \{q\})$, one has from (6.28) and (6.29) that

$$z(S_{k-1} \cup \{q\}) \geq z(S_k).$$

Hence, adding q in position u to σ_{k-1} yields a schedule $\tilde{\sigma}$ that is not worse than σ_k, so $\tilde{\sigma}$ is optimal and the thesis holds. □

Theorem 9 allows a very simple greedy algorithm to be devised for UIMSPS$_k$ when $p_j = 1$ (Algorithm 3). Starting from an empty set S, at each step, the job that increases the net expected reward by the most is added to S, until the size k is reached.

Algorithm 3 Greedy algorithm for UIMSPS$_k$ when $p_j = 1$

Input: job set J, values r_j, c_j for $j = 1, \ldots, n$, time horizon T;
Output: optimal set S_k of selected jobs
$S_0 = \{\emptyset\}$;
for $i = 1$ *to* k **do**
  max=0;
  for $j \notin S_{i-1}$ **do**
    compute $z(S_{i-1} \cup \{j\})$ by rule (6.5);
    if $z(S_{i-1} \cup \{j\})$ >max **then**
    $h := j$, $max := z(S_{i-1} \cup \{j\})$
  $S_i := S_{i-1} \cup \{h\}$;
S_k is the optimal solution.

Let us now consider the complexity. Suppose that we maintain a vector $R(u)$ containing the total reward of the jobs occupying positions from u to the end of the schedule, i.e., to position i. For each value of i, we must select the job q that maximizes the marginal reward. For each job $j \notin S_i$, its optimal position can be found in $O(\log n)$ and the marginal reward computed in constant time through (6.29). Once the job q is found and its position $p(q)$ in the schedule is determined, we must update the vector $R(u)$ for all positions u preceding $p(q)$, which takes $O(n)$. As there are k steps, the following result holds.

Theorem 10. *When $p_j = 1$ for all j, UIMPSP$_k$ can be solved in $O(nk \log n)$.*

By solving for all values of k from 1 to n, the optimal value for UIMSPS is given by $\max_k\{z(S_k)\}$, so we obtain the following result:

Theorem 11. *When $p_j = 1$ for all j, UIMPSP can be solved in $O(n^2 \log n)$.*

UIMSPS$_k$(1) with Identical Selection Costs

Finally, let us consider the special case of UIMSPS$_k$(1) in which all jobs have the same cost, i.e., $c_j = c$ for all j. In this case, the total cost is kc independent of S; hence, the problem is to select exactly k jobs so that the expected reward is maximized. In view of (6.4), and identifying the rewards r_j with job weights, the problem is equivalent to that of selecting k jobs so that the value of their total weighted completion time is minimized. In turn, this can be shown to be equivalent to the special case of the classical job rejection problem, in which, given n jobs, $n - k$ jobs can be rejected, so that the total weighted completion time of the selected jobs is minimized. The complexity of this problem is currently open ([109], [110]).

6.2.3 UIMSP(m)

Let us now consider UIMSP with two parallel machines (UIMSP(2)). The time horizon T is supposed to be the same for both machines and nonbinding. Consider the following NP-complete problem in decision form:

$P2||\sum w_iC_i$. A set J of n' jobs is given, with two identical machines and a value F. Each job has a processing time p'_i and a weight w_i. Is there a schedule for the jobs on the two machines such that their total weighted completion time does not exceed F?

We next prove the NP-completeness of UIMSP(2) in decision form, as follows.

UIMSP(2)_DEC. *A set J of n' jobs is given with two machines, each of which has the linear risk function*

$$1 - \frac{t}{T}.$$

Each job has a processing time p_i and a reward r_i, $i = 1, \ldots, n'$. Is there a schedule of the jobs on the two machines such that the expected reward is at least R?

Theorem 12. UIMSP(2)_DEC *is NP-complete.*

Proof. Obviously, UIMSP(2)_DEC is in NP. Consider an instance of $P2||\sum w_jC_j$, with n' jobs and with job i having a processing time p'_i and a weight w_i, for $i = 1, \ldots, n'$ and a threshold value F. We define an instance of UIMSP(2)_DEC as follows. There are $n := n'$ jobs, with job i having a processing time $p_i := p'_i$ and a reward $r_i := w_i$. The time horizon is defined as $T := \sum_i p_i$, which is nonbinding. Then, we let $R := \sum_i r_i - \frac{F}{T}$. Consider now a schedule σ for UIMSP(2)_DEC, and let $S_1(\sigma)$ and $S_2(\sigma)$ denote the jobs assigned to M_1 and M_2, respectively, in σ. Denoting with $C_j(\sigma)$ the sum of the processing times of the jobs preceding j (including j) on the respective machine, the expected reward of σ is given by

$$\sum_{j=1}^{n} r_j - \frac{1}{T}\left(\sum_{j\in S_1(\sigma)} r_jC_j(\sigma) + \sum_{j\in S_2(\sigma)} r_jC_j(\sigma)\right) = \sum_{j=1}^{n} r_j - \frac{1}{T}\left(\sum_{j=1}^{n} r_jC_j(\sigma)\right). \quad (6.30)$$

Hence, one can determine that there exists a schedule σ with an expected reward of at least R if, and only if,

$$\sum_{j=1}^{n} r_j - \frac{1}{T}\left(\sum_{j=1}^{n} r_jC_j(\sigma)\right) \geq R = \sum_{j=1}^{n} r_j - \frac{F}{T}. \quad (6.31)$$

Since $r_j = w_j$ and the processing times in the two problems are identical, (6.31) holds if, and only if,

$$\sum_{j=1}^{n} w_jC_j(\sigma) \leq F.$$

□

Consider now the special case in which the jobs have the same reward, $r_i = r$ for all i. In this case, UIMSP(m) can be efficiently solved.

