

Daniel Clavel Villagrasa

Methods and Formal Models for Healthcare Systems Management

Departamento
Informática e Ingeniería de Sistemas

Director/es
Silva Suárez, Manuel
Mahulea, Cristian Florentin

<http://zaguan.unizar.es/collection/Tesis>



Reconocimiento – NoComercial – SinObraDerivada (by-nc-nd): No se permite un uso comercial de la obra original ni la generación de obras derivadas.

© Universidad de Zaragoza
Servicio de Publicaciones

ISSN 2254-7606



Universidad
Zaragoza

Tesis Doctoral

**METHODS AND FORMAL MODELS FOR
HEALTHCARE SYSTEMS MANAGEMENT**

Autor

Daniel Clavel Villagrasa

Director/es

Silva Suárez, Manuel
Mahulea, Cristian Florentin

UNIVERSIDAD DE ZARAGOZA
Informática e Ingeniería de Sistemas

2019

Departamento de Informática e Ingeniería de Sistemas
Universidad de Zaragoza

PhD Thesis

Methods and Formal Models for Healthcare Systems Management

Daniel Clavel

Supervisors:

Cristian Mahulea
Manuel Silva Suárez

September 2019

To Ester

Agradecimientos

I would like to thank all the people who, during the last four years help me in many ways, making possible the present work.

I would like to thank, first and foremost, Manuel Silva and Cristian Mahulea, my Ph.D. advisors at the University of Zaragoza, Spain. This thesis is based on the work done together. I cannot remember how many meetings we had together that clarified many problems. I am very proud to work with two of the most important figures in the field of Petri nets.

Thanks a lot to Jorge Albareda, head of the Orthopedic Department at the “Lozano Blesa” Hospital with which I have collaborated in this work. Dear Jorge, I will miss the meetings about hospital management that ends up being medical stories worthy of a hospital TV shows.

Several people have played a decisive role in my work, and I want to mention here Xiaolan Xie, professors at the University of Saint Etienne, that have visited our Department, but also for their hospitality during my stay in their group, in Saint Etienne. Thanks a lot to Xiaolan Xie and Carla Seatzu for reading this thesis as international reviewers providing me good reports that help to improve the presentation and quality of the document.

I would like to thank all the people in the Department of Informatics and Systems Engineering at the University of Zaragoza. Professors, students, and administrative staff, for their selfless help during these four years. I would like to thank all the lab partners and especially those I have worked with. Luis Parilla thanks for being a person with a big heart always willing to help me. Diana Botez, thank you very much for receiving with a smile the continuous modifications to be implemented in CIPLAN tool. Thanks a lot also to Enmanuel

Vitolo, we don't work much together, but you are available every "Juepincho" to go for "tapas". Thanks a lot to Eduardo Moya, one of the most intelligent people I have ever met. We studied together the degree and the master and now we share Ph.D. laboratory and coffee breaks every morning. Eduardo, thank you for your useful advice during these 9 years.

I want to thank the institution that provided me financial support during this thesis: a Ph.D. grant from Spanish Ministry of Science and Education (reference BES-2015-074914) and the projects CICYT - FEDER DPI2014-57252-R from the Spanish Ministry of Science and Education.

On a personal level, I would like to thank all my friends and family for their support. Grandparents, uncles, and cousins of families Clavel and Villagrasa, very thanks for your love. To my friends of the "Peña 0,0" in Bujaraloz in Bujaraloz thanks for being there every weekend.

I would like to especially thank my parents, Cesar and Raquel, for teaching me that every effort has its reward, but above all for helping me to be a happy person. Cesar, besides my brother, you are a life partner, thank you for the holidays, tennis matches and parties that we have spent together because it has been a special way to disconnect of the thesis. Many thanks to my lovely couple, Elena, for her patience, understanding, and solidarity with this project, for the time she has granted me, or somehow I have robbed her. I want to mention here my grandmother, Ester. A kind person that wherever she is, I am sure she will be very proud of her grandson. I wish to dedicate this thesis to her.

Methods and Formal Models for Healthcare Systems Management.

Summary

A health system is the sum of all organizations, institutions and resources whose main objective is to improve health. A health system needs personnel, financing, information, supplies, transportation and communications, as well as general guidance and direction. It also has to provide good treatments and services that respond to the needs of the population and are financially fair. So, the management of healthcare systems is a complex task due to its size, the huge number of agents involved and their different expectations. In this work methods and formal models for healthcare systems management are proposed. Particularly we focus on the management of hospitals. The first part of the thesis deals with the modeling and analysis of healthcare systems using Petri Nets. This formalism allows a mathematical or graphic representation of a discrete event system in which the topology of a distributed, parallel or concurrent system can be described. Formal subclasses of Petri Nets (S^4PR and DSSP) in which structural techniques can be applied to analyze different properties of the system are considered. Particularly, resource management facet is modeled by S^4PR while handshake between clinical pathways facet is modeled by DSSP. Moreover, two different techniques for liveness enforcement in DSSP are proposed. The first one, based on buffers preassignment, is easy to apply, however, it only works in some DSSP structures. The second liveness method constructs a control PN that avoids deadlock markings in the original

DSSP. This method works in general DSSP structures. In the second part of the thesis, the surgery scheduling issue is approached. The huge size of the surgery waiting lists is becoming a serious problem for healthcare administrations. So, in this work, taking into account the scheduling criteria in Spanish hospitals, different mathematical programming problems and heuristics solutions for surgery scheduling are proposed and compared. Moreover, a decision support system for the management of a surgical service developed in a software tool is given. A real case study using the tool concludes that around more than 70 additionally patients per year could be operated only considering the Orthopedic Surgery Department.

Contents

Introduction	1
I Modeling and analysis in hospitals	9
1 Preliminary: Petri Nets and previous results	11
1.1 Petri Net	12
1.1.1 Basic concepts	12
1.1.2 Structural concepts	13
1.1.3 Conflicts and structural conflict relations	14
1.1.4 Liveness and deadlock-freeness	14
1.1.5 Subclasses of Petri nets	14
1.1.6 Rank theorem in DSSP	17
1.2 Previous work: clinical pathways assessment	18
1.2.1 Stochastic Well-formed Net	20
1.3 Discussion	25
2 From HSS to formal models: S4PR and DSSP	27
2.1 Motivation	28
2.1.1 Clinical pathways	28
2.1.2 Starting point	28
2.1.3 Facets of the healthcare system	29
2.2 Modifications to get formal models	29
2.2.1 Resource management facet	30

2.2.2	Handshake between clinical pathways facet	34
2.3	Case application	36
2.4	Discussion	40
3	Pre-assignment method for liveness of DSSP	41
3.1	Starting point	42
3.1.1	Introduction	42
3.1.2	Controlling bad siphons in DSSP systems	43
3.1.3	The idea behind of buffer pre-assignment method	44
3.2	Buffer pre-assignment approach	46
3.2.1	Definitions	47
3.2.2	Local vs. global pre-assignment perspective	49
3.2.3	Buffer pre-assignment algorithm	51
3.2.4	Limitations of the approach	54
3.3	Discussion	58
4	DSSP liveness enforcement by a control PN	61
4.1	Introduction	62
4.1.1	A two level approach for DSSP liveness enforcement	62
4.1.2	How limitation of pre-assignment method are approached	64
4.1.3	Overview of the proposed approach	67
4.2	Construction of the Control PN	68
4.3	Liveness of the Control PN	72
4.3.1	Is it liveness guaranteed in the Control PN?	73
4.3.2	Liveness enforcement of the Control PN	75
4.4	Control policy and system evolution	82
4.5	Improving permissiveness & working with a unique PN	87
4.5.1	Improving the permissibility	87
4.5.2	Composition of DSSP system and control PN system	88
4.6	Discussion	91
II	Surgical planning and scheduling	93
5	Surgery scheduling under OR block booking	95
5.1	Background	96

5.1.1	Introduction	96
5.1.2	Scheduling Criteria	98
5.1.3	Problem Statement	100
5.2	Surgery scheduling imposing an occupation rate	101
5.2.1	Scheduling problem modeled as a QAP	101
5.2.2	Scheduling problem modeled by a MILP	104
5.2.3	Scheduling problem modeling by a GAP	105
5.2.4	Heuristic methods	107
5.2.5	Simulation results	109
5.3	Surgery scheduling under uncertain durations	118
5.3.1	Related work	120
5.3.2	Surgery scheduling under uncertain durations	122
5.3.3	Heuristics approaches	125
5.3.4	Simulation results	130
5.4	Discussion	136
6	Scheduling of elective and urgent patients	139
6.1	Introduction	140
6.2	Problem Statement	142
6.3	Scheduling and sequencing of patients	146
6.3.1	Scheduling elective patients using MILP problem	148
6.3.2	Scheduling urgent patients	148
6.3.3	Sequencing the surgeries	150
6.4	Simulation Approach	152
6.4.1	Patients generation and scheduling	153
6.4.2	Surgery activity simulation in a OR	154
6.5	Numerical Results	156
6.6	Discussion	158
7	Decision Support System and Software tool	161
7.1	Decision Support System	162
7.1.1	Manage medical team and OR time-table	163
7.1.2	Updating the waiting list	164
7.1.3	Iterative planning	164
7.1.4	Updating and customizing the averages durations	165
7.1.5	Overview of the DSS	165

7.2	A software tool for surgery scheduling	167
7.2.1	Features of CIPLAN	168
7.2.2	Back end implementation	168
7.2.3	Database	171
7.2.4	User interface	173
7.3	Real case application using CIPLAN	185
7.3.1	Hospital Description	185
7.3.2	Methodology of the case study	186
7.3.3	Comparing the manual scheduling method with CIPLAN	189
7.4	Discussion	197
	Conclusion	199
	Bibliography	203
	A Scheduling case application	213

List of Figures

1.1	A S^4PR net example.	16
1.2	Non live DSSP net.	18
1.3	Care example modeled by HSS profile and transformed to SWN	23
2.1	Relaxations abstractions and modifications	31
2.2	Order in which rules R8 and R9 must be applied	34
2.3	UML model of two clinical pathways	37
2.4	Resource management facet of the UML model in Fig. 2.3	38
2.5	Handshake between clinical pathway facet of the UML in Fig. 2.3	39
3.1	A not live DSSP net with its monitor places	44
3.2	Buffer pre-assignment in a non live DSSP net.	45
3.3	Liveness check in DSSP.	47
3.4	Pre-assignment from a local perspective vs. from a global one	50
3.5	Pre-assignment method applied to net in Fig. 2.5	55
3.6	Pre-assignment method applied to net in Fig. 2.5	57
4.1	Motivation example of problem 1	65
4.2	Motivation example of problem 2	66
4.3	Overview of the liveness enforcement methodology	67
4.4	Agent without waiting place	69
4.5	Control PN obtained from DSSP structure in Fig. 4.1(a)	72
4.6	Diagram for ensuring liveness in the Control PN	73
4.7	Simplified control PN of net in Fig. 4.5	74
4.8	Adding new buffers in a control PN	77

4.9	A non live subnet of a simplified Control PN plus control places . . .	79
4.10	Virtual check transition t_{ch} of the set of check transition $\{t_1, t_2, t_3\}$.	81
4.11	Control diagram of a DSSP structure using a control PN	83
4.12	Example of the control policy	86
4.13	Advance of token production in the Control PN	88
4.14	Equivalent controlled net (\mathcal{N}^e) of DSSP in Fig. 4.1(a)	90
5.1	Receding horizon Strategy	108
6.1	Operation Room time distribution	146
6.2	Flowchart overview of the 3 Step approach	147
6.3	BIM problem terminology	151
7.1	Organizational structure in the studied hospital departments	163
7.2	Flowchart of the DSS	166
7.3	Welcome window of CIPLAN	169
7.4	Login window for CIPLAN	169
7.5	Database structure for CIPLAN	171
7.6	Medical teams panel for CIPLAN	174
7.7	Medical teams management subpanel for CIPLAN	174
7.8	Medical staff management for CIPLAN	175
7.9	Patient list subpanel in CIPLAN	176
7.10	Search patient in CIPLAN	177
7.11	Add new patient in CIPLAN	177
7.12	Update patinets information in CIPLAN	178
7.13	Surgery panel in CIPLAN	179
7.14	OR timetable panel in CIPLAN	180
7.15	Create scheduling in CIPLAN	181
7.16	See scheduling in CIPLAN	182
7.17	Set a surgery as completed in CIPLAN	183
7.18	Create a new user in CIPLAN	183
7.19	Delete a existing user in CIPLAN	184
7.20	Modify access information for a existing user in CIPLAN	184
7.21	Logout in CIPLAN	185
7.22	Occurrence of surgeries in the OSD of the LBH	187
7.23	Expected occupation rate: Manual VS CIPLAN	190

7.24	Real occupation rate: Manual VS CIPLAN	191
7.25	Confidence level of not overtime: Manual VS CIPLAN	192
7.26	Ending time of the blocks: manual VS CIPLAN	193
7.27	Manual scheduling of the blocks	195
7.28	Automatic scheduling of the blocks	196
A.1	Waiting list: patients from 1-150	214
A.2	Waiting list: patients from 151-300	215
A.3	Waiting list: patients from 301-397	216
A.4	Manual scheduling: blocks 1-50	217
A.5	Automatic scheduling: blocks 1-50	218
A.6	Manual scheduling: blocks 51-100	219
A.7	Automatic scheduling: blocks 51-100	220
A.8	Manual scheduling: blocks 101-150	221
A.9	Automatic scheduling: blocks 101-150	222

List of Tables

3.1	Changes in the net in Fig. 3.1 (a) to obtain the net in Fig. 3.1 (b).	46
3.2	Local and Global T-semiflows of net in Fig 3.4	49
4.1	Local and Global T-semiflows of \mathcal{N}^d in Fig. 4.1(a)	64
5.1	Computational time to solve optimally the QAP (5.9)	110
5.2	Computational time to solve the QAP (5.9) using GA	111
5.3	Computational time to solve optimally the MILP	112
5.4	Computational time for solving optimally the GAP	113
5.5	Computational time to solve the GAP (5.13) using the SDMAM . .	114
5.6	Instances to evaluate	114
5.7	Comparing the RHS, SDMAM and Meta-Heuristic GA approaches	116
5.8	MILP: Statistic analysis of occupation rate depending of β	118
5.9	MILP: order of patients analysis depending of Beta	119
5.10	Exact solution for N-MIQCP problem ($ \mathcal{W} = 100$ and $ \mathcal{B} = 3$) . . .	131
5.11	Surgery scheduling problem solution: RHS vs HSA	133
5.12	Comparison of the SHA with different chain-constrained approaches	135
5.13	Analysis of the SHA approach with different confidence level Cl . .	136
6.1	Master Surgery Schedule in the OSD of the LBH	145
6.2	Analysis of different strategies in the OSD of the LBH	156
7.1	OR time-table in the Orthopedic Surgical Department of the HCU	186
7.2	Duration of the surgeries more performed in the OSD of the LBH .	188

Introduction

A healthcare system is an organization of people, institutions, and resources that deliver healthcare services to meet the health needs of target populations. The management of any healthcare system is typically directed through a set of policies and plans adopted by governments, private sector business and other groups in areas such as personal healthcare delivery and financing, pharmaceuticals, health human resources, and public health.

The size of the systems, the huge number of agents involved and their different expectations make the management of healthcare systems a thorough task which could be alleviated through the use of technology. Technology is everywhere. We all use it. Even if we just wash clothes, play video-games, drive the car, or construct robots to help us. We can find the latest available technology starting from smart-phones to smart sidewalks or from smart cars to smart homes. Everything is technologized nowadays so, why not use this power to help in the management of healthcare systems?

Necessary tools for proper health information coding and management include clinical guidelines, formal medical terminologies, and computers and other information and communication technologies. The kinds of health data processed may include patients' medical records, hospital administration and clinical functions, and human resources information.

In this thesis, new methods and formal models for healthcare system management are considered. These new methods will help to prevent system failures and on the other hand, to improve the quality and efficiency of the hospitals. Particularly, the thesis is divided in two main parts: the first one has to do with the *modeling and analysis in hospitals* by the use of clinical pathways

while the second one deals with the *planning and scheduling of patients* in the operation rooms.