Theorem 13. *If $r_j = r$ for all j,* UIMSP(m) *can be solved in $O(n \log n)$.*

Proof. Given a feasible schedule for UIMSP(m), let $S_1, S_2, \ldots, S_m$ denote the subsets of jobs assigned to machines $M_1, M_2, \ldots, M_m$, respectively ($S_1 \cup S_2 \cup \cdots \cup S_m = J$), and let

$\sigma^\ell(i)$ denote the job in position i on machine M_ℓ. From (6.3), we can rewrite the expected reward as

$$= n - \frac{r}{T} \sum_{\ell=1}^{m} \sum_{i=1}^{|S_\ell|} \left(\sum_{i=1}^{s} p_{\sigma^\ell(i)} \right). \tag{6.32}$$

Now observe that maximizing (6.32) is equivalent to minimizing the rightmost term, i.e., the total completion time of all jobs. Hence, the problem is equivalent to an instance of $P||\sum_j C_j$, which is solved by ordering the jobs in SPT order and assigning them in this order to the machines, in a round-robin fashion. The complexity is therefore given by the sorting step, $O(n \log n)$. □

6.2.4 UIMSPS(2) and UIMSPS$_k$(2) with Identical Rewards or Identical Processing Times

In the general case, as UIMSP(2) is NP-hard, so too is UIMSPS(2). We next focus on the special cases of UIMSPS$_k$(2) in which either $r_j = r$ for all $i = 1, \ldots, n$, or $p_j = p$ for all $i = 1, \ldots, n$, when T is nonbinding. Both of these cases can be reduced to the assignment problem.

If $r_j = r$ for all $j = 1, \ldots, n$, recalling that we want to *minimize* $\bar{z}(S_k)$ (given in (6.22)), from (6.23), one has that, if a job j is assigned position h on machine M_ℓ and a total of k_ℓ jobs are assigned to M_ℓ, the contribution to the objective function $\bar{z}(S_k)$ of job j is

$$\frac{r}{T}(k_\ell - h + 1)p_j + c_j. \tag{6.33}$$

while, if $p_j = p$ for all $j = 1, \ldots, n$, and recalling that we want to *maximize* $z(S_k)$, if j is assigned position h on machine M_ℓ, it completes at time hp, so its contribution to the objective function $z(S_k)$ is given by

$$r_j \left(1 - \frac{hp}{T}\right) - c_j. \tag{6.34}$$

Now, consider UIMSPS$_{k_1,k_2}$(2) in which $k_1 \leq k$ jobs must be assigned to M_1 and $k_2 = k - k_1$ jobs to M_2. If we let $x_{jh}^{(\ell)} = 1$ and if j is assigned position h on M_ℓ, the problem is solved through the following assignment problem:

$$\min \quad \sum_{j=1}^{n} \left(\sum_{h=1}^{k_1} q_{jh}^{(1)} x_{jh}^{(1)} + \sum_{h=1}^{k_2} q_{jh}^{(2)} x_{jh}^{(2)} \right) \tag{6.35a}$$

$$\sum_{j=1}^{n} x_{jh}^{(1)} = 1 \quad \forall\, h = 1, \ldots, k_1 \tag{6.35b}$$

$$\sum_{j=1}^{n} x_{jh}^{(2)} = 1 \quad \forall\, j = 1, \ldots, k_2 \tag{6.35c}$$

$$\sum_{h=1}^{k_1} x_{jh}^{(1)} + \sum_{h=1}^{k_2} x_{jh}^{(2)} \leq 1 \quad \forall\, i = 1, \ldots, n \tag{6.35d}$$

$$x_{jh}^{(1)} \geq 0 \quad \forall\, j = 1, \ldots, n \quad h = 1, \ldots, k_1 \tag{6.35e}$$

$$x_{jh}^{(2)} \geq 0 \qquad \forall\, j = 1, \ldots, n \quad h = 1, \ldots, k_2 \tag{6.35f}$$

where $r_j = r$ for all j, and it is assumed that the jobs are numbered by nondecreasing processing times:

$$q_{jh}^{(\ell)} = \begin{cases} \frac{r}{T}(k_\ell - h + 1)p_j + c_j & \text{if} \quad h \leq j \\ +\infty & \text{if} \quad h > j \end{cases} \tag{6.36}$$

If: $p_j = p$ for all j, assuming that the jobs are numbered by nonincreasing rewards, and recalling that we want to *maximize* $z(S_k)$,

$$q_{jh}^{(1)} = q_{jh}^{(2)} = \begin{cases} c_j - r_j \left(1 - \frac{jh}{T}\right) & \text{if} \quad h \leq j \\ +\infty & \text{if} \quad h > j \end{cases} \tag{6.37}$$

Note that, in both cases, j cannot be assigned a position h larger than j (on any machine).

Now, denoting by $S^*_{k_1,k_2}$ the optimal set of selected jobs for UIMSPS$_{k_1,k_2}(2)$ and denoting by $z(S^*_{k_1,k_2})$ the corresponding net expected reward, the optimal set S^*_k for UIMSPS$_k(2)$ is given by

$$z(S^*_k) = \max_{k_1,k_2}\{z(S^*_{k_1,k_2})\}.$$

Since the value k_1 can be chosen in k different ways (we can rule out $k_1 = 0$), the following result holds:

Theorem 14. *When $r_j = r$ for all j, or when $p_j = p$ for all j, and T is nonbinding, UIMSPS$_k(2)$ can be solved in $O(n^4)$.*

Notice that the same reasoning can be extended to UIMSPS$_k(m)$, fixing in all possible ways the number of jobs to be assigned to each machine $k_1, k_2, \ldots, k_m$ and showing that, in these two special cases, for *fixed* m, UIMSPS$_k(m)$ is polynomially solvable.

Theorem 15. *When $r_j = r$ for all j, or when $p_j = p$ for all j, T is nonbinding, and m is fixed, UIMSPS$_k(m)$ can be solved in $O(n^{m+2})$.*

The complexity of UIMSPS$_k(m)$ when m is not fixed is open.

Complexity results are summarized in Table 6.3 for UIMSP and in Table 6.4 for UIMSPS and UIMSPS$_k$.

Problem	Special cases	Complexity	
UIMSP(1)	T nonbinding	$O(n \log n)$	(Theorem 4)
UIMSP(1)	T binding	open	
UIMSP(2)		NP-hard	(Theorem 12)
UIMSP(m)	$r_j = r$	$O(n \log n)$	(Theorem 13)

Table 6.3: Summary of complexity results for UIMSP.

Problem	Special cases	Complexity	
UIMSPS(1)		open	
UIMSPS$_k$(1)		NP-hard	(Theorem 6)
UIMSPS$_k$(1)	$r_j = r$, T nonbinding	$O(n^3)$	(Theorem 7)
UIMSPS(1)	$r_j = r$, T nonbinding	$O(n^4)$	(Theorem 8)
UIMSPS$_k$(1)	$p_j = 1$	$O(nk \log n)$	(Theorem 10)
UIMSPS(1)	$p_j = 1$	$O(n^2 \log n)$	(Theorem 11)
UIMSPS$_k$(1)	$c_j = c$	open	
UIMSPS$_k$(2)	$r_j = r$	$O(n^4)$	(Theorem 14)
UIMSPS$_k$(2)	$p_j = p$	$O(n^4)$	(Theorem 14)
UIMSPS$_k$(m)	$r_j = r$ or $p_j = p$, m fixed	$O(n^{m+2})$	(Theorem 15)
UIMSPS$_k$(m)	$r_j = r$ or $p_j = p$, m not fixed	open	

Table 6.4: Summary of complexity results for UIMSPS and UIMSPS$_k$.

6.3 Results

In this section, we present the results of a computational experiment concerning UIMSPS(1). The aim of the experiment was to assess the viability of the dynamic programming algorithm presented in Section 6.2.2 to solve UIMSPS(1). The experiments were carried out in Python 3.11 within the PyCharm 2022.3.2 environment on a system equipped with a 12th Gen Intel(R) Core(TM) i9-12900 processor (2.40 GHz) and 128 GB of RAM. The machine runs Windows Server 2022 Standard, version 21H2.

We considered randomly generated instances for various values of the problem parameters. In particular, we generated instances with $n \in \{100, 1000, 10{,}000\}$, integer processing times uniformly drawn from $[1, 100]$, and a time horizon $T \in \{25n, 50n, 75n\}$. The rewards are uniformly distributed in $[1, 100]$. To define the costs, for each job, we sampled a coefficient

$$u_j \sim \text{Uniform}[0, 0.8],$$

and set

$$c_j = \lfloor r_j \cdot u_j \rfloor.$$

Notice that, in this way and for all jobs, the cost of the job is smaller than its reward, so no job was discarded a priori in our instances.

For each pair (n, T), 10 instances were generated. Table 6.5 reports the average CPU time of the dynamic programming algorithm. The results show that the computational time grows according to the theoretical complexity expressed in Theorem 5, and the dynamic program is a viable algorithm even for very large instances. In order to test the limits of the approach, we even ran an instance with $n = 100{,}000$ jobs and $T = 2{,}500{,}000$, which took a little more than 5 hours to complete.

n	T	CPU Time [s]
100	2500	0.020
	5000	0.039
	7500	0.059
1000	25,000	2.004
	50,000	4.054
	75,000	6.094
10,000	250,000	209.418
	500,000	447.123
	750,000	702.516

Table 6.5: Average CPU time (seconds) in each scenario (average over 10 instances).

6.4 Conclusions

In this chapter, we addressed a new model for scheduling jobs on machines subject to unrecoverable interruptions. The model applies to all those situations in which a value (reward) is attached to the successful accomplishment of a job, but the resource is withdrawn within a certain time horizon T. Here, we addressed the case of linear risk; i.e., the probability of the resource being withdrawn is uniform across T. In this scenario, we established the complexity of various job-selection and scheduling problems.

Tables 6.3 and 6.4 summarize the complexity results provided in the chapter, and they also point out relevant open problems from the viewpoint of complexity. In particular, the precise status of UIMSPS(1) is still open for what concerns NP-hardness. Concerning this problem, an interesting research topic is the analysis of the greedy algorithm presented in Section 6.2.2. Future research might pursue a theoretical bound on the approximation provided by such an algorithm, complemented by a thorough experimental study of the greedy and/or other heuristics.

Other relevant topics for future research on both UIMSP and UIMSPS deal with modeling issues, including scenarios in which (i) interruptions follow distributions other than the linear risk model or (ii) correlations exist among unrecoverable interruptions on different machines.