Regarding the modeling and analysis of healthcare systems, in [6] was proposed a domain-specific modeling language called Healthcare System Specifications (HSS), which is defined as a Unified Modeling Language (UML) profile. The proposed HSS-UML profile allows specifying the hospital description through the development of clinical pathways used for the treatment of different medical pathologies. Moreover, in order to be able to perform an analysis of the system, a model to model (M2M) transformation was given (recalled in Sec. 1.2.1). This transformation obtains a colored Petri Net model, particularly a Stochastic Well-formed Net (SWN) [12].

Since healthcare systems are complex due to their size and structure, depending on different visions and expectations, the system can be treated from different perspectives called facets. In chapter 2 two facets are identified: (1) facet of resource management and (2) handshake between clinical pathways facet. They are obtained by applying to the SWN a set of relaxations, abstractions and modifications (given in Sec. 2.2). In the first facet (Sec. 2.2.1), the subclass of S^4PR [68] is obtained which is a characteristic model of the resource allocation systems (state machines interconnected through shared resources modeling the flow of patients) while in the second facet (Sec. 2.2.2) Deterministically Synchronized Sequential Process (DSSP) [56] are considered (state machines modeling the clinical pathways which are interconnected by buffers that model the communication channels). Both nets (S^4PR and DSSP) are formal subclasses (Sec. 1.1.5) of Petri Nets.

Petri Nets constitute a family of formalisms appropriate for the modeling of discrete event systems (systems in which state variables take discrete values from a set that can be counted, not necessarily finite). Its distributed state is a vector of non-negative integers. This is an important advantage with respect to other forms of modeling of discrete systems, such as automata, where the state space is a set of unstructured symbols of global states. This property has been exploited to develop very diverse analysis techniques that do not require the enumeration of the state space (structural analysis).

The subclass of DSSP is commonly used to model distributed systems formed by various cooperative agents. This cooperation is carried out through a set of buffers from/to agents consume/produce items. In healthcare sys-

tems modeled by DSSP, the agents define different clinical pathways while the buffers represent the channel information. In chapters 3 and 4, we will focus on the liveness of the DSSP systems resulting from the facet of communication between clinical pathways. That is, to prevent any patient from being trapped in a medical protocol without the possibility of continuing.

In Sec. 3.2 the first approach for liveness enforcement in DSSP system is given. The idea behind the approach is to advance the buffer consumption to the first conflict transition in the agents. Considering healthcare systems modeled by a DSSP, this means that before a patient starts a clinical pathway, all required information must be available. Unfortunately, this preassignment method only works in some particular DSSP structures which are characterized. A more general approach (than buffer preassignment) for liveness enforcing in non-live DSSP is given in Sec. 4.1.3. The approach is formalized on two levels: execution and control. The execution level uses the original DSSP structure while for the control level we compute a new net system called the *control PN* (Sec. 4.2). This net system is obtained from the original DSSP and has a predefined type of structure. The control PN must be live (Sec. 4.3) and it will evolve synchronously with the non-live DSSP ensuring that the deadlock states will not be reached. The states (marking) of the control PN will enable or disable some transitions in the original DSSP, while some transitions in the control PN should fire synchronously with some transitions of the original DSSP. The control police and systems evolution is given in Sec. 4.4.

The second part of the thesis deals with surgery scheduling of patients in a hospital department. The Operating Rooms (ORs) are one of the most expensive material resources in hospitals, being the bottleneck of surgical services. Moreover, the aging population together with the improvement in surgical techniques are producing an increase in the demand for surgeries [28]. This increase also causes an increase in the time that patients must wait to be operated on. So, the optimal use of the ORs time is crucial in healthcare service management.

Managing the operating theater, however, is hard due to the conflicting priorities and the preferences of its stakeholders [33], but also due to the scarcity of costly resources. Moreover, health managers have to anticipate the increasing demand for surgical services caused by the aging population. These factors

clearly stress the need for efficiency and necessitate the development of adequate planning and scheduling procedures.

In the past 60 years, a large body of literature on the management of operating theaters has evolved [10]. In this thesis, we focus on the planning and scheduling of patients in Spanish hospitals departments considering its organizational structure particularities as well as the concerns and specifications of their doctors. The proposed approaches have been tested and compared considering realist data of the Orthopedic Surgery Department in the “Lozano Blesa” Hospital in Zaragoza.

In chapter 5 the scheduling of elective patients under ORs block booking is considered. Sec. 5.1.2 establishes the three scheduling criteria to consider for the scheduling while Sec. 5.1.3 gives the problem statement. The first criterion is to optimize the use of the OR, the second criterion is to prevent that the total available time in a block will be exceeded and the third criterion is to respect the preference order of the patient in the waiting list.

In Sec. 5.2 three different mathematical programming models for the scheduling of elective patients are proposed. These problems try to obtain a given occupation rate of the ORs at the same time that respects the order of the patients in the waiting list. These are combinatorial problems with high computational complexity, so three different heuristic solution methods are proposed (Sec. 5.2.4) and compared (Sec. 5.2.5). The results show that a Mixed Integer Linear Programming (MILP) problem solved by Receding Horizon Strategy (RHS) obtains the better scheduling in lowest time.

Doctors using the MILP problem must fix an appropriate occupation rate for optimizing the use of the ORs but without exceeding the available time. This has two main problems: i) inexperienced doctors could find difficult to fix an appropriate occupation rate, and ii) the uncertain in the surgery durations (large standard deviation) could result in scheduling with an over/under utilization. In order to overcome these problems, a New Mixed Integer Quadratic Constrained Programming (N-MIQCP) model is proposed in Sec. 5.3.2.

In the N-MIQCP model the surgeries duration as well as other durations involved in a surgical block (p.e., cleaning time) are considered random variables with normal probability density function. Considering some probabilistic concepts, quadratic constraints are included in N-MIQCP model to prevent the scheduling of blocks with a high risk of exceeding the available time. Moreover,

the objective function is composed by two balance terms, the first one tries to maximize the occupation rate of the ORs while the second one penalizes the disorder of the patients scheduled.

Two heuristic methods for solving the N-MIQCP problem are proposed in Sec. 5.3.3. Moreover, they are compared (Sec. 5.3.4) with other chance-constrained approaches in bibliography [62, 34]. The results conclude that the best schedulings are achieved using our Specific Heuristic Algorithm (SHA) due to similar occupation rate than using other approaches are obtained but our SHA respects much more the order of the patients in the waiting list.

In chapter 6 the combined scheduling of elective and urgent patients is considered. A 3 steps approach is proposed in Sec. 6.3. In the first step, the elective patients are scheduled for a target Elective Surgery Time (EST) in the ORs, trying to respect the order of the patients on the waiting list. In the second one, the urgent patients are scheduled in the remaining time ensuring that an urgent patient does not wait more than 48 hours. Finally, in the third step, the surgeries assigned to each OR (elective and urgent) are sequenced in such a way that the maximum time that an emergency patient should wait is minimized. Considering realistic data of the OSD in the LBH and following the simulation methodology explained in Sec. 6.4, different policies of time reserved in the ORs for elective and urgent patients are evaluated in Sec. 6.5. The results show that all ORs must be used to perform elective and urgent surgeries instead of reserving some ORs exclusively for one type of patients.

Finally in chapter 7 a software solution for surgery service management is given. First a Decision Support System for elective surgery scheduling is proposed in Sec. 7.1. The DSS use as core the SHA for the scheduling of elective patients, but it has other features related to the management of a surgery department. In Sec. 7.2 a software tool called CIPLAN which is based on the DSS is explained. The software tool has a friendly interface which has been developed in collaboration with doctors in the OSD of the LBH. A real case study comparing the scheduling using the manual method with the scheduling obtained by using CIPLAN is discussed in Sec. 7.3. The results show that 128.000 euros per year could be saved using CIPLAN in the OSD of the LBH. Moreover, the use of the tool allows doctors to reduce the time spent in scheduling to use it medical tasks.

The contributions of this thesis can be summarized in the following items:

- A set of abstractions, relaxations and modifications that allows to obtain formal subclasses of Petri Nets (S^4PR and DSSP) from SWN modeling healthcare systems are given in Chapter 2 (a first publication on the topic [20])
- A buffer pre-assignment method to ensure the liveness of some DSSP systems is provided in Chapter 3 (a first publication on the topic [19])
- A general approach for liveness enforcement on non live DSSP systems is given in Chapter 4 (a first publication on the topic [18])
- Three different mathematical programming models for scheduling of elective patients trying to obtain a given occupation rate at the same time that respect the order of the patients are proposed in Chapter 5. Moreover three different heuristic methods are evaluated and compared considering realistic data from HSD of the LBH (a first publication on the topic [21] and [17])
- A chance probability constrained problem (N-MIQCP) for scheduling of elective patients and a Specific Heuristic Algorithm that allows solving large instance in reasonable time are proposed in Chapter 5 (a first publication on the topic [15] and [16])
- A three step approach for the combined scheduling and the sequencing of elective and urgent surgeries is proposed in Chapter 6 (a first publication on the topic [22])
- A DSS for scheduling of elective patients implemented in a Software tool called CIPLAN is given and testing with a real case study in Chapter 7 (a first publication on the topic [15] and [16])

This thesis is structured as follows: in Chapter 1 basic definition and previous concepts related to Petri Nets are given. Moreover, previous results that will be used as a starting point are recalled. In Chapter 2 a methodology for obtaining formal subclasses of Petri Nets (S^4PR and DSSP) from SWN modeling healthcare systems is given. In Chapter 3 a buffer preassignment method for ensuring liveness in DSSP system is proposed. A more general method based on two-level nets for liveness enforcement of DSSP system is

explained in Chapter 4. Mathematical programming models and heuristic solutions methods for scheduling of elective patients are discussed in Chapter 5. The combined scheduling of elective and urgent patient is approached by a three-step methodology in Chapter 6, discussing different policies of OR time reservation. Chapter 7 gives a DSS for the management of surgical service. Moreover, a software tool based on the DSS (called CIPLAN) is explained and analyzed by a real case study. Finally, the concluding remarks are given.

Part I

Modeling and analysis in hospitals by the use of clinical pathways

Chapter 1

Preliminary: Petri Nets and previous results

Summary

This chapter introduces some basic definitions and concepts related to (discrete) Petri Nets. Classical results that will be used afterward are recalled. Moreover, some subclasses of Petri Nets (S^4PR and DSSP in particular) which are useful for modeling healthcare systems are detailed here. Finally, a methodology previously proposed for obtaining Stochastic Well-Formed Nets modeling healthcare systems is shown. The results derived from this methodology are the starting point for the next chapter. This chapter is based on [25, 68, 56, 66, 6].

1.1 Petri Net

Petri nets are a well-known formalism to deal with discrete event systems [25]. Discrete event systems that are characterized as being concurrent, asynchronous, distributed, parallel, nondeterministic, and/or stochastic can be modeled and analyzed by means of Petri nets. As a graphical tool, Petri nets can be used as a visual-communication aid similar to flow charts, block diagrams, and networks. In addition, tokens are used in these nets to simulate the dynamic and concurrent activities of systems. As a mathematical tool, it is possible to set up state equations, algebraic equations, and other mathematical models governing the behavior of systems.

The reader is assumed to be familiar with Petri nets and for a more deep introduction the following [63, 25, 49, 24] can be consulted. Here, the basic ideas are given.

1.1.1 Basic concepts

Definition 1.1. A Petri net system is a pair $\langle \mathcal{N}, \mathbf{m}_0 \rangle$, where \mathcal{N} is a net, $\mathcal{N} = \langle P, T, \mathbf{Pre}, \mathbf{Post} \rangle$ is a PN. P and T are disjoint (finite) sets of places and transitions, while \mathbf{Pre} and \mathbf{Post} are $|P| \times |T|$ sized, natural valued, incidence matrices, and \mathbf{m}_0 is the initial marking.

A PN can be graphically represented as a weighted bipartite graph. The nodes are represented by the set of places drawn as circles and by the set of transitions drawn as rectangles. These nodes are connected with a set of oriented arcs, each arc can connect a place with a transition or a transition with a place. $\mathbf{Pre}[p, t] = w > 0$ means that there is an arc from p to t with *weight* (or *multiplicity*) w , and $\mathbf{Post}[p, t] = w > 0$ means that there is an arc from t to p with *weight* w . The classical concepts of graph theory, as connectedness, strong connectedness can be directly applied to PN. For a node $v \in P \cup T$, the sets of its input and output nodes are denoted as $\bullet v$, and $v \bullet$, respectively. Each place can contain a natural number of tokens, this number represents the *marking* of the place. The initial distribution of tokens is called *initial marking* and is denoted, in general, by \mathbf{m}_0 .

A transition t is *enabled* at \mathbf{m} iff for every $p \in \bullet t$, $\mathbf{m}[p] \geq \mathbf{Pre}[p, t]$. The firing of t in a certain amount $\alpha \in \mathbb{N}$, $\alpha \leq \text{enab}(t, \mathbf{m})$ leads to a new marking

$\mathbf{m}' = \mathbf{m} + \alpha \cdot \mathbf{C}[P, t]$, where $\mathbf{C} = \mathbf{Post} - \mathbf{Pre}$ is the *token flow matrix* and we say that \mathbf{m}' is reachable from \mathbf{m} . This firing is also denoted by $\mathbf{m}[t(\alpha)]\mathbf{m}'$ or by $\mathbf{m}_0 \xrightarrow{\alpha t} \mathbf{m}_1$.

If \mathbf{m} is reachable from \mathbf{m}_0 through a sequence σ , a *state* (or *fundamental equation*) can be written:

$$\mathbf{m} = \mathbf{m}_0 + \mathbf{C} \cdot \boldsymbol{\sigma}, \quad (1.1)$$

where $\boldsymbol{\sigma} \in \mathbb{N}^{|T|}$ is the firing count vector, i.e., $\boldsymbol{\sigma}(t_i)$ indicates the times that transition t_i is fired in a sequence σ . The set of all markings reachable from \mathbf{m}_0 , or reachability set is denoted by $RS(\mathcal{N}, \mathbf{m}_0)$.

If the number of tokens in a place is always less than or equal to a natural number, then the place is *bounded*. A PN system is *bounded* when every place is bounded ($\forall p \in P, \exists b_p \in \mathbb{N}$ with $\mathbf{m}[p] \leq b_p$ at every reachable marking \mathbf{m}).

1.1.2 Structural concepts

Annulers of the incidence matrix are important because they induce certain invariant relations which are useful for reasoning about the behavior. *Flows* (*semiflows*) are integer natural annulers of \mathbf{C} . Right and left annulers are called T- and P-(semi)flows, respectively. We call a semiflow \mathbf{v} minimal when its support, $\|\mathbf{v}\|$, i.e., the set of its non-zero components, is not a proper superset of the support of any other semiflow, and the greatest common divisor of its elements is one.

If $\mathbf{y} \geq \mathbf{0}$ is such that $\mathbf{y} \cdot \mathbf{C} = \mathbf{0}$ then, every marking \mathbf{m} reachable from \mathbf{m}_0 satisfies: $\mathbf{y} \cdot \mathbf{m} = \mathbf{y} \cdot \mathbf{m}_0$. This provides a “token balance law”. Analogously, if $\mathbf{x} \geq \mathbf{0}$ is such that $\mathbf{C} \cdot \mathbf{x} = \mathbf{0}$, then $\mathbf{m} = \mathbf{m}_0 + \mathbf{C} \cdot \mathbf{x} = \mathbf{m}_0$. That is, T-semiflows correspond to *potential* repetitive sequences that bring the system from \mathbf{m}_0 back to \mathbf{m}_0 .

If $\mathbf{x} > \mathbf{0}$ exists such that $\mathbf{C} \cdot \mathbf{x} = \mathbf{0}$, the net is said to be *consistent*, and if $\mathbf{y} > \mathbf{0}$ exists such that $\mathbf{y} \cdot \mathbf{C} = \mathbf{0}$, the net is said to be *conservative*.

Traps and siphons are structural dual concepts with high importance in the analysis of many net properties as deadlock-freeness. A set of places, Θ , is a *trap* iff $\Theta^\bullet \subseteq \bullet\Theta$. In discrete net systems traps when marked, cannot get empty. Analogously, a set of places, Φ , is a *siphon* iff $\bullet\Phi \subseteq \Phi^\bullet$. One interesting property is that empty siphons will unavoidably remain empty throughout all the evolution of the net system.