Part IV

General conclusions

General conclusions

This dissertation addresses three distinct thematic areas, all pertaining to resource allocation challenges. In Part I, we examined problems related to healthcare delivery, specifically focusing on surgery case assignment and the allocation of health services to healthcare facilities. The study on surgery case assignment (Chapter 2) presented a decision support framework designed to balance operational efficiency with scheduling flexibility, grouping patients into homogeneous clusters according to their resource consumption. We formulated the problem as an integer linear programming model and conducted a comprehensive simulation spanning one year, utilizing real data from a medium-size hospital. The results demonstrate that our approach can significantly improve measurable performance indicators compared to current practice. Importantly, the implementation shows that data-driven scheduling approaches need not replace clinical judgment; rather, they can complement it by providing decision support that respects both operational efficiency and clinical considerations. To facilitate practical adoption, we developed a user-friendly application that enables direct implementation of our model within the hospital environment, bridging the gap between theoretical optimization and real-world clinical practice. Future research could compare the cluster approach against purely centralized selection based solely on system performance, or allow random selection within the top-ranked patients of each cluster to better reflect surgeons' discretion based on individual clinical conditions.

Chapter 3 addressed the problem of allocating healthcare resources to Community Houses through a mixed-integer linear programming model. Data obtained from the Local Healthcare Authority, related to both the entire territory and chronic patients specifically, were used to validate the model and conduct illustrative case studies exploring its potential applications for organizational planning purposes. Compared to existing approaches in the literature, our model demonstrates a significant reduction in total distance traveled, providing evidence that systematic resource assignment strategies can yield substantial benefits in terms of accessibility and quality of life for patients. The model's effectiveness was further validated across different Community Houses configurations, consistently showing improved performance. Additionally, the integration of telemedicine services emerged as a crucial factor in reducing the total distance traveled by patients, highlighting the importance of hybrid care delivery models in modern healthcare systems. Future work

should introduce maximum distance constraints to prevent excessive travel burdens for individual patients, testing progressively tighter thresholds to balance system efficiency with equity. A multi-objective framework could explicitly trade off total distance minimization against equitable burden distribution across geographic areas.

Part II consists of Chapter 4, which addresses the problem of minimizing project makespan while controlling the number of activity preemptions. We present an integer linear programming approach and evaluate its performance on standard benchmark instances as well as on a real-world case study from the pharmaceutical industry. A comprehensive analysis was conducted on scenarios involving resources with limited overtime availability. The results reveal that allowing activity interruptions can yield significant improvements in project makespan, though the extent of these benefits depends critically on resource availability patterns. This finding suggests that preemption strategies should be carefully tailored to the specific resource constraints of each project environment. Several extensions merit investigation. Methodologically, event-based formulations for variable resources and heuristic approaches for larger instances could enhance computational tractability. The modeling framework could be enriched through multi-mode and multi-skill generalizations, as well as preemption models that account for forgetting effects. Theoretical analysis of makespan bounds under limited preemptions would complement these practical extensions, while a unified conceptual framework remains necessary to systematically address the variety of preemptive settings arising in applications.

Part III addresses scheduling problems arising in the context of machines subject to failures. Two main class of problems were presented, along with their variants: the Unreliable Job Selection and Sequencing Problem (UJSSP, Chapter 5) and the Unrecoverable Interruption Machine Scheduling Problem with Job Selection (UIMSPS, Chapter 6). In the UJSSP, given a set of jobs, a subset must be selected for processing on a single machine. Each job incurs a selection cost and yields a reward upon successful completion, with job-specific probabilities of success determining the expected outcomes. In the UIMSPS, each job requires a certain processing time, and the machine failure process follows the linear-risk model, according to which the probability of machine failure is uniformly distributed across the planning horizon. For both problems and their variants, we conducted a comprehensive computational complexity analysis, establishing theoretical complexity bounds and characterizing the tractability of different problem configurations, highlighting open problems.

Bibliography

[1] E. Lee, J.-Y. Oh and E. Pines, "Practical managerial decision making tools: Operations research," *The Journal of Applied Business and Economics*, vol. 8, no. 2, p. 11, 2008.

[2] G. Seaman, "47 - Industrial/Management Engineering in Health Care," in *Clinical Engineering Handbook*, ser. Biomedical Engineering, J. F. Dyro, Ed., Burlington: Academic Press, 2004, pp. 181–188.

[3] D. N. Kleinmuntz, "Resource Allocation Decisions," in *Advances in Decision Analysis: From Foundations to Applications*, W. Edwards, R. F. Miles Jr. and D. von Winterfeldt, Eds. Cambridge University Press, 2007, pp. 400–418.

[4] M. L. Pinedo, *Scheduling*. Springer Cham, 2022.

[5] A. Rais and A. Viana, "Operations Research in Healthcare: a survey," *International Transactions in Operational Research*, vol. 18, no. 1, pp. 1–31, 2010.

[6] J. Beliën, S. Brailsford, E. Demeulemeester, D. Demirtas, E. W. Hans and P. Harper, "Fifty years of operational research applied to healthcare," *European Journal of Operational Research*, vol. 326, no. 2, pp. 189–206, 2025.

[7] M. Al Amin, R. Baldacci and V. Kayvanfar, "A comprehensive review on operating room scheduling and optimization," *Operational Research*, vol. 25, no. 1, 2025.

[8] M. Samudra, C. Van Riet, E. Demeulemeester, B. Cardoen, N. Vansteenkiste and F. E. Rademakers, "Scheduling operating rooms: achievements, challenges and pitfalls," *Journal of Scheduling*, vol. 19, no. 5, pp. 493–525, 2016.

[9] B. Cardoen and E. Demeulemeester, "A decision support system for surgery sequencing at UZ Leuven's day-care department," *International Journal of Information Technology and Decision Making*, vol. 10, pp. 435–450, 2011.

[10] W. S. Ross and E. G. Canacari, "Surgical Scheduling: A Lean Approach to Process Improvement," *AORN Journal*, vol. 99, pp. 147–159, 2014.

[11] D. Johnston, A. Diamanti and F. Quereshi, "Why do surgeons schedule their own surgeries?" *Journal of Operations Management*, pp. 1–20, 2019.

[12] E. W. Hans, M. van Houdenhoven and P. J. Hulshof, "A Framework for Healthcare Planning and Control," in *Handbook of Healthcare System Scheduling*, R. Hall, Ed. Boston, MA: Springer US, 2012, pp. 303–320.

[13] F. Guerriero and R. Guido, "Operational research in the management of the operating theatre: a survey," *Health Care Management Science*, vol. 14, no. 1, pp. 89–114, 2011.

[14] J. T. Blake, F. Dexter and J. Donald, "Operating Room Managers' Use of Integer Programming for Assigning Block Time to Surgical Groups: A Case Study," *Anesthesia & Analgesia*, vol. 94, no. 1, pp. 143–148, 2002.

[15] S. Zhu, W. Fan, S. Yang, J. Pei and P. Pardalos, "Operating room planning and surgical case scheduling: a review of literature," *Journal of Combinatorial Optimization*, vol. 37, no. 3, pp. 757–805, 2019.

[16] I. E. Santos de Melo and F. S. Fogliatto, "Integration of decision levels in operating room scheduling problems: Systematic review and proposition of a decision support framework," *Computers & Operations Research*, 2025.

[17] L. Wang, E. Demeulemeester, N. Vansteenkiste and F. E. Rademakers, "Operating room planning and scheduling for outpatients and inpatients: A review and future research," *Operations Research for Health Care*, vol. 31, p. 100 323, 2021.

[18] D. R. Knight et al., "Artificial intelligence for patient scheduling in the real-world health care setting: A metanarrative review," *Health Policy and Technology*, vol. 12, no. 4, p. 100 824, 2023.

[19] V. Bellini, M. Russo, T. Domenichetti, M. Panizzi, S. Allai and E. G. Bignami, "Artificial Intelligence in Operating Room Management," *Journal of Medical Systems*, vol. 48, no. 1, p. 19, 2024.

[20] A. Pasquer, S. Ducarroz, J. C. Lifante, S. Skinner, G. Poncet and A. Duclos, "Operating room organization and surgical performance: a systematic review," *Patient Safety in Surgery*, vol. 18, no. 1, p. 5, 2024.

[21] F. Dexter and R. H. Epstein, "Fundamentals of operating room allocation and case scheduling to minimize the inefficiency of use of the time," *Perioperative Care and Operating Room Management*, vol. 35, p. 100 379, 2024.

[22] D. Salvatore, D. Numerato and G. Fattore, "Physicians' professional autonomy and their organizational identification with their hospital," *BMC Health Services Research*, vol. 18, p. 775, 2018.

[23] N. Meskens, D. Duvivier and A. Hanset, "Multi-objective operating room scheduling considering desiderata of the surgical team," *Decision Support Systems*, vol. 55, pp. 650–659, 2018.

[24] Y. Yang, B. Shen, W. Gao, Y. Liu and L. Zhong, "A surgical scheduling method considering surgeons' preferences," *Journal of Combinatorial Optimization*, vol. 30, pp. 1016–1026, 2015.

[25] J. Park, B.-I. Kim, M. Eom and B. K. Choi, "Operating room scheduling considering surgeons' preferences and cooperative operations," *Computers and Industrial Engineering*, vol. 157, p. 107 306, 2021.

[26] I. Marques and M. E. Captivo, "Different stakeholders' perspectives for a surgical case assignment problem: Deterministic and robust approaches," *European Journal of Operational Research*, vol. 261, no. 1, pp. 260–278, 2017.

[27] J. M. H. Vissers, I. J. B. F. Adan and J. A. Bekkers, "Patient mix optimization in tactical cardiothoracic surgery planning: a case study," *IMA Journal of Management Mathematics*, vol. 16, no. 3, Jul. 2005.

[28] P. Cappanera, F. Visintin and C. Banditori, "Comparing resource balancing criteria in master surgical scheduling: A combined optimisation-simulation approach," *International Journal of Production Economics*, vol. 158, pp. 179–196, 2014.

[29] X. Wu et al., "Top 10 algorithms in data mining," *Knowledge and Information Systems*, 2008.

[30] "Ministero della Salute," Accessed: 28th Dec. 2025.

[31] Y. Ma et al., "Telemedicine application in patients with chronic disease: a systematic review and meta-analysis," *BMC Medical Informatics and Decision Making*, 2022.

[32] V. Ezeamii et al., "Revolutionizing Healthcare: How Telemedicine Is Improving Patient Outcomes and Expanding Access to Care," *Cureus*, 2024.

[33] M. Doneda et al., "A decision support tool for the location, districting and dimensioning of Community Health Houses," *Health Care Management Science*, 2025.