1.1.3 Conflicts and structural conflict relations

A *conflict* is the situation when not all enabled transitions can occur at once. Formally, $t, t' \in T$ are in conflict relation at marking \mathbf{m} if there exist $k, k' \in \mathbb{N}$ such that $\mathbf{m} \geq k \cdot \mathbf{Pre}[P, t]$ and $\mathbf{m} \geq k' \cdot \mathbf{Pre}[P, t']$, but $\mathbf{m} \not\geq k \cdot \mathbf{Pre}[P, t] + k' \cdot \mathbf{Pre}[P, t']$. To fulfill the above condition it is necessary that $\bullet t \cap \bullet t' \neq \emptyset$. When $\mathbf{Pre}[P, t] = \mathbf{Pre}[P, t'] \neq 0$, t and t' are in *equal conflict (EQ) relation*. This means that they are both enabled whenever one is. By defining that a transition is always in EQ with itself, this is an *equivalence* relation on the set of transitions and each equivalence class is an *equal conflict set* denoted, for a given t , $EQS(t)$. $SEQS$ is the set of all the equal conflict sets of a given net.

1.1.4 Liveness and deadlock-freeness

A transition t is *live* iff it can ultimately occur from every reachable marking, i.e., for every $\mathbf{m} \in RS^{cut}(\mathcal{N}, \mathbf{m}_0)$, $\mathbf{m}' \in RS^{cut}(\mathcal{N}, \mathbf{m})$ exists such that t is enabled in \mathbf{m}' . A PN system $\langle \mathcal{N}, \mathbf{m}_0 \rangle$ is *live* if every transition is live. Liveness ensures that no single action in the system can become unattainable.

A PN is *deadlock-free* when any reachable marking enables some transitions. Clearly, deadlock-freeness is a necessary condition for liveness.

A PN is *structurally live (deadlock-free)* if an initial marking \mathbf{m}_0 exists for which the net system $\langle \mathcal{N}, \mathbf{m}_0 \rangle$ is live (deadlock-free).

1.1.5 Subclasses of Petri nets

The nets can be classified according to their structure and in this thesis two different subclasses that are used in Chapter 2 for modeling healthcare systems are considered.

The class of S^4PR nets [68] model cyclic concurrent sequential processes sharing resources. In S^4PR nets modeling healthcare systems the sequential processes represent clinical pathways that share the hospital resources (beds, doctors, nurses, test machine, etc). The formal definition is as follows:

Definition 1.2. Let $I_{\mathcal{N}} = \{1, 2, \dots, m\}$ be a finite set of indices. An S^4PR net is a connected generalized self-loop free Petri net $\mathcal{N} = \langle P, T, \mathbf{Pre}, \mathbf{Post} \rangle$ where:

1. $P = P_0 \cup P_S \cup P_R$ is a partition such that:
 - a) $P_S = \bigcup_{i \in I_N} P_{S_i}$, $P_{S_i} \neq \emptyset$ and $P_{S_i} \cap P_{S_j} = \emptyset$, for all $i \neq j$. Places of P_S are called process places.
 - b) $P_0 = \bigcup_{i \in I_N} \{p_{0_i}\}$. Places of P_0 are called idle places.
 - c) $P_R = \{r_1, r_2, \dots, r_n\}$, $n > 0$. Places of P_R are called resource places.
2. $T = \bigcup_{i \in I_N} T_i$, $T_i \neq \emptyset$, $T_i \cap T_j = \emptyset$, for all $i \neq j$.
3. For all $i \in I_N$, the subnet \mathcal{N}_i generated by $P_{S_i} \cup \{p_{0_i}\} \cup T_i$ is a strongly connected state machine, such that every cycle contains p_{0_i} .
4. For each $r \in P_R$ there exists a minimal P -Semiflow, $\mathbf{y}_r \in \mathcal{N}^{|P|}$, such that $\{r\} = \|\mathbf{y}_r\| \cap P_R$, $\mathbf{y}_r[r] = 1$, $P_0 \cap \|\mathbf{y}_r\| = \emptyset$, and $P_S \cap \|\mathbf{y}_r\| \neq \emptyset$.
5. $P_S = \bigcup_{r \in P_R} (\|\mathbf{y}_r\| \setminus \{r\})$.

Example 1.3. Let us consider the PN in Fig. 1.1. This net is a S^4PR composed by two subnets (\mathcal{N}_1 and \mathcal{N}_2) and 3 resources (r_1 , r_2 and r_3).

The set of places is divided as follows:

- Process places $\rightarrow P_S = \{p_1, p_2, p_3, p_4, p_5\}$
- Idle places $\rightarrow P_0 = \{p_{01}, p_{02}\}$
- Resource places $\rightarrow P_R = \{r_1, r_2, r_3\}$

Condition 3 in Def. 1.2 states that each subnet is a strongly connected state machine, where every cycle contains the idle place p_{0_i} . In Fig. 1.1 the subnets \mathcal{N}_1 and \mathcal{N}_2 are two state machines (each transition has one input and one output place) which are strongly connected. The net \mathcal{N}_1 has two cycles, $\{p_{01} \rightarrow p_1 \rightarrow p_2\}$ and $\{p_{01} \rightarrow p_1 \rightarrow p_3\}$, containing the idle place p_{01} . Similarity, the cycles in \mathcal{N}_2 are $\{p_{02} \rightarrow p_4\}$ and $\{p_{02} \rightarrow p_5\}$ which contain the idle place p_{02} .

Condition 4 in Def. 1.2 specifies that for each resource there is a unique minimal P -invariant whose support is not containing any idle place. In Fig 1.1 the P -invariants \mathbf{y}_{r_i} corresponding to each resource place r_i are:

- $\mathbf{y}_{r_1} = \{r_1, p_2\}$
- $\mathbf{y}_{r_2} = \{r_2, p_1, p_2, p_3, p_4\}$

- $\mathbf{y}_{r_3} = \{r_3, p_5\}$

Finally, condition 5 in Def. 1.2 says that all process places must have assigned at least a resource place. Seeing the P -invariants \mathbf{y}_{r_i} it can be checked that all process places in Fig. 1.1 are covered by at least one resource place.

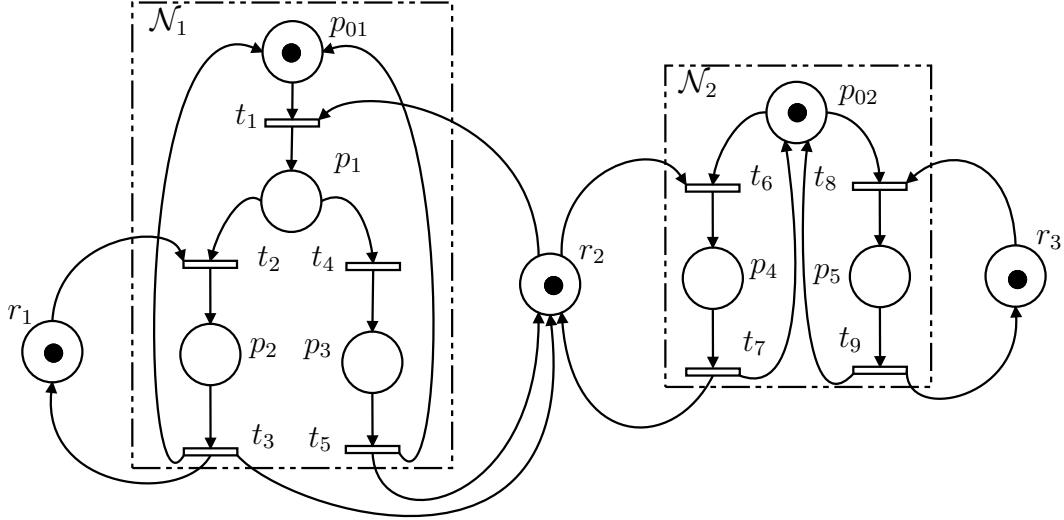


Figure 1.1: A S^4PR net example.

Deterministically Synchronized Sequential Processes (DSSP) [56] are modular Petri Net (PN) systems composed by a set of state machines PNs (called also agents) cooperating in a distributed way through asynchronous message passing. This cooperation is realized through a set of buffers to/from which the agents consume/produce the items (messages). Considering healthcare systems, each agent models a clinical pathway while the buffers represent the information channels. The formal definition of DSSP is as follows:

Definition 1.4. A P/T system $\langle P, T, \mathbf{Pre}, \mathbf{Post}, \mathbf{m}_0 \rangle$ is a DSSP where $P = P_1 \cup P_2 \cup \dots \cup P_n \cup B$, $T = T_1 \cup T_2 \cup \dots \cup T_n$, and the following holds:

1. Let $\mathcal{N}_i = \langle P_i, T_i, \mathbf{Pre}[P_i, T_i], \mathbf{Post}[P_i, T_i] \rangle$ be a subnet called agent i . Then, $\langle \mathcal{N}_i, \mathbf{m}_0[P_i] \rangle$ is a 1-marked live state machine. Given a node $x \in \bigcup_{i=1}^n (P_i \cup T_i)$, $I(x)$ denotes the index of the subnet it belongs to, i.e., $x \in P_{I(x)} \cup T_{I(x)}$.

2. For every buffer $b \in B$, $\text{dest}(b) \in \{1, \dots, n\}$ exists such that $b^\bullet \subseteq T_{\text{dest}(b)}$.
3. If $t, t' \in p^\bullet$, where $p \in P_{\text{dest}(b)}$, then $\mathbf{Pre}[b, t] = \mathbf{Pre}[b, t']$.

Example 1.5. Let us consider the PN in Fig. 1.2. This net is a consistent and conservative DSSP composed by:

1. Two agents \mathcal{N}_1 and \mathcal{N}_2 , each one having two T -semiflows. In particular, \mathcal{N}_1 has $\mathbf{x}_1 = t_1 + t_2^*$ and $\mathbf{x}_2 = t_3 + t_4$ while \mathcal{N}_2 has $\mathbf{x}_3 = t_5 + t_6$ and $\mathbf{x}_4 = t_7 + t_8$.
2. Two pairs of buffers (b_1, b_2) and (b_3, b_4) in consumption - production relation.

Condition 1 in Def. 1.4 imposes that each agent is a 1-marked live state machine. In Fig. 1.2 the subnets \mathcal{N}_1 and \mathcal{N}_2 are state machines with only a token.

Condition 2 in Def. 1.4 defines the buffers as private destination, this means that a buffer can only have output transitions in one agent. In Fig. 1.2 buffers b_1 and b_3 are private destination of \mathcal{N}_1 while buffers b_2 and b_4 are private destination of \mathcal{N}_2 .

Finally, condition 3 in Def. 1.4 imposes that a buffer never constraints internal choices of its destinations. The conflicts in the net of Fig. 1.2, $\{t_1, t_3\}$ and $\{t_1, t_7\}$, are not constrained by the buffers.

1.1.6 Rank theorem in DSSP

For DSSP, rank theorem gives a necessary and sufficient condition to characterize the structural liveness [66, 55].

Theorem 1.6 (Rank Theorem). [65] Let \mathcal{N} be a conservative DSSP net. A marking \mathbf{m}_0 exists such that $\langle \mathcal{N}, \mathbf{m}_0 \rangle$ is a live DSSP iff \mathcal{N} is consistent and $\text{rank}(C) = |\text{SEQS}| - 1$. ■

*In this thesis we use the multi-set notation for vectors. For example, $\mathbf{x}_1 = t_1 + t_2$ is a vector with the only elements different by zero corresponding to t_1 and t_2 . Here, $\mathbf{x}_1 = [1 \ 1 \ 0 \ 0]^T$.

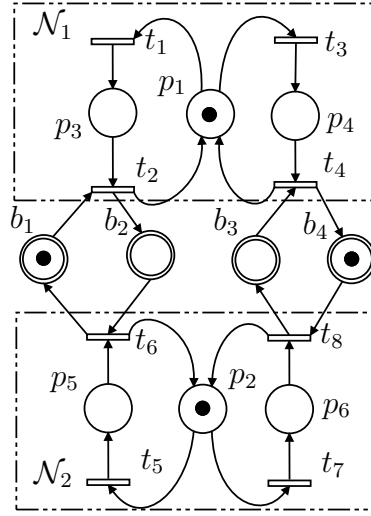


Figure 1.2: Motivation example: A not live DSSP net

Example 1.7. Let us consider the PN in Fig. 1.2. This net is a consistent and conservative DSSP where the set of all equal conflict sets is: $SEQS = \{\{t_1, t_3\}, \{t_2\}, \{t_4\}, \{t_5, t_7\}, \{t_6\}, \{t_8\}\}$. The net does not fulfill the rank theorem: $|SEQS| - 1 = 5 \neq \text{rank}(\mathbf{C}) = 6$. According to Th. 1.6 it is structurally not live. The net will reach a deadlock marking when the EQs in both subnets will be solved by firing the T-semiflows whose consumption buffer is empty. For the marking in Fig. 1.2, by firing \mathbf{x}_2 in \mathcal{N}_1 in parallel with \mathbf{x}_3 in \mathcal{N}_2 . In particular, the sequence t_3t_5 will lead to the deadlock marking $\mathbf{m}' = p_3 + p_5 + b_1 + b_4$.

In Chapters 3 and 4 two different approaches to force liveness in non-live DSSP systems are given.

1.2 Previous work: clinical pathways assessment

In [6] a modeling methodology based on clinical pathways was proposed for hospital assessment. This methodology is based on modeling clinical pathways associated with the cure of different diseases and medical problems in the hospital. The mentioned methodology obtains *Stochastic Well-formed Net* (SWN) modeling healthcare systems. In Chapter 2, we propose a set of re-

laxation, abstraction and modification that applied to SWNs, allows to obtain simpler structures for analyzing different facets of the system.

They propose a very simple graphical modeling language based on a small number of primitive elements through which the medical doctors could introduce a clinical pathway for a specific disease. Three essential aspects related to a clinical pathway can be specified in this language: (1) patient flow; (2) resource utilization; and (3) information interchange. This high-level language is a domain specific modeling language called *Healthcare System Specification* (HSS), and it is defined as an *Unified Modeling Language* (UML) profile [31, 59, 60]. The domain model can be represented graphically with a class diagram. In the context of healthcare systems two views are considered: structural and behavioral. The first one is related with the specification of the hospital assets (beds, operation rooms, patients information, etc) while the second one is concerned with the specification of the clinical scenarios, that is, the set of clinical pathways that compose the system.

Clinical pathways are modeled by *activity diagram* that represent the flow of activities through the nodes. Notice that the elements of the UML activity diagram ensuring the parallel execution of activities are not included, limiting the systems that can be modeled. In particular, the modeling methodology proposed can be used to model the flow of treatments, medical tests and cures that a patient should follow according to the clinical pathways in order to get healthier. Other aspects, as for example decisions of different medical test that can be taken in parallel are not possible to model. The motivation of not using these elements is to provide a modeling language with the smallest set of primitives that: (i) comply with the requirements of the final users, i.e., the medical doctors, for the specification of clinical pathways, and (ii) are easy to learn and use by the medical doctors. As by-product, the formal models generated with the transformation have well-defined structures, thus efficient structural analysis techniques can be applied. Nevertheless, this does not mean that the models are sequential, the concurrency is appearing due to the parallel execution of different clinical pathways competing for common hospital resources.

The HSS-UML profile is useful to specify the hospital description through the development of the clinical pathways. However, UML is a semi-formal modeling language that cannot be used for analysis purposes. In order to be

able to perform an analysis of the system, a *model-to-model* (M2M) transformation was proposed. This transformation is based on a set of rules that are applied to a UML profile for obtaining a colored Petri Net model, particularly a *Stochastic Well-formed Net* (SWN).

In the following of this section, a brief introduction to SWN is given and the M2M transformation is recalled. For a detailed explanation consult [6].

1.2.1 Stochastic Well-formed Net

Colored Petri nets (CPN) as well as other high-level Petri net formalisms (e.g., Pr/T nets) are crucial from the point of view of the expressive power of this class of formalisms. The possibility of associating information with tokens and of parameterizing transition firing made it possible to represent very concisely systems that would have required huge uncolored nets to be described. Stochastic Well-formed colored nets (SWN's) [12] are substantially identical to CPNs from the expressive power point of view. However, the syntactic definition of SWN's leads to new, more efficient analysis algorithms based on the original concept of *symbolic marking*. In the sequel we give a short formal description of the SWN formalism.