[34] D. J. Weiss et al., "Global maps of travel time to healthcare facilities," *Nature Medicine*, vol. 26, no. 12, 2020.

[35] J. Pan, Y. Deng, Y. Yang and Y. Zhang, "Location-allocation modelling for rational health planning: Applying a two-step optimization approach to evaluate the spatial accessibility improvement of newly added tertiary hospitals in a metropolitan city of China," *Social Science & Medicine*, vol. 338, p. 116 296, 2023.

[36] A. M. Mestre, M. D. Oliveira and A. P. Barbosa-Póvoa, "Location–allocation approaches for hospital network planning under uncertainty," *European Journal of Operational Research*, vol. 240, no. 3, pp. 791–806, 2015.

[37] M. Acar and O. Kaya, "A healthcare network design model with mobile hospitals for disaster preparedness: A case study for Istanbul earthquake," *Transportation Research Part E: Logistics and Transportation Review*, vol. 130, pp. 273–292, 2019.

[38] W. Gu, X. Wang and S. E. McGregor, "Optimization of preventive health care facility locations," *International Journal of Health Geographics*, vol. 9, 1 2010.

[39] J. Teixeira, A. Antunes and D. Peeters, "An Optimization-Based Study on the Redeployment of a Secondary School Network," *Environment and Planning B: Planning and Design*, vol. 34, no. 2, pp. 296–315, 2007.

[40] Z. Wang, Y. Wang, Y. Qiu and C. Jin, "Strategic Resource Allocation in Dual-Channel Healthcare Systems: The Role of Telemedicine in Physician Assignment," *IEEE Transactions on Automation Science and Engineering*, vol. 22, pp. 13 569–13 587, 2025.

[41] C. Stolarchuk et al., "Optimizing Healthcare Delivery: Strategies for Workforce Retention and Resource Allocation," *Journal of Surgical Specialties and Rural Practice*, vol. 6, 1 2025.

[42] A. Vanckavičienė, A. Blaževičienė, D. Zagurskienė and K. Stašaitis, "Health workforce forecast in the university hospital: evidence from Lithuania," *BMC Nurs*, vol. 23, 1 2024.

[43] D. Bonfim, A. C. C. N. Mafra, D. da Costa Palacio and T. Rewa, "Assessment of staffing needs for registered nurses and licensed practical nurses at primary care units in Brazil using Workload Indicators of Staffing Need (WISN) method," *Human Resources for Health*, vol. 19, p. 130, 1 2022.

[44] M. Peršolja, "The effect of nurse staffing patterns on patient satisfaction and needs: a cross-sectional study," *Journal of nursing management*, vol. 26, 7 2018.

[45] ISTAT. "Resident Population on 1 January – Demo ISTAT," Accessed: 28th Nov. 2025. [Online]. Available: `https://demo.istat.it/app/?l=en&a=2023&i=POS`

[46] ARS. "Population density on 1 January," Accessed: 28th Nov. 2025. [Online]. Available: `https://www.ars.toscana.it/banche-dati/dettaglio_indicatore-1452-densita-abitativa-numero-residenti-km-quadrato?provenienza=aziendale_capitoli&par_top_geografia=203&dettaglio=ric_anno_geo_ausl`

[47] ARS. "Demographic indicators on 1 January," Accessed: 28th Nov. 2025. [Online]. Available: `https://demo.istat.it/tavole/?t=indicatori&l=it`

[48] Istituto Superiore di Sanità. "Epicentro passi – patologie croniche," Accessed: 26th Nov. 2025. [Online]. Available: `https://www.epicentro.iss.it/passi/dati/croniche`

[49] ARS. "Proter–Macro: Territorial Surveillance Program – Chronic Diseases," Accessed: 13th Dec. 2025. [Online]. Available: `https://visual.ars.toscana.it/proter_macro`

[50] L. E. Drezet and J. C. Billaut, "A project scheduling problem with labour constraints and time-dependent activities requirements," *International Journal of Production Economics*, vol. 112, no. 1, pp. 217–225, 2008.

[51] R. Klein, "Project scheduling with time-varying resource constraints," *International Journal of Production Research*, vol. 38, no. 16, pp. 3937–3952, 2000.

[52] S. Hartmann, "Time-Varying Resource Requirements and Capacities," in *Handbook on project management and scheduling vol.1*, C. Schwindt and J. Zimmermann, Eds., Berlin: Springer, 2015, pp. 163–176.

[53] S. Hartmann and D. Briskorn, "An updated survey of variants and extensions of the resource-constrained project scheduling problem," *European Journal of Operational Research*, vol. 297, pp. 1–14, 2022.

[54] E. L. Demeulemeester and W. S. Herroelen, "An efficient optimal solution procedure for the preemptive resource-constrained project scheduling problem," *European Journal of Operational Research*, vol. 90, pp. 334–348, 1996.

[55] J. H. Patterson, "A Comparison of Exact Approaches for Solving the Multiple Constrained Resource, Project Scheduling Problem," *Management Science*, vol. 30, no. 7, pp. 854–867, 1984.

[56] A. Moukrim, A. Quilliot and H. Toussaint, "An effective branch-and-price algorithm for the Preemptive Resource Constrained Project Scheduling Problem based on minimal Interval Order Enumeration," *European Journal of Operational Research*, vol. 244, pp. 360–368, 2015.

[57] J. Zhu, X. Li and W. Shen, "Effective genetic algorithm for resource-constrained project scheduling with limited preemptions," *International Journal of Machine Learning and Cybernetics*, vol. 2, pp. 55–65, 2011.

[58] S. Quintanilla, P. Lino, Á. Pérez, F. Ballestín and V. Valls, "Integer preemption problems," in *Handbook on project management and scheduling vol.1*, C. Schwindt and J. Zimmermann, Eds., Berlin: Springer, 2015, pp. 231–250.

[59] V. Valls, F. Ballestín and S. Quintanilla, "A hybrid genetic algorithm for the resource-constrained project scheduling problem," *European Journal of Operational Research*, vol. 85, no. 2, pp. 495–508, 2008.

[60] Z. Ma, M. Ning and Y. Wang, "Proactive Project Scheduling With Activity Splitting and Resource Transfer Times Under Uncertain Environments," *IEEE Access*, vol. 10, pp. 87 490–87 499, 2022.

[61] Y. Shu, Y. Li and C. Lai, "Hybrid particle swarm optimization for preemptive resource-constrained project scheduling," *Neurocomputing*, vol. 148, pp. 122–128, 2015.

[62] B. Afshar-Nadjafi and M. Majlesi, "Resource constrained project scheduling problem with setup times after preemptive processes," *Computers and Chemical Engineering*, vol. 69, pp. 16–25, 2014.

[63] M. Vanhoucke and J. S. Coelho, "Resource-constrained project scheduling with activity splitting and setup times," *Computers and Operations Research*, vol. 109, pp. 230–249, 2019.

[64] S. Kreter, A. Schutt and P. J. Stuckey, "Modeling and solving project scheduling with calendars," in *Principles and practice of constraint programming*, G. Pesant, Ed., Berlin: Springer, 2015, pp. 262–278.

[65] B. Afshar-Nadjafi, "A solution procedure for preemptive multi-mode project scheduling problem with mode changeability to resumption," *Applied Computing and Informatics*, vol. 14, no. 2, pp. 192–201, 2018.

[66] J. Damay, A. Quillot and E. Sanlaville, "Linear programming based algorithms for preemptive and non-preemptive RCPSP," *European Journal of Operational Research*, vol. 182, pp. 1012–1022, 2007.

[67] C. Schwindt and T. Paetz, "Continuous Preemption Problems," in *Handbook on Project Management and Scheduling Vol.1. International Handbooks on Information Systems*, C. Schwindt and J. Zimmermann, Eds., Berlin: Springer, 2015, pp. 251–295.

[68] C. Le Pape and P. Baptiste, "Resource constraints for preemptive job-shop scheduling," *Constraints*, vol. 3, no. 4, pp. 263–287, 1998.

[69] J. Buddhakulsomsiri and D. S. Kim, "Properties of multi-mode resource-constrained project scheduling problems with resource vacations and activity splitting," *European Journal of Operational Research*, vol. 175, pp. 279–295, 2006.

[70] J. Cheng, J. Fowler, K. Kempf and S. Mason, "Multi-mode resource-constrained project scheduling problems with non-preemptive activity splitting," *Computers and Operations Research*, vol. 53, pp. 275–287, 2015.

[71] R. Van Eynde, M. Vanhoucke and J. Coelho, "On the summary measures for the resource-constrained project scheduling problem," *Annals of Operations Research*, vol. 337, pp. 593–625, 2024.

[72] V. T'Kindt and J. C. Billaut, *Multicriteria scheduling: theory, models and algorithms*, second. Springer, 2006.

[73] L. A. Kaplan, *Resource-constrained project scheduling with preemption of jobs*, Doctoral dissertation, University of Michigan. University of Michigan Library, 1988.

[74] R. Kolisch and A. Sprecher, "PSPLIB - A project scheduling problem library," *European Journal of Operational Research*, vol. 96, no. 1, pp. 205–216, 1997.

[75] O. Koné, C. Artigues, P. Lopez and M. Mongeau, "Event-based MILP models for resource-constrained project scheduling problems," *Computers and Operations Research*, vol. 38, pp. 3–13, 2011.

[76] D. F. Cooper, "Heuristics for Scheduling Resource-Constrained Projects: An Experimental Investigation," *Management Science*, vol. 22, no. 11, pp. 1186–1194, 1976.

[77] R. Kolisch, A. Sprecher and A. Drexl, "Characterization and Generation of a General Class of Resource-Constrained Project Scheduling Problems," *Management Science*, vol. 41, no. 10, pp. 1693–1703, 1995.

[78] F. Ballestín, V. Valls and S. Quintanilla, "Scheduling projects with limited number of preemptions," *Computers and Operations Research*, vol. 36, pp. 2913–2925, 2009.

[79] C. K. Chang, H. Jiang, Y. Di, D. Zhu and Y. Ge, "Time-line based model for software project scheduling with genetic algorithms," *Information and Software Technology*, vol. 50, pp. 1142–1154, 2008.

[80] R. C. Ash, "A time-based learning model for resource constrained development projects," *American Business Review*, vol. 18, no. 1, pp. 69–76, 2000.

[81] G. Schmidt, "Scheduling with limited machine availability," *European Journal of Operational Research*, vol. 121, no. 1, pp. 1–15, 2000.

[82] I. Adiri, J. Bruno, E. Frostig and A. Rinnooy Kan, "Single Machine Flow-Time Scheduling With a Single Breakdown," *Acta Informatica*, vol. 26, no. 7, pp. 679–696, 1989.