A *Stochastic Well-formed Net* (SWN) [12] is a high level Petri Net $\mathcal{N} = \langle P, T, C, D, W^-, W^+, W^h, \phi, \pi, \Omega, M_0 \rangle$ where:

P is the set of places, T is the set of transitions, C is the set of basic color classes and D is a function that associates a color domain to each place and transition of the Net.

As in PNs, places of SWNs together with their marking play the role of describing the system state while transitions represent events that cause the state changes. In SWNs a token can incorporate some information, indeed a token can be regarded as an instance of a data structure with a certain number of fields whose semantics depend on the place the token belongs to. The definition of the “data type” associated with each place is called *place color domain*. The fields data types are selected from a set of basic types called *basic color classes*. The specification of the basic color classes is part of the net definition. The color domain of place p is denoted $C(p)$. Places in SWN models represent the state of a (multi)set of possibly distinguishable objects.

The transitions in SWNs can be considered to be procedures with formal parameters. The formal parameters are called *transition color domain*; their declaration is part of the net description, and the type associated with each parameter must be a basic color class. A transition color domain is defined in the same way as a place color domain. The list of classes in the color domain defines the type associated with the transition parameters. The color domain of a transition t (denoted $C(t)$) is constrained by the color domains of its input, inhibitor, and output places. The relation between transition and place color domains is defined through the input, output and inhibitor arc functions W^- , W^+ , W^h .

ϕ is a function that associates to each transition $t \in T$ a guard expression while Π is the priority function that assigns a priority level to each transition. Ω is a function that associates to each timed transition a (mean) firing rate. In order to fire a transition, it is necessary to specify actual values for its formal parameters. The enabling check of a transition instance and the state change caused by its firing depend (again) on the arc expressions that label the arcs connected to the transition. The state change caused by the firing amounts to subtracting/adding from/to each input output place p the multiset, resulting from evaluating the corresponding arc expression.

Finally M_0 is the initial marking function. A more detailed explanation can be found in [12]. The notation $M(p)$ denotes the marking of place p , i.e., the multiset of $C(p)$ contained in p according to marking M .

Model to model transformation: from UML-HSS to SWN

Using techniques of model to model transformations [50], SWNs models are obtained from the HSS-UML models. In Chapter 2, a set of relaxations, abstractions and modifications are applied to these SWN in order to obtain particular facets of the system. The first relaxation applied is a *decolorization*, so in the explanation of the M2M transformation we will pay attention to the obtained net structure in the SWN, but not to the color classes. A detailed explanation of the UML-HSS profile and the M2M transformation can be found in [6].

Fig. 1.3 shows a motivation example of a treatment where a doctor has to decide between performing task 1 or task 2 on a patient. Task 1 is performed

by the doctor and task 2 is performed by a nurse. After performing one of these two activities, the treatment has finished.

The upper part of Fig. 1.3 shows this treatment modeled by UML-HSS profile while the downer part of Fig. 1.3 shows the successive steps applied for obtaining the SWN structure during the model-to-model transformation.

The methodology for transforming UML-HSS profiles into a SWN model is based on a set of rules that are applied sequentially in three steps following Alg. 1

Algorithm 1: Model to model transformation algorithm: from HSS-UML to SWN

- 1: **Step 1.** *Applying transformations rules R1 to R7.*
 - Rules R1-R5 transform each node in the AD in a SWN subnet.
 - Rules R6 & R7 transform the resources and information channels in isolated places.
 - 2: **Step 2.** *Composition of the SWN elements generated in Step 1.*
 - The places with the same label are merged.
 - 3: **Step 3.** *Assignment of resources and manipulation of healthcare information by applying rules R8-R12.*
 - R8 & R9 assign and release the resources respectively.
 - R10-R12 create, use and reads health information.
-

Step 1. *Applying transformations rules R1 to R7.* Rules R1-R5 are applied to the nodes in the activity diagrams representing the clinical pathways:

- Rule R1 maps the initial node of the activity diagram into a single SWN place labeled with the activity edge leaving the initial node.
- Rule R2 transforms an action node *task A* into a SWN subnet composed by a timed transition with an input and an output place. The label of the input (output) place is the identifier of the *task A* in-coming (out-coming) edge.
- Rule R3 transforms a choice node into two conflicting SWN immediate transitions.
- Rule R4 transforms a merge node into a SWN subnet that unify alternative flows in a place.

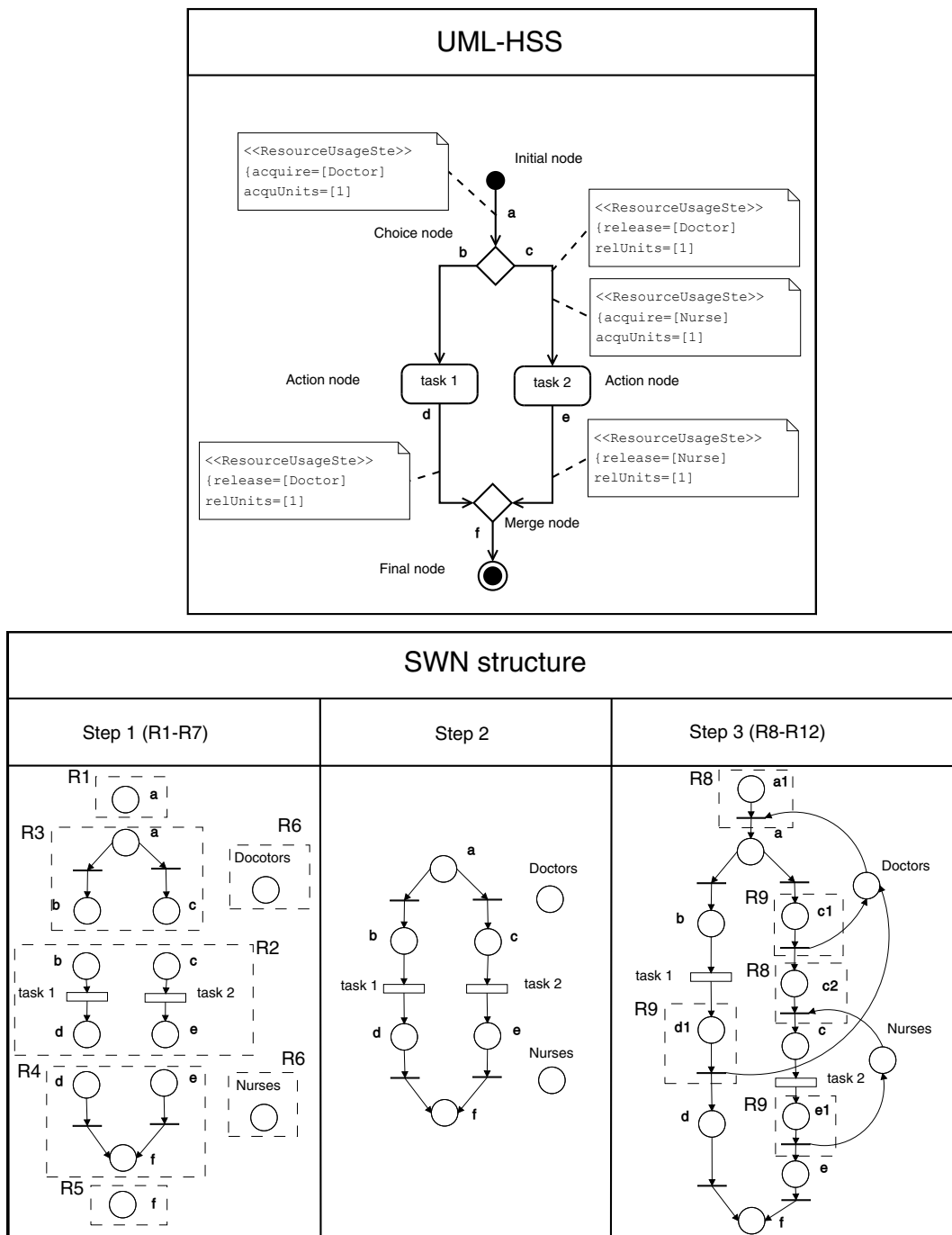


Figure 1.3: A simple motivation care example modeled by UML-HSS and transform to SWN structure.

- Rule R5 maps the final node of the activity diagram into a single SWN place label with the activity edge incoming the final node.

Rules R6 and R7 are applied to the class diagram modeling the available resources and the healthcare information.

- Rule R6 maps a resource type into a single SWN place.
- Rule R7 maps a healthcare information type into a single SWN place.

Step 2. *Composition of the SWN elements generated in Step 1.* In this step, the SWN places with common labels are merged. The result after performing this composition is a set of SWN subnets and a set of isolated places. Each subnet represents the control flow of the patients according to a clinical pathway while the isolated places represent the resources and the healthcare information channels.

Step 3. *Assignment of resources and manipulation of healthcare information.* Clinical pathways are modeled by UML activity diagrams and there are some edges in these activity diagrams interacting with resource type and health information types. Resources can be assigned and released (rules R8 and R9) while health information can be created, used and read (rules R10-R12). When in the activity diagram an edge j interacts with one of these types (resource or information), in the SWN a place and an immediate transition are added before the place labeled as j . The new added transitions have arcs (according to rules R8-R12) from/to the places mapped by rules R6 and R7.

- Rule R8 assigns a resource. In the SWN an arc is added from the place representing the resource type (mapped by R6) to the transition.
- Rule R9 releases a resource. Similar to rule R8, but with the arc from the transition to the place representing the resource.
- Rule R10 creates a health information. In the SWN an arc is added from the transition to the place representing the health information type (mapped by R7).
- Rule R11 uses a health information. In the SWN an arc is added from the place representing the health information type (mapped by R7) to the transition.

- Rule R12 reads a health information. In the SWN a bidirectional arc is added between the place representing the health information type and the transition.

1.3 Discussion

Petri Nets constitute a family of formalisms appropriate for modeling discrete event systems (systems in which state variables take discrete values from a set that can be counted, not necessarily finite). Its distributed state is a vector of non-negative integers. This is an important advantage over other modeling formalities of discrete systems, such as automata, where the state space is a set of unstructured symbols of global states. This property has been exploited to develop very diverse analysis techniques that do not require the enumeration of the state space (structural analysis). From a modeling perspective, a key feature of the PN is their capacity for graphic representation and visualization of the model. In this thesis, PNs will be used for modeling and analysis of healthcare systems, particularly the subclasses of S^4PR and DSSP nets. In this chapter, basic definitions and concepts related with PN are given. Moreover a methodology [6] to model healthcare systems using *Stochastic Well-formed Nets* (SWN) is recalled. This methodology will be used as a starting point in the next chapter.

Chapter 2

From healthcare system specification to formal models

Summary

A domain-specific modeling language called Healthcare System Specifications (HSS) is used for developing clinical pathways. It was also proposed a model to model transformation which obtains from pathways HSS specification, a Stochastic Well-formed Net (SWN). The SWN has the capacity to model the complex healthcare system, however, it is hard to perform qualitative and quantitative analysis in this kind of nets. This chapter presents a set of *relaxations, abstractions and modifications* to be applied in the SWNs in order to obtain subclasses of Petri Nets in which formal analysis can be performed. In particular, we obtain nets belonging to the following classes: S^4PR and DSSP. The first one can be used to analyze the shared resources of the hospital while the models in the second class allow managing the interchange of information between clinical pathways. The results in this chapter have been published on [20].

2.1 Motivation

2.1.1 Clinical pathways

A *clinical pathway* [37] is a set of medical activities (tasks) and medical decisions that must be followed to cure a disease. They are protocolized and must be developed and approved by medical staff in the hospitals. In this way, the use of clinical pathways protect medical doctor against legal issues, although, any deviation must be well justified. From the patient's point of view, the use of clinical pathways standardizes the process of curing a disease. Regardless the doctor, all patients with the same pathology and conditions follow the same clinical process.

To carry out each task or activity in a clinical pathway some resources could be required. A correct assignment and release of the resources is desired to have benefits in the implementation of the clinical pathways. Moreover, considering the interconnection of clinical pathways sharing resources, non-desired behaviors could appear in the system. To verify the correct usage (assignment and release) of the resources it is necessary the use of formal models allowing to study the behavior of the system.

2.1.2 Starting point

In [6] a new methodology to describe healthcare system (mainly hospitals) was proposed. This methodology (recalled in Sec. 1.2) is based on modeling clinical pathways associated with the cure of different diseases and medical problems in the hospital. First, a UML profile called HSS is used to specify the hospital description and then a *model-to-model* (M2M) transformation is applied to obtain SWN from the HSS-UML profile.

Considering state-based techniques or event simulations, the SWN model can be used to estimate performance indicators in the hospital. Many software tools exists for the analysis of the colored Petri net models, e.g., CPN tools [72], TimeNet [73] or GreatSPN [57]. However, due to the state explosion problem, the techniques based on exploring the state space could be difficult to apply. On the other hand, the techniques based on event simulations could require a lot of time.

2.1.3 Facets of the healthcare system

In order to overcome this time and space problems, in this chapter, we define some relaxations, abstractions and modifications for obtaining formal subclasses of Petri Nets in which net-level analysis techniques can be applied. These new system abstractions are called facets and each one captures some relevant aspects of the system behavior.

The SWN is suitable for representing the multifaceted-nature of a complex healthcare system. In this chapter two important facets and their formal models are identified:

- *Resource management facet* which we prove to belong to the class of S^4PR net model and it is focused on the resource competition.
- *Handshake between clinical pathways facet*. The formal model considered for this facet is a DSSP which studies the cooperation between clinical pathways.

The chapter is organized as follows: Sec. 2.2 gives the methodology to get the proposed facets, moreover it is proved that the formal models obtained are Petri Net of classes S^4PR and DSSP respectively. Sec. 2.3 shows a case application where from a SWN, the facets of the system are obtained. Finally, Sec. 2.4 discuss the chapter.

2.2 Relaxations, abstractions and modifications to get Formal Models

Public healthcare systems are complex due to their structure and their large size. Moreover, there are different groups of people (patients, doctors, managers, government, etc) with different interests, expectations and visions of the healthcare system. The most relevant aspects of the behavior of the system for each one of these groups (stakeholders) are called *facets*. The SWN models, generated from the UML-HSS specifications, allow to represent the multifaceted nature of healthcare systems. However, it is difficult to obtain qualitative and quantitative properties in these models. The methodology

proposed here is based on some *relaxations* (Re), *abstractions* (Ab) and *modifications* (Mo) that are applied to SWN models in order to obtain formal subclasses of PN representing particular facets of the system. The proposed Re, Ab and Mo are illustrated in Fig. 2.1 and they are explained in the following subsections.

2.2.1 Resource management facet

To analyze the use of resources in healthcare systems, we propose a procedure composed by Re1, Mo2, Ab3 and Mo4 that are sequentially applied to the SWN model. The result is a Petri Net of class S⁴PR (Def. 1.2) where qualitative and quantitative analysis can be performed. The steps necessary to get a S⁴PR from a SWN net are as follows:

Re1: Decolorization. SWNs are colored PN where each place has assigned a color domain. The first relaxation proposed is a decolorization of the SWN model. Decolorization consist on removing the differences between individuals making them indistinguishable. So, all places added by applying rules [R1]-[R12] will not have any color domain assigned and all arcs inscriptions are removed. This decolorization removes customized synchronization generated by rule R11 where a given patient could require health information of herself. This synchronization is also lost by decolorizing rule R12 which reads information of a given patient.

Mo2: Strongly connected subnets. In order to have strongly connected subnets, fusion of places representing the initial and final nodes (mapped by rules [R1] and [R5] respectively) is performed. This modification imposes that a patient is returning to the initial place once she/he is completing the clinical pathway. Since *Re1* makes all patients indistinguishable, each time that a token starts a clinical pathway, it can be considered a new patient. Notice that if in the UML-HSS specifications, resources are acquired at the output of the initial node, the place to be merged is not the place that represents the initial node, but the previous place without resources assigned. For example, in downer part of Fig. 1.3 the places to be merged will be *a1* with *f*.

Ab3: Abstraction of information interchange. Places modeling the healthcare information are removed together with the connected arcs. These places model the message channels and, from the usage of resources point of

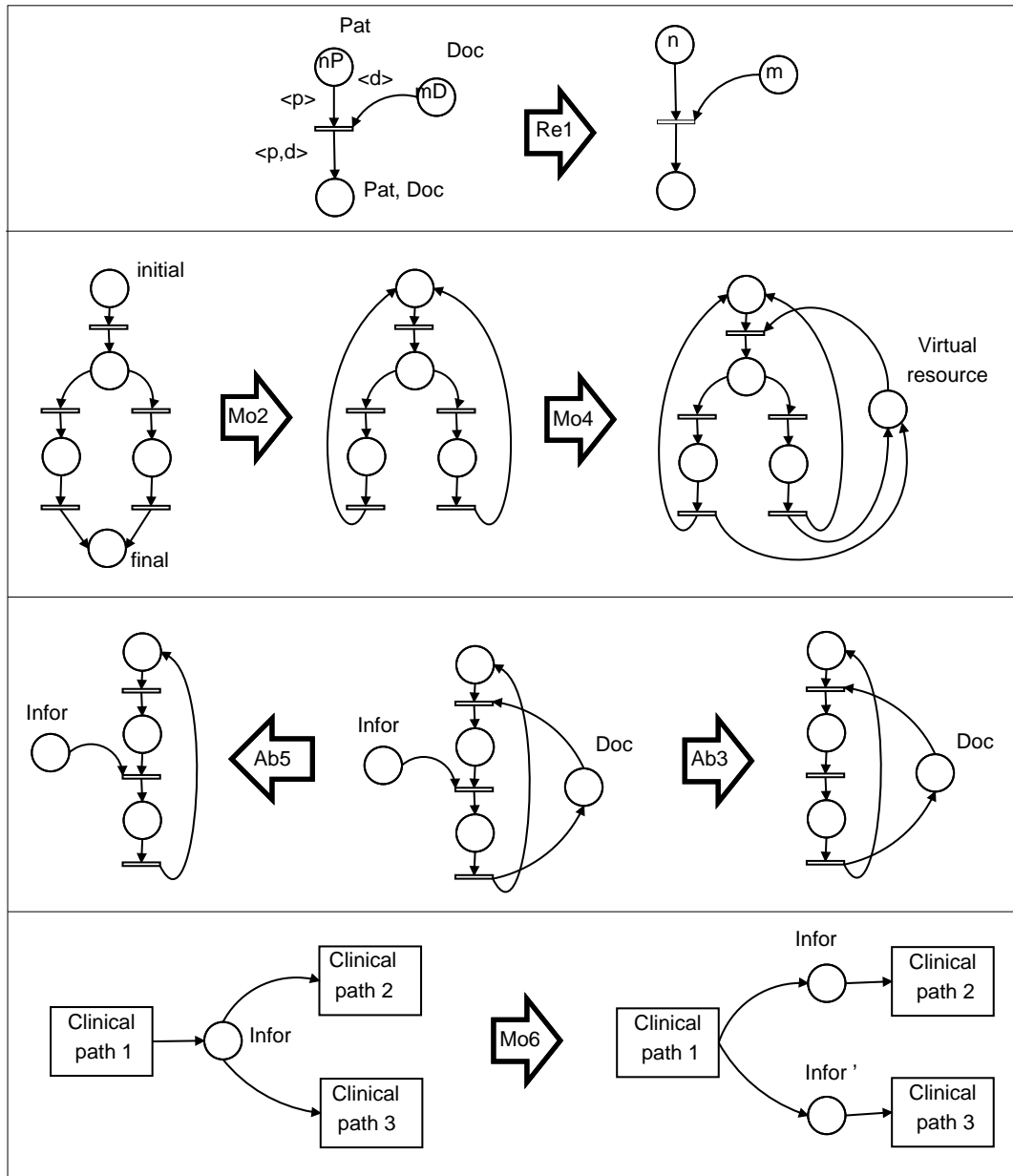


Figure 2.1: Relaxations abstractions and modifications applied to SWN model in order to obtain different facets of the healthcare system

view, can be ignored.

Mo4: Adding virtual resources. A new place modeling a virtual resource is added. This resource must be used in all steps of each clinical pathway. So, in each clinical pathway, this resource is assigned at the beginning (output transitions of the merged places in *Mo2*) and it is released at the end (input transitions of merged places in *Mo2*). This resource can be seen as an identification bracelet given to the arrived patient. Assuming that the new added place has an initial marking sufficiently high, the new virtual resource will not compromise the performance of the system.

Lemma 2.1. *Let Σ be a healthcare system modeled according to the UML-HSS specifications given in Sec. 1.2. Let Σ' be the SWN system obtained by applying to Σ the M2M transformation rules described in Alg. 1. Let Σ_1 be the net system obtained by applying *Re1*, *Mo2*, *Ab3* and *Mo4* to Σ' , then Σ_1 is a Petri Net system of S^4PR class.*

Proof. Lemma 2.1 is proved showing that the resulting net \mathcal{N}_1 fulfills all conditions of Def. 1.2.

Condition 3 of Def. 1.2 states that a S^4PR net is composed by a set of strongly connected state machines. In the UML-HSS profile, each clinical pathway is modeled by an activity diagram. It is composed by an initial node connected with a final node through one or more alternative paths. Notice that parallel execution of activities and synchronization are not allowed in UML-HSS specifications (only decisions and joins are allowed). A SWN subnet for each activity diagram (clinical pathway) is obtained after applying Step 1 and Step 2 of Alg. 1. These subnets have the structure of state machines (each transition has a unique input and a unique output place). Step 3 of Alg. 1 transforms some places p of the previously generated SWN subnets into sequences $p_i \rightarrow t \rightarrow p$ which do not affect the state machine structure of the subnets. By relaxation *Re1*, the color domain of the places and all arc inscriptions are removed. Modification *Mo2* transforms each subnet in a strongly connected net by fusion of the initial place with the final one. Therefore, in the resource management facet, each clinical pathway of the healthcare system is modeled by a strongly connected state machine.

Condition 1 of Def. 1.2 states that the set of places $P = P_0 \cup P_S \cup P_R$ of a S^4PR net is a partition such that: P_S is the set of process places, P_0 is the set

of idle places while P_R is the set of resource places. In each strongly connected state machine, the idle place is the merged place obtained after modification *Mo2*. The other places in each state machine are process places. Finally the resources places are the places mapped by rule R6 (M2M transformation) and the new virtual resource place added by modification *Mo4*. Notice that places generated by R7 (healthcare information places) are removed by abstraction *Ab3*.

Condition 2 of Def. 1.2 says that the set of transitions is composed by the union of transitions belonging to each strongly connected state machine.

Condition 4 of Def. 1.2 specifies that for each resource there is a unique minimal P-invariant whose support is not containing any idle place. In the UML-HSS profiles, the resources are assumed to be no consumable, so they are assigned for performing one or more activities and released when these activities have finished. The places modeling the set of activities for which a given resource is required together with the resource place compose the P-invariant of the mentioned resource. The idle place models the state of patients who are waiting for starting a clinical pathway, so these idle places never have assigned resources.

Finally, condition 5 of Def. 1.2 says that all process places must have assigned at least a resource. Modification *Mo4* adds a virtual resource which is assigned to all steps in each clinical pathway, so all process places have assigned at least the added virtual resource.

The net in the system obtained for resource management facet fulfill all conditions in Def. 1.2, so it is proved to be a S^4PR net. \square

Using this facet it can be studied the utilization of the resources. Net level techniques for deadlock analysis, prevention and avoidance have been studied in literature for S^4PR [67]. Besides, has been observed that a more correct assignment and release of the resources is achieved when rule R9 is applied before rule R8. In this way, if an edge of a clinical pathway in the UML-HSS profile require and release resources, in the SWN and consequently in the S^4PR , first the resources are released and then the resources are acquired. Fig. 2.2 shows an illustrative example where rules R8 and R9 are applied in different order. Option 1 applies first rule R8 and then R9 while option 2 considers first rule R9 and then R8. Structure obtained by option 1 could result in deadlock

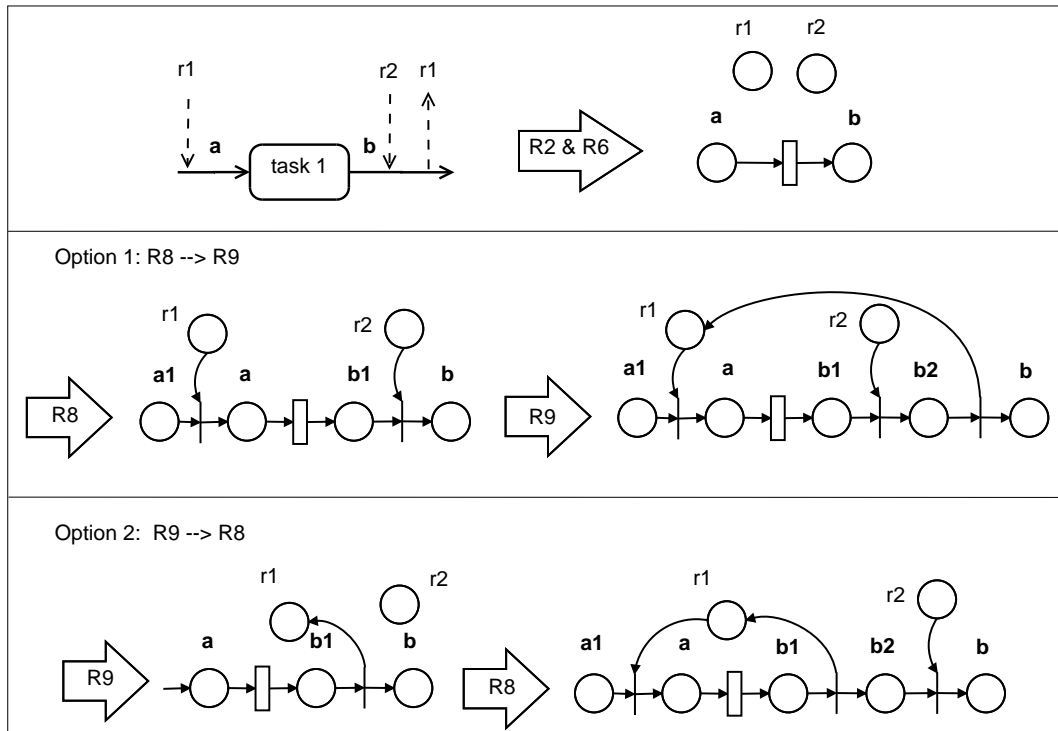


Figure 2.2: Differences between the net structures obtained depending on the order that rules R8 and R9 of the M2M transformation are applied.

situations easier than option 2 due to the need of acquiring resource $r2$ before releasing $r1$.

2.2.2 Handshake between clinical pathways facet

In addition to resource sharing, clinical pathways also interact between them through the use of healthcare information, particularly by medical records. In this facet, only the flow of patients and the interchange of health information is considered. This facet is obtained from the SWN by a similar procedure that in the previous facet. The first two steps, *Re1* (decolorization) and *Mo2* (strongly connected subnets) are the same. Then, a new abstraction (*Ab5*) and modification (*Mo6*) are applied:

Ab5: Abstraction of resources. Places modeling the resources (mapped by R6) are removed together with the connected arcs. These places model the

availability of resources and from the information interchange point of view can be ignored.

Mo6: Duplicate some information channels. In the case in which the UML-HSS profile is using or reading from a health information place from more than one clinical pathway, the corresponding place modeling this health information channel (mapped by rule R7) is duplicated as many time as clinical pathways are reading or using this information. Moreover, each one of these duplicated places is transformed in a private clinical pathway destination by removing the output arcs to transitions belonging to others clinical pathways. After applying Mo6 the information in a channel can only be read by one private clinical pathway making the system distributed. For example, considering Mo6 in last row of Fig. 2.1: place “infor” has output arcs in “clinical path 2” and “clinical path 3”. After applying Mo6 the place “infor” has been duplicated in two places, each one having output arc only in one clinical path (2 or 3).

Lemma 2.2. *Let Σ be a healthcare system modeled according the UML-HSS specifications given in Sec. 1.2. Let Σ' be the SWN system obtained by applying to Σ the M2M transformation rules proposed in Sec. 1.2.1. Let Σ_2 be the net system obtained by applying *Re1*, *Mo2*, *Ab5* and *Mo6* to Σ' , then Σ_2 is a Petri Net system belonging to the DSSP subclass.*

Proof. Condition 1 of Def. 1.4 imposes that each agent is a 1-marked live state machine. Lemma 2.1 proofs that each clinical pathway is modeled by a strongly connected state machine after applying to the SWN the relaxation *Re1* and the modification *Mo2*. The place that must be marked in each state machine is the merged place resulting from modification *Mo2*.

Condition 2 of Def. 1.4 states that the buffers are private destination, this means that a buffer can only have output transitions in one agent. Abstraction *Ab5* removes resource places, so the only places that do not belong to the state machines and that therefore can be seen as buffers, are the health care information places. Modification *Mo5* ensures that each one of these healthcare information places have a unique destination state machine.

Finally, condition 3 of Def. 1.4 imposes that a buffer never constraints internal choices of its destinations. The healthcare information places (mapped by rule R7) can be seen as the buffers of the system. Their information can be

used (rule R11) or can be read (rule R12) by the clinical pathways (agents). When in the UML-HSS model, an edge j of an activity diagram representing a clinical pathway reads/uses some information, in the SWN a place and an immediate transition are added just before the place modeling the edge j (rule R11 and R12). The information is read/used by an arc from the healthcare information place to the new added transition. So, a buffer never is assigned to a conflict transition and in this way never constraints internal choices. \square

Performance analysis cannot be performed on this facet (DSSP) due to two reasons: resources are not considered (removed by Ab5), and there is only 1 patient (represented by a token) in each clinical pathway (state machine). However, this facet is interesting from the point of view of cooperation between clinical pathways in the information interchange. Properties related with the behavior of the net can be analyzed, e.g., deadlock free of the net. The analysis of these properties allows us to check if a communication protocol is bad designed and consequently a patient can be blocked during a clinical pathway. The possible errors can be corrected in the DSSP net and these corrections will be propagated back to be included in the HSS description in order to contain the correct specification for the stakeholders.

2.3 Case application

In this section, we show a case application where from the SWN model we obtain the two facets presented in this chapter. The clinical scenario is composed by two clinical pathways that interact between them by competing for a resource and interchange information. Each one of these clinical pathways models the workflow of a X-rays team (team A or team B) performing diagnostic tests on patients with back or knee illness.

Fig. 2.3 shows the SWN of the scenario to be analyzed. Patients following each clinical pathway are waiting in places A0 and B0, respectively. Each clinical pathway is composed by two alternative X-rays diagnostic test according to the type of illness of the patient: knee or back. These clinical pathways compete for the use of the X-ray diagnostic machine. Moreover, in order to improve the efficiency of the system, the synchronization of both clinical pathways is required in such a way that both teams perform diagnostic tests of

patient with the same condition: back or knee. The synchronization is based on the fact that the X-ray machine must be prepared for a type of diagnostic test. In this way, if two patients with the same condition are consecutively performed the diagnostic test, the time and cost are reduced.

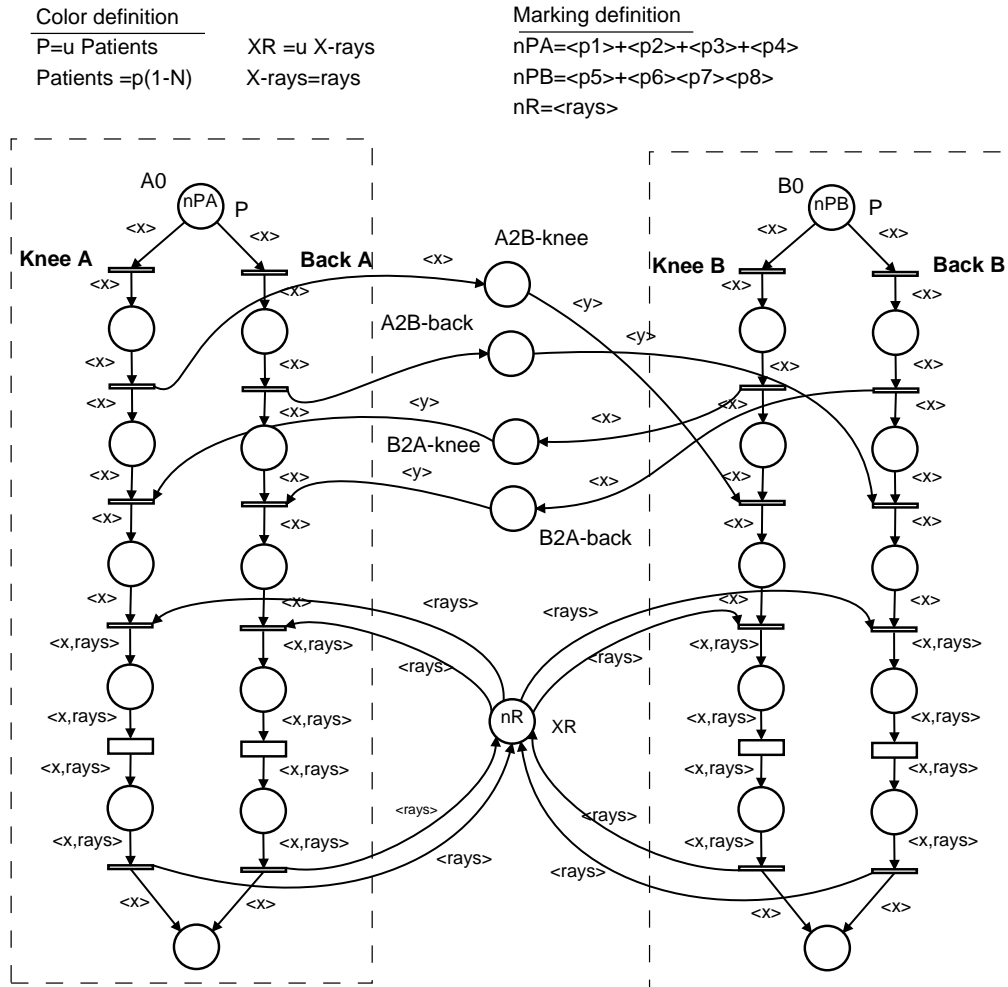


Figure 2.3: UML model of two clinical pathways that: i) interchanging information through places A2B-knee, A2B-back, B2A-knee and B2A-back; ii) competing for the shared resource X-ray machine.

We consider to extract from the UML in Fig. 2.3, the nets representing the two facets proposed in Sec. 2.2. First the facet of resource management is mod-

eled by a S⁴PR system and then, by a DSSP system, the facet of handshake between clinical pathways is considered. Fig. 2.4 shows the S⁴PR system ob-

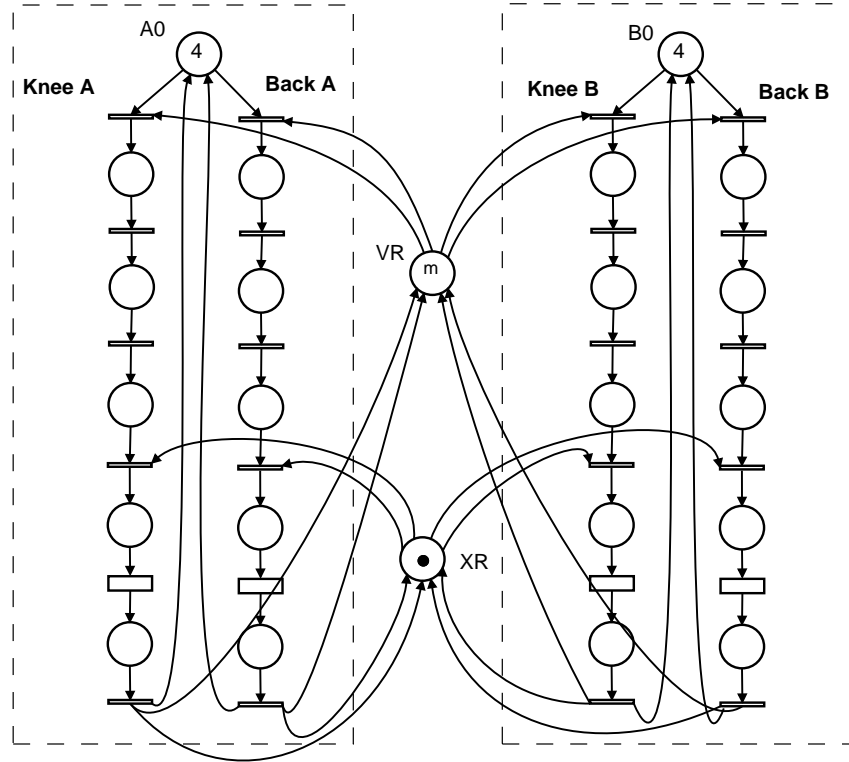


Figure 2.4: Resource management facet: S⁴PR system obtained from the UML in Fig. 2.3 by applying *Re1*, *Mo2*, *Ab3* and *Mo4*.

tained from the UML in Fig. 2.3. This net is obtained by applying *Re1*, *Mo2*, *Ab3* and *Mo4* explained in Sec 2.2.1. First the net is decoloured (*Re1*) by removing the colour domain of the places and removing the arcs inscriptions. After this, by applying *Mo2* each subnet modeling a clinical pathways is a strongly connected net. Then places modelling information channels (A2B-knee, A2B-back, B2A-knee and B2A-back) and their arcs are removed (*Ab3*). Finally a new virtual resource (place VR) is added to the net (*Mo4*). In this facet is possible to study the management of the resources and the performance of the system. In this case there is only one resource (x-ray machine) and the S⁴PR net is live.

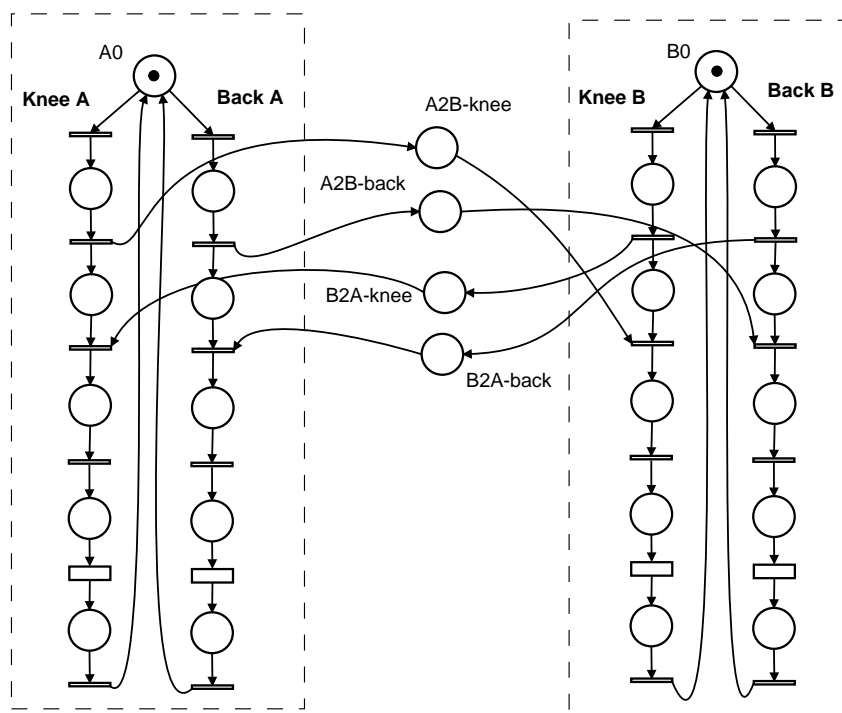


Figure 2.5: Handshake between clinical pathway facet: DSSP system obtained from the UML in Fig. 2.3 by applying *Re1*, *Mo2*, *Ab5* and *Mo6*.

In Fig. 2.5 can be seen the DSSP system obtained from the UML in Fig. 2.3. For that, *Re1*, *Mo2*, *Ab5* and *Mo6* (Sec. 2.2.2) have been sequentially applied. Like in the previous facet, first the net is decoloured (*Re1*) and then a strongly connected net is obtained (*Mo2*). In this facet, the interest is the communication protocol, so the place (XR) modeling a resource and its arcs are removed (*Ab5*). Finally, since each information place (A2B-knee, A2B-back, B2A-knee and B2A-back) has only output transitions in one clinical pathway, *Mo6* does not introduce changes in the net structure. In this facet is possible to study the communication protocol between clinical pathways. In particular, applying structural net-level analysis techniques (rank theorem) it is possible to identify that this DSSP system is not-live and consequently the communication protocol is bad designed. If the clinical pathway on the left, perform the diagnostic test on a knee-patient (i.e., transition Knee A is fired) and the clinical pathway on the right perform the diagnostic test on a back-patient (i.e., transition Back

B is fired), the net reach a deadlock. It happens because the clinical pathways do not perform the diagnostic test on a patients with the same illness (back or knee) and consequently they cannot be synchronized. The structural reason come from the fact that each clinical pathway take independent decision before they synchronized. Adjustment is based on modifying the communication protocol so that one clinical pathway takes a non-free decision. That is, a pathway takes a free decision (the boss) and the other one must assume the decision previously taken (the follower). Assuming that team A is the boss, the output arc of the place A2B-knee is pre-assigned to the transition Knee B and the output arc of the place A2B-back is pre-assigned to the transition Back B. A formal procedure for liveness enforcement of DSSP net system based on buffers pre-assignment [19] is explained in Chapter 3.

2.4 Discussion

A healthcare system can be modeled as a set of clinical pathways interacting between them by sharing resources and interchanging health information. A *Unified Modeled Language* (UML) profile called *Healthcare System Specifications* (HSS) was proposed to describe healthcare system. Moreover, a *model to model* (M2M) transformation was given to obtain *Stochastic Well Formed* (SWN) nets from the UML-HSS profile. The SWN models allow representing the multifaceted-nature of the healthcare system. State based techniques or event simulations are normally used for their analysis, however, they have frequently time and space problems. In this chapter it is proposed a methodology based on a set of relaxations, abstraction and modifications that are applied to the SWN in order to obtain formal subclasses of *Petri Nets*. In these net subclasses, structural net-level analysis techniques can be applied to analyze qualitative properties. Moreover, easier quantitative analysis can be performed due to the lower size and lower complexity of the resulting nets. Each one of these subclasses of PN considers a particularity of the behavior of the healthcare system that are called *facets*. Two facets are obtained in this chapter: *resource management facet* and *handshake between clinical pathways facet*. In the first one S⁴PR are obtained while for the second facet *Deterministically Synchronized Sequential Process* (DSSP) are considered.

Chapter 3

Buffer pre-assignment method for liveness of DSSP

Summary

In the previous chapter, a facet of the healthcare systems that studies the communication protocols between clinical pathways was proposed. It was shown that this facet can be modeled by Deterministically Synchronized Sequential Processes (DSSP), a particular subclass of Petri Nets. DSSP are composed by a set of state machines (called also agents) cooperating through asynchronous message passing. The structure of DSSPs allows strong analytical results, for example, the rank theorem provides necessary and sufficient condition for structural liveness. This chapter considers a synthesis problem of liveness enforcing. For some particular structures of DSSP in which the rank theorem does not hold we provide a very easy and intuitive technique based on the pre-assignment of the buffers in order to ensure that the model becomes live. The buffer pre-assignment method is given in [19].

3.1 Starting point

3.1.1 Introduction

DSSP are commonly used to model distributed systems composed by cooperative multi-agents [56]. This cooperation is realized by a set of buffers to/from which the agents consume/produce (partial) products. There exist two constraints in the buffer assignment (see Def. 1.4): (1) *tokens from a given buffer can only be consumed by a particular agent* and, (2) *a buffer never conditions internal choices of its destination*. For healthcare systems, buffers model information channels while the agents (state machines) model the clinical pathways that must be followed by the patients. So, the tokens of these buffers model the availability or requirement of information to complete a clinical pathway.

Bad design in communication protocols between clinical pathways could lead to patients “trapped” in the middle of a pathway without the possibility to continue. This means that a patient cannot continue because some information is required and it has not been generated. This situation can be checked analyzing the DSSP net that models the healthcare system. Our motivation here comes from the liveness enforcement of non-live DSSP nets. In this way, bad design communication protocols could be corrected preventing the deadlock of the patients.

Structural liveness of DSSP has been studied in literature [56]. Rank theorem (see Def.1.6) gives a necessary and sufficient condition to characterize the structural liveness [66, 55], but up to our knowledge there exists no solution to the synthesis problem of liveness enforcement in this setting. The first idea for liveness enforcement is to use the siphon-based method frequently used in Resource Allocation Systems [29, 53, 23, 46, 9, 71]. However, we illustrate that by applying this method to DSSP, the new buffers (corresponding to the monitor places) in addition of conditioning internal choices of agents will not be anymore private to agents.

This chapter provides a different method which will keep the restriction of DSSP that the buffers are private to agents. The basic idea of the proposed method is that whenever a conflict transition of an agent is fired, at least one local T-semiflow can be fired until the end. That is, all input buffers of T-semiflows have enough tokens to fire their transitions. To get this, the buffers are pre-assigned at the conflict transitions. We show that this is not enough

and the global T-semiflows should be considered. Moreover, we provide an algorithm that for some structures of global T-semiflows ensures the structural liveness of the net system.

3.1.2 Controlling bad siphons in DSSP systems

A well-know method for liveness enforcement of ordinary nets consists in controlling the *bad siphons* (siphons without any trap). In this method, first the set of “bad siphons” is computed and then they are prevented to become empty by using for example, monitor places (as in the case of generalized mutual exclusion constraints [32]).

Definition 3.1. *Let $\Phi_k = \{p_1, \dots, p_n\}$ be a bad minimal siphon and let p_k be the monitor place that prevents the emptiness of Φ_k . The following hold:*

1. $C[p_k, t_j] = \sum_{l=1}^n C[p_l, t_j]; p_l \in \Phi_k;$
2. $m_0[p_k] = \sum_{l=1}^n m_0[p_l] - 1; p_l \in \Phi_k.$

However, this strategy applied to DSSP results in new buffers (the monitor places can be seen as new buffers) that in addition to condition the internal choices of agents may have output transitions in more than one agent. So, by controlling the bad siphons, the resulting PN is not a DSSP because conditions 2 and 3 of Def. 1.4 are not satisfied.

Example 3.2. *In ex. 1.7 was proved (by rank theorem) that the DSSP net in Fig. 1.2 is non-live. This net is shown with continuous arcs in Fig. 3.1. The bad siphons in this net are $\Phi_1 = \{p_1, p_3, p_2, p_6, b_2, b_3\}$ and $\Phi_2 = \{p_1, p_4, p_2, p_5, b_1, b_4\}$. The places p_{m1} and p_{m2} illustrated by discontinuous circles in Fig. 3.1 prevent the emptiness of the bad siphons, Φ_1 and Φ_2 respectively. The new places (buffers) p_{m1} and p_{m2} provide tokens to both agents \mathcal{N}_1 and \mathcal{N}_2 losing the distributedness property of the system.*

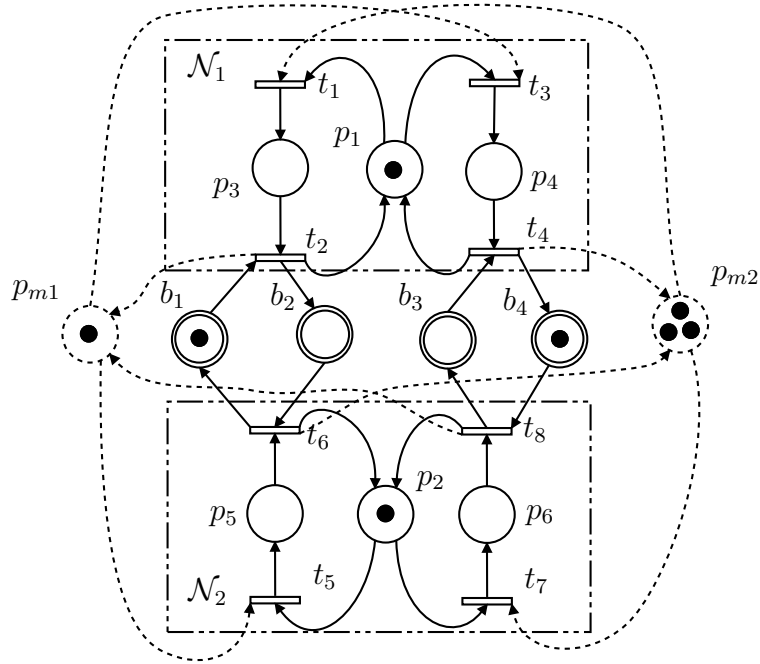


Figure 3.1: Motivation example: A not live DSSP net with the monitor places p_{m1} and p_{m2} that make the system live

3.1.3 The idea behind of buffer pre-assignment method

If a DSSP is consistent and conservative but the rank condition of the rank theorem is not satisfied, it is impossible to prevent livelock markings without losing the class of net. This is due to the constraint that prohibits buffers to influence internal choices. However, we would like to keep in the controlled net the second property of Def. 1.4 because of the possible distributed locations of the agents.

Knowing that the decision should be influenced by the buffers, in this chapter we propose a technique based on the idea of pre-assigning the buffers at the transitions in EQ relation. The main idea of this method is to ensure that when a conflict transition is fired, at least one local T-semiflow that contains the corresponding conflict transition can be fired completely. The resulting controlled PN is not a DSSP but only condition 3 in Def. 1.4 (buffers cannot condition internal choices in the agents) is not satisfied, so the system remains distributed.

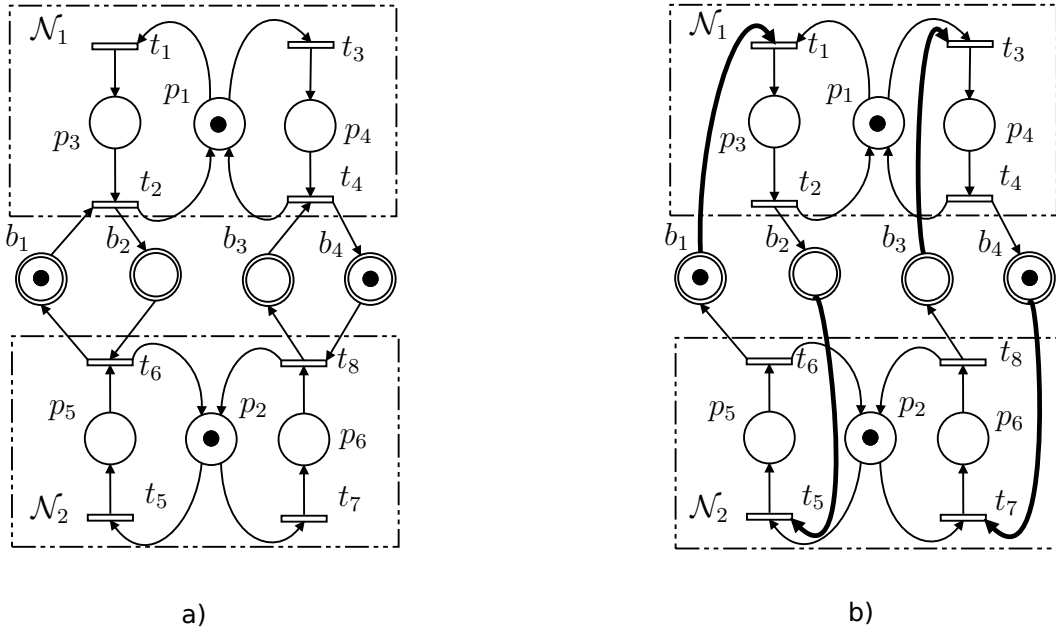


Figure 3.2: Motivation example: a) A not live DSSP net b) Resulting live system after buffer pre-assignment

Example 3.3. Let us consider the non-live DSSP in Fig. 3.2 (a). After pre-assigning the buffer, the live system in Fig. 3.2 (b) is obtained. In this net, we relax the third constraint of DSSP allowing buffers to condition internal choices. The buffer assignment are advanced from their original transition to the conflict transitions. For example, the buffer b_1 is an input buffer of the local T -semiflow $\mathbf{x}_1 = \{t_1, t_2\}$ because there is an arc from b_1 to t_2 . Before applying the buffer pre-assignment (Fig. 3.2 (a)), the local T -semiflow \mathbf{x}_1 can start firing t_1 although buffer b_1 is empty. The firing of t_1 could result in a block of agent \mathcal{N}_1 if b_1 is empty for any reachable marking. In order to prevent this situation we propose to advance the assignment of buffer b_1 from transition t_2 to transition t_1 . In this way only when b_1 has a token, the local T -semiflow \mathbf{x}_1 can be fired.

Notice that in Fig. 3.1 all the other buffers have been pre-assigned, i.e., b_2 is pre-assigned to t_5 , b_4 to t_7 and b_3 to t_3 . Table 3.1 shows the modifications done to transform the non-live DSSP into the live net system. The first two columns indicate the subnet and the T -semiflow in which the modifications have been

performed. The third and fourth columns are the removed and added arcs.

Table 3.1: Changes in the net in Fig. 3.1 (a) to obtain the net in Fig. 3.1 (b).

Agent	T-semiflow	Removed arc	New arc
\mathcal{N}_1	$\mathbf{x}_1 = \{t_1, t_2\}$	(b_1, t_2)	(b_1, t_1)
\mathcal{N}_1	$\mathbf{x}_2 = \{t_3, t_4\}$	(b_3, t_4)	(b_3, t_3)
\mathcal{N}_2	$\mathbf{x}_3 = \{t_5, t_6\}$	(b_2, t_6)	(b_2, t_5)
\mathcal{N}_2	$\mathbf{x}_4 = \{t_7, t_8\}$	(b_4, t_8)	(b_4, t_7)

The DSSP in Fig. 3.2 (a) has a simple net structure (two agents and four local T-semiflows) where the buffer pre-assignments is immediate and it can be made in a intuitive way. In the next subsection, we show more complex DSSP net structures where buffers pre-assignment is not easy to be done, being necessary to follow an algorithmic methodology.

3.2 Buffer pre-assignment approach

In this section it is given a buffer pre-assignment algorithm that forces the liveness in some DSSP net structures initially not live. First, some definitions are given (Sec. 3.2.1), then the pre-assignment from a local vs. a global perspective is discussed (Sec. 3.2.2) and finally the algorithm is proposed (Sec. 3.2.3).

The flowchart in Fig. 3.3 shows the steps to follow in order to get a live DSSP. First, the net structure \mathcal{N} must be consistent and conservative. If \mathcal{N} satisfies the rank theorem, there exists a live initial marking \mathbf{m}_0 . On the contrary, it will be necessary to apply Alg. 2 to obtain an alternative but live system. However, the resulting net after the application of Alg. 2 is not necessarily live and an additional step will be required in some cases. In this chapter we focus only on the definition of the algorithmic methodology and on the identification of the type of DSSP structures to which the algorithm can be applied.

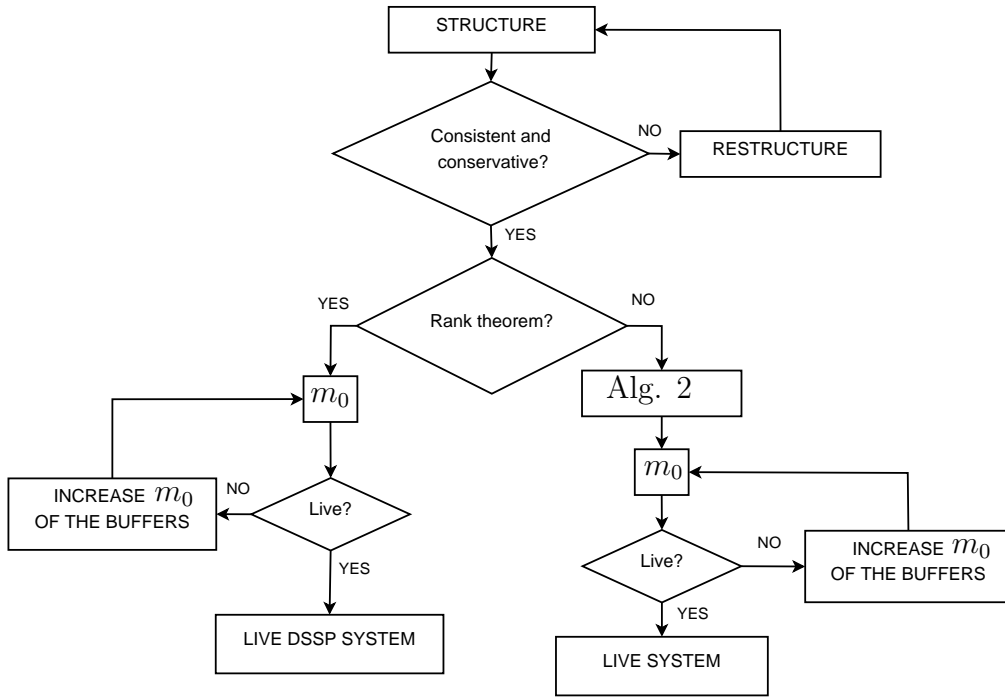


Figure 3.3: Liveness check in DSSP.

3.2.1 Definitions

Before explaining how to pre-assign the buffers, it is necessary to introduce some notation and some preliminary definitions and results.

In this chapter, we formalize the definition of two kinds of T-semiflows in a DSSP: *local* and *global*.

Definition 3.4. *Let \mathcal{N} be a DSSP (Def. 1.4) composed by n agents $\mathcal{N}_1, \mathcal{N}_2, \dots, \mathcal{N}_n$. A T-semiflow of \mathcal{N}_i is called local T-semiflow of agent i while a T-semiflow of \mathcal{N} is called global T-semiflow.*

Def. 3.4 states local T-semiflow as those T-semiflows of an agent, i.e., without considering the buffers and their arcs. On the contrary, global T-semiflows are computed considering all places and arcs of the DSSP net.

Property 3.5. *If \mathbf{x}_g is a global T-semiflow, then it can be written as a linear combination of two or more local T-semiflows, namely, $\mathbf{x}_g = \alpha \cdot \mathbf{x}_1 + \beta \cdot \mathbf{x}_2 + \dots + \gamma \cdot \mathbf{x}_n$ where $\alpha, \beta, \dots, \gamma \in \mathbb{R}_{\geq 0}$ and $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n$ are local T-semiflows.*

Local T-semiflows are not necessarily global T-semiflows. We consider that a global T-semiflow is composed by two or more local T-semiflows.

Moreover, for each agent we define a waiting place as a common input place of all its local T-semiflows.

Definition 3.6. *Let \mathcal{N}_i be an agent of a DSSP \mathcal{N} , let $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n$ be the minimal (local) T-semiflows of \mathcal{N}_i and let $\bar{P}_i = \left(\bigcap_{n=1}^k \bullet \|\mathbf{x}_n\| \right) \cap P_i$ be the set of common places of the local T-semiflows, where P_i is the set of places of \mathcal{N}_i (i.e., without the buffer places)*

- *If $k = 1$ (i.e., in the case of only one T-semiflow) the waiting place is the marked place at the initial marking (which is unique according to Def. 1.4).*
- *If $k \geq 2$ and $\bar{P}_i \neq \emptyset$, place $p_e \in \bar{P}_i$ is called waiting place if there exists no direct path in \mathcal{N}_i from p_e to other places in $\bar{P}_i \setminus \{p_e\}$.*
- *If $k \geq 2$ and $\bar{P}_i = \emptyset$ there does not exist a waiting place in \mathcal{N}_i .*

In DSSP structures obtained applying the methodology in Chapter 2, there always exists the waiting place. This happens because each clinical pathway defined as HSS profile has an initial and a final node. In the M2M transformation (R1 and R5) the initial and final node are transformed to SWN places. Finally, these places are decolored (Re1) and merged in a unique place (Mo2) from where start all possible paths that can be followed in a clinical pathway.

Example 3.7. *Let us consider the DSSP in Fig. 3.4 containing all continuous and dotted arcs. The DSSP is composed by two agents \mathcal{N}_1 and \mathcal{N}_2 , both having 4 local T-semiflows. Moreover, the global net has also 4 T-semiflows, all of them given in Tab. 3.2. Notice that the global T-semiflow \mathbf{x}_1 is composed by \mathbf{x}_5 (local T-semiflow of \mathcal{N}_1) and \mathbf{x}_9 (local T-semiflow of \mathcal{N}_2), i.e., $\mathbf{x}_1 = \mathbf{x}_5 + \mathbf{x}_9$. Additionally, $\mathbf{x}_2 = \mathbf{x}_6 + \mathbf{x}_{10}$, $\mathbf{x}_3 = \mathbf{x}_7 + \mathbf{x}_{11}$ and $\mathbf{x}_4 = \mathbf{x}_8 + \mathbf{x}_{12}$. Moreover, p_1 is the waiting place of \mathcal{N}_1 due to is the unique common input place of its local*

T-semiflows: $p_1 = \bar{P}_1 = \left(\bigcap_{n=5}^8 \bullet \|\mathbf{x}_n\| \right) \cap P_1$. Analogously it can be checked that the waiting place of \mathcal{N}_2 is p_8 .

Table 3.2: Local and Global T-semiflows of net in Fig 3.4

Id.	Net	Type	Transitions	Global
\mathbf{x}_1	\mathcal{N}	Global	t_1-t_6	-
\mathbf{x}_2	\mathcal{N}	Global	t_1, t_6-t_{10}	-
\mathbf{x}_3	\mathcal{N}	Global	$t_{11}-t_{16}$	-
\mathbf{x}_4	\mathcal{N}	Global	$t_{11}, t_{16}-t_{20}$	-
\mathbf{x}_5	\mathcal{N}_1	Local	t_1-t_3	\mathbf{x}_1
\mathbf{x}_6	\mathcal{N}_1	Local	t_1, t_9, t_{10}	\mathbf{x}_2
\mathbf{x}_7	\mathcal{N}_1	Local	$t_{11}-t_{13}$	\mathbf{x}_3
\mathbf{x}_8	\mathcal{N}_1	Local	t_{11}, t_{19}, t_{20}	\mathbf{x}_4
\mathbf{x}_9	\mathcal{N}_2	Local	t_4-t_6	\mathbf{x}_1
\mathbf{x}_{10}	\mathcal{N}_2	Local	t_6-t_7	\mathbf{x}_2
\mathbf{x}_{11}	\mathcal{N}_2	Local	$t_{14}-t_{16}$	\mathbf{x}_3
\mathbf{x}_{12}	\mathcal{N}_2	Local	$t_{16}-t_{18}$	\mathbf{x}_4

3.2.2 Local vs. global pre-assignment perspective

In the following we show that if the buffers are pre-assigned by looking only at the local T-semiflows the system could remain non live. Moreover, it will be necessary to add some arcs from transitions to buffers. The purpose of these new arcs is to update the marking of a buffer that has been pre-assigned to a transition of a local T-semiflow and finally this T-semiflow has not been fired.

Let us consider the DSSP in Fig. 3.4. After the pre-assignment of the buffers, the dotted arcs are removed and the discontinuous ones are added. In the left part of the net, composed by global T-semiflows \mathbf{x}_1 and \mathbf{x}_2 , the pre-assignment of the buffers have been carried out from a local perspective. This means, local T-semiflows have been pre-assigned without considering the global ones. Particularly, first the input buffers of the local T-semiflow \mathbf{x}_5 has

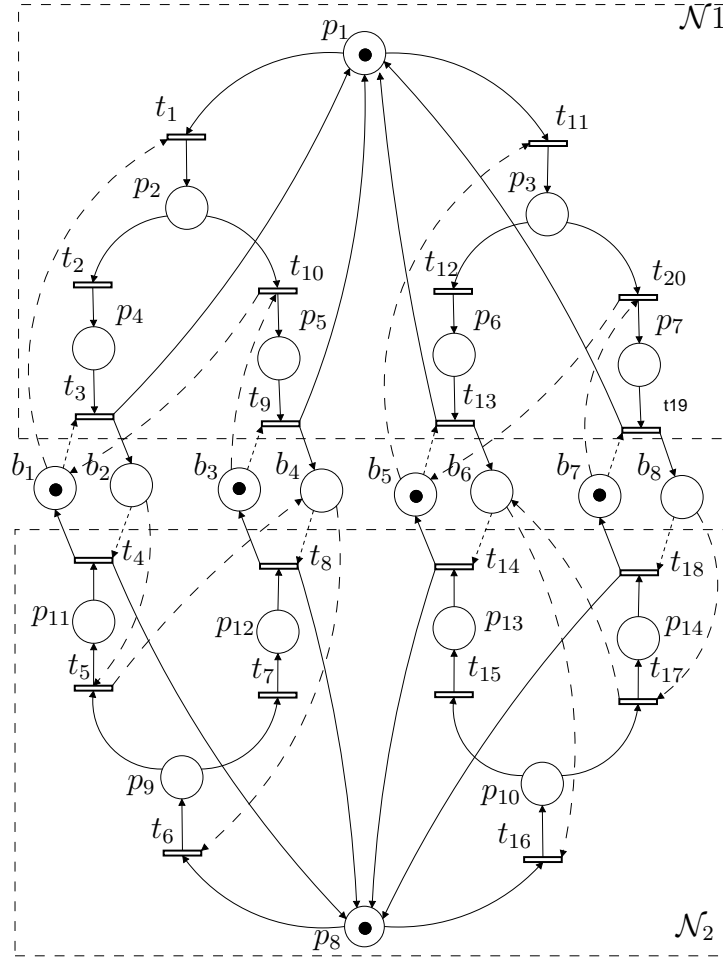


Figure 3.4: Pre-assignment from a local perspective vs. from a global one

been pre-assigned, then \mathbf{x}_6 , \mathbf{x}_{10} and \mathbf{x}_9 has been considered. The modifications performed are as follows:

- $\mathbf{x}_5 \rightarrow b_1$ is pre-assigned at t_1 ensuring that \mathbf{x}_5 starts to fire only if b_1 has a token. In order to do this, the arc (b_1, t_3) is replaced by (b_1, t_1) . Arc (t_{10}, b_1) is also added to send back the token to the buffer b_1 if the conflict at p_2 is solved through t_{10} .
- $\mathbf{x}_6 \rightarrow$ buffer b_3 is pre-assigned at transition t_{10} : the arc (b_3, t_9) is replaced by (b_3, t_{10}) .

- $\mathbf{x}_{10} \rightarrow$ buffer b_4 is pre-assigned at transition t_6 : the arc (b_4, t_8) is replaced by (b_4, t_6) . Arc from t_5 to b_4 is added.
- $\mathbf{x}_9 \rightarrow$ buffer b_2 is pre-assigned at t_5 : the arc (b_3, t_4) is replaced by (b_3, t_5) .

After the preassignment of \mathbf{x}_1 and \mathbf{x}_2 from a local T-semiflow perspective, the left part of the net can block. This happens if sequence $\sigma = t_1 t_2 t_3$ is fired at the marking in the figure. Notice that by firing σ , a token is removed from b_1 and a token is created in b_2 . Now, transitions t_1 and t_6 never can be fired since both have input empty places (buffers), i.e., $b_1 \in \bullet t_1$ and $b_4 \in \bullet t_6$.

In this case, the non-liveness of $\|\mathbf{x}_1\|$ and $\|\mathbf{x}_2\|$ appears because the pre-assignment of the buffers has been done by ensuring that only local T-semiflows finish but not global ones.

The right part of the DSSP net in Fig. 3.4 has been pre-assigned from a global perspective. This means, the local T-semiflows belonging to a global one are pre-assigned sequentially. Particularly in this case first the local T-semiflows (\mathbf{x}_7 and \mathbf{x}_{11}) composing the global T-semiflow \mathbf{x}_3 have been pre-assigned and then the local T-semiflows (\mathbf{x}_8 and \mathbf{x}_{12}) composing the global T-semiflow \mathbf{x}_4 . This part of the net is live because it will always be possible to fire \mathbf{x}_7 or \mathbf{x}_{11} , the local T-semiflows composing \mathbf{x}_3 . Moreover, buffers b_7 and b_8 will enable the firing of \mathbf{x}_8 and \mathbf{x}_{12} (local T-semiflows of \mathbf{x}_4), although \mathbf{x}_8 and \mathbf{x}_{12} are also conditioned by the possibility to fire \mathbf{x}_7 or \mathbf{x}_{11} respectively.

3.2.3 Buffer pre-assignment algorithm

Alg. 2 gives the steps to follow in order to pre-assign buffers in a DSSP. First, global T-semiflows, local T-semiflows and waiting places are computed. Then, for each local T-semiflow, the firing sequence of their transitions viewed from the waiting place is generated. After this, sequentially for each global T-semiflow \mathbf{x}_g , the following steps are considered. For each local T-semiflow \mathbf{x}_l composing the global one (\mathbf{x}_g), its input buffers are pre-assigned to the first transition t_k of its firing sequence σ_l that has not been pre-assigned before. The firing sequence σ_{ll} of each local T-semiflow \mathbf{x}_{ll} containing t_k is compared with the firing sequence σ_l of the local T-semiflow pre-assigned \mathbf{x}_l . The comparison of the sequences σ_l and σ_{ll} is made transition to transition from the common

transition t_k . The first transition t_p of σ_u that does not match, will send back tokens to the buffers pre-assigned.

Algorithm 2: Buffer preassignment in a DSSP net \mathcal{N}

- 1: Compute the set of local T-semiflows (X_L), the set of global T-semiflows (X_G) and the waiting places (p_e) of the net (\mathcal{N}).
 - 2: Generate the firing sequence σ_l of each local T-semiflow \mathbf{x}_l from its p_e .
 - 3: **for all** $\mathbf{x}_g \in X_G$ **do**
 - 4: **for all** local T-semiflow \mathbf{x}_l composing \mathbf{x}_g **do**
 - 5: The input buffers of \mathbf{x}_l (b_i s.t. $b_i^\bullet \cap \mathbf{x}_l \neq \emptyset$) are pre-assigned to the first transition t_k of the sequence σ_l that has not been pre-assigned before.
 - 6: Let \mathcal{N}_j be the state machine to which \mathbf{x}_l belongs
 - 7: **for all** local T-semiflow \mathbf{x}_u belonging to \mathcal{N}_j except \mathbf{x}_l **do**
 - 8: **if** $t_k \in \mathbf{x}_u$ **then**
 - 9: σ_l and σ_u are compared from t_k
 - 10: The first transition t_p of σ_u that does not match, will send back tokens to the buffers pre-assigned.
 - 11: **end if**
 - 12: **end for**
 - 13: **end for**
 - 14: **end for**
-

In the following we show how Alg. 2 is applied to the right part of the DSSP net (\mathcal{N}) in Fig 3.4. The global T-semiflows, the local T-semiflows and the waiting places of \mathcal{N} have been given in ex. 3.7. Moreover, Tab. 3.2 summarize the T-semiflows of \mathcal{N} . The firing sequence σ_i of each local T-semiflow \mathbf{x}_i starting from the corresponding waiting place p_e and the input/output buffers of \mathbf{x}_i are:

1. Firing sequences of the local T-semiflows in \mathcal{N}_1 starting from p_1
 - $\mathbf{x}_5 \Rightarrow \sigma_5 = \{t_1 t_2 t_3\} \Rightarrow b_1/b_2$ (input buffers/output buffers)
 - $\mathbf{x}_6 \Rightarrow \sigma_6 = \{t_1 t_{10} t_9\} \Rightarrow b_3/b_4$
 - $\mathbf{x}_7 \Rightarrow \sigma_7 = \{t_{11} t_{12} t_{13}\} \Rightarrow b_5/b_6$
 - $\mathbf{x}_8 \Rightarrow \sigma_8 = \{t_{11} t_{20} t_{19}\} \Rightarrow b_7/b_8$
2. Firing sequences of the local T-semiflows in \mathcal{N}_2 starting from p_8

- $\mathbf{x}_9 \Rightarrow \sigma_9 = \{t_6 t_5 t_4\} \Longrightarrow b_2/b_1$
- $\mathbf{x}_{10} \Rightarrow \sigma_{10} = \{t_6 t_7 t_8\} \Longrightarrow b_4/b_3$
- $\mathbf{x}_{11} \Rightarrow \sigma_{11} = \{t_{16} t_{15} t_{14}\} \Longrightarrow b_6/b_5$
- $\mathbf{x}_{12} \Rightarrow \sigma_{12} = \{t_{16} t_{17} t_{18}\} \Longrightarrow b_8/b_7$

First the pre-assignment is performed in \mathbf{x}_3 and then in \mathbf{x}_4 . The global T-semiflow $\mathbf{x}_3 = \mathbf{x}_7 + \mathbf{x}_{11}$, so the input buffer of both local T-semiflow \mathbf{x}_7 and \mathbf{x}_{11} must be pre-assigned. Let us start with \mathbf{x}_7 , its input buffer b_5 is pre-assigned from transition t_{13} to the first transition in σ_7 that has not been pre-assigned before (t_{11}). So the arc (b_5, t_{13}) is removed and a new arc (b_5, t_{11}) is added. Now, with the purpose to update the marking of the buffer b_5 if t_{11} is fired but finally the local T-semiflow \mathbf{x}_7 is not completely fired, we compare the firing sequence σ_7 with others firing sequences of local T-semiflows in \mathcal{N}_1 which contain t_{11} . The firing sequence σ_8 contain transition t_{11} , so $\sigma_7 = \{t_{11} t_{12} t_{13}\}$ is compared with $\sigma_8 = \{t_{11} t_{20} t_{19}\}$ starting from t_{11} . The next transition of the sequences does not match ($t_{12} \neq t_{20}$), so an arc (t_{20}, b_5) is added. Moreover, like \mathbf{x}_7 has been pre-assigned, their transition are removed from other firing sequences. In this case t_{11} is removed from σ_8 .

The modification performed during the preassignment algorithm can be showed graphically as follows: each color represents the changes in each global T-semiflow. A circle surrounding a transition indicates that there is an arc from the buffer of that color to the transition. A line on the transition indicates that the arc goes from the transition to the buffer. In case a transition is crossed out it means that it has already been pre-assigned. Red color represents the modification in \mathbf{x}_3 :

- $\mathbf{x}_5 \Rightarrow \sigma_5 = \{t_1 t_2 t_3\} \Longrightarrow b_1/b_2$
- $\mathbf{x}_6 \Rightarrow \sigma_6 = \{t_1 t_{10} t_9\} \Longrightarrow b_3/b_4$
- $\mathbf{x}_7 \Rightarrow \sigma_7 = \{\overset{\circ}{t_{11}} t_{12} t_{13}\} \Longrightarrow \overset{\circ}{b_5}/b_6$
- $\mathbf{x}_8 \Rightarrow \sigma_8 = \{t_{11} \overset{\circ}{\overline{t_{20}}} t_{19}\} \Longrightarrow \overset{\circ}{b_7}/b_8$

The pre-assignment of \mathbf{x}_{11} , the other local T-semiflow composing \mathbf{x}_3 (red color), is as follows:

- $\mathbf{x}_9 \Rightarrow \sigma_9 = \{t_6 t_5 t_4\} \Longrightarrow b_2/b_1$
- $\mathbf{x}_{10} \Rightarrow \sigma_{10} = \{t_6 t_7 t_8\} \Longrightarrow b_4/b_3$
- $\mathbf{x}_{11} \Rightarrow \sigma_{11} = \{\overset{\circ}{t_{16}} t_{15} t_{14}\} \Longrightarrow \overset{\circ}{b_6}/b_5$
- $\mathbf{x}_{12} \Rightarrow \sigma_{12} = \{t_{16} \overset{\circ}{t_{17}} t_{18}\} \Longrightarrow \overset{\circ}{b_8}/b_7$

Once all local T-semiflows composing \mathbf{x}_3 have been pre-assigned, the global T-semiflow $\mathbf{x}_4 = \mathbf{x}_8 + \mathbf{x}_{12}$ is pre-assigned. The modifications are indicated in green color.

Pre-assignment method applied to the DSSP in Fig. 2.5

In chapter 2 a clinical scenario composed by two clinical pathways interacting between them was given. Remember that each one of these clinical pathways models the workflow of a X-rays team (team A or team B) performing diagnostic tests on patients with back or knee illness. The DSSP net modeling the handshake between clinical pathway facet of the system is shown in Fig. 2.5. This DSSP is not live due to the fact that each clinical pathway (agent) take independent decisions before they synchronized.

Let us apply the buffer pre-assignment method in order to force the liveness of the DSSP system in Fig. 2.5. The resulting live system is shown in Fig. 3.5.

In this net, the buffers have been pre-assigned from their original transitions to the conflict transition of the local T-semiflow. Notice that the initial marking with token in the places B2A-knee and B2A-back make that team A takes a free decision between transition knee A or Back A (being the boss) while team B must assume the decision previously taken by team A (being the follower).

3.2.4 Limitations of the approach

The pre-assignment method applied by Alg. 2 does not work with general DSSP structures. In this subsection, we state the DSSP structure net under which, the application of Alg. 2 leads to a live system. With this purpose, a special type of global T-semiflows called TSF_1 is introduced in Def. 3.8.

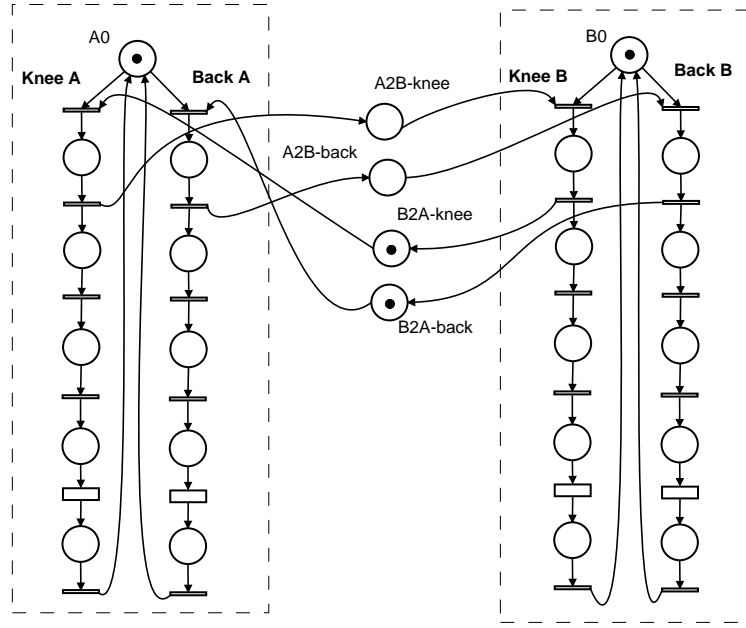


Figure 3.5: Resulting system after applying pre-assignment method to system in Fig. 2.5

Definition 3.8. Let \mathcal{N} be a DSSP composed by n agents $\langle \mathcal{N}_1, \mathcal{N}_2, \dots, \mathcal{N}_n \rangle$. A global T -semiflow \mathbf{x}_g is an TSF_1 if it fulfills the following conditions:

1. It is a linear combination of two or more local T -semiflows: $\mathbf{x}_g = \alpha \cdot \mathbf{x}_1 + \beta \cdot \mathbf{x}_2 + \dots + \gamma \cdot \mathbf{x}_l$ where $\alpha, \beta, \dots, \gamma \in \mathbb{R}_{\geq 0}$ and $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_l \in \bigcup_{i=1, \dots, n} STF_L^i$;
2. \mathbf{x}_g is composed by maximum one local T -semiflow per agent;
3. $\forall b \in (\bullet \|\mathbf{x}_g\| \cup \|\mathbf{x}_g\| \bullet) \Rightarrow (\bullet b \cup b \bullet) \in \|\mathbf{x}_g\|$.

■

A global T -semiflow of type TSF_1 ensures that tokens of their input/output buffers can only be consumed/produced by transitions belonging to their local T -semiflows. Moreover, if \mathcal{N} is consistent and conservative, there exists a firing sequence of their local T -semiflows that allows to fire TSF_1 completely. If the input buffers of all local T -semiflows composing the global one (TSF_1) are

pre-assigned to the first transitions of the sequence viewed from the waiting place, we are ensuring that:

1. There is an initial marking \mathbf