[83] Y. Yin, Y. Wang, T. Cheng, W. Liu and J. Li, "Parallel-machine scheduling of deteriorating jobs with potential machine disruptions," *Omega*, vol. 69, pp. 17–28, 2017.

[84] A. Benoit, M. Hakem and Y. Robert, "Contention awareness and fault-tolerant scheduling for precedence constrained tasks in heterogeneous systems," *Parallel Computing*, vol. 35, no. 2, pp. 83–108, 2009.

[85] J. G. Szmerekovsky, P. Venkateshan and P. D. Simonson, "Project scheduling under the threat of catastrophic disruption," *European Journal of Operational Research*, vol. 309, no. 2, pp. 784–794, 2023.

[86] T. Unluyurt, "Sequential testing problem: a follow-up review," *Discrete Applied Mathematics*, vol. 377, pp. 356–369, 2025.

[87] Z. Li, V. Chang, H. Hu, C. Li and J. Ge, "Real-time and dynamic fault-tolerant scheduling for scientific workflows in clouds," *Information Sciences*, vol. 568, pp. 13–39, 2021.

[88] A. Benoit, Y. Robert, A. L. Rosenberg and F. Vivien, "Static Strategies for Worksharing with Unrecoverable Interruptions," *Theory of Computing Systems*, vol. 53, pp. 386–423, 2013.

[89] J. Kadane, "Quiz show problems," *Journal of Mathematical Analysis and Applications*, vol. 27, no. 3, pp. 609–623, 1969.

[90] W. Stadje, "Selecting jobs for scheduling on a machine subject to failure," *Discrete Applied Mathematics*, vol. 63, no. 3, pp. 257–265, 1995.

[91] J. Hellerstein and M. Stonebraker, "Predicate migration: Optimizing queries with expensive predicates," in *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, New York, NY, USA: ACM, 1993, pp. 267–276.

[92] A. Agnetis, P. Detti, M. Pranzo and M. Sodhi, "Sequencing unreliable jobs on parallel machines," *Journal of Scheduling*, vol. 12, no. 1, pp. 45–54, 2009.

[93] L. Mitten, "An analytic solution to the least cost testing sequence problem," *Journal of Industrial Engineering*, vol. 11, no. 1, p. 17, 1960.

[94] T. Ünlüyurt, "Sequential testing of complex systems: A review," *Discrete Applied Mathematics*, vol. 142, no. 1–3, pp. 189–205, 2004.

[95] K. Glazebrook, "Scheduling stochastic jobs on a single machine subject to breakdowns," *Naval Research Logistics Quarterly*, vol. 31, no. 2, pp. 251–264, 1984.

[96] E. Frostig, "A Note on Stochastic Scheduling on a Single Machine Subject to Breakdown–The Preemptive Repeat Model," *Probability in the Engineering and Informational Sciences*, vol. 5, no. 3, pp. 349–354, 1991.

[97] X. Qi, J. Bard and G. Yu, "Disruption management for machine scheduling: The case of SPT schedules," *International Journal of Production Economics*, vol. 103, pp. 166–184, 2006.

[98] J. Birge, J. Frenk, J. Mittenthal and A. Rinnooy Kan, "Single-Machine Scheduling Subject to Stochastic Breakdowns," *Naval Research Logistics*, vol. 37, pp. 661–677, 1990.

[99] A. Agnetis, M. Benini, P. Detti, B. Hermans and M. Pranzo, "Replication and sequencing of unreliable jobs on parallel machines," *Computers & Operations Research*, vol. 139, p. 105 634, 2022.

[100] W. Olszewski and R. Vohra, "Simultaneous selection," *Discrete Applied Mathematics*, vol. 200, pp. 161–169, 2016.

[101] H. Chade and L. Smith, "Simultaneous Search," *Econometrica*, vol. 74, no. 5, pp. 1293–1307, 2006.

[102] U. Feige and J. Vondrák, "The submodular welfare problem with demand queries," *Theory of Computing*, vol. 6, pp. 247–290, 2010.

[103] C. Ng, M. Barketau, T. Cheng and M. Kovalyov, ""Product Partition" and related problems of scheduling and systems reliability: Computational complexity and approximation," *European Journal of Operational Research*, vol. 207, no. 2, pp. 601–604, 2010.

[104] A. Agnetis, R. Leus, E. Perneel and I. Salvadori. "The unreliable job selection and sequencing problem." arXiv: `2511.17105 [cs.DM]`.

[105] U. Pferschy, J. Schauer and C. Thielen. "The product knapsack problem: approximation and complexity." arXiv: `1901.00695 [math.OC]`.

[106] A. Agnetis and T. Lidbetter, "The largest-Z-ratio-first algorithm is 0.8531-approximate for scheduling unreliable jobs on m parallel machines," *Operations Research Letter*, vol. 48, no. 4, pp. 405–409, 2020.

[107] A. Agnetis, P. Detti and P. Martineau, "Scheduling nonpreemptive jobs on parallel machines subject to exponential unrecoverable interruptions," *Computers & Operations Research*, vol. 79, pp. 109–118, 2017.

[108] W. Smith, "Various optimizers for single stage production," *Naval Research Logistics Quarterly*, vol. 3, pp. 59–66, 1956.

[109] D. Shabtay, N. Gaspar and M. Kaspi, "A survey on offline scheduling with rejection," *Journal of Scheduling*, vol. 16, pp. 3–28, 2013.

[110] D. Schmidt, "Scheduling with position-dependent speed," M.S. thesis, Technische Universität München, Fakultät für Mathematik, 2017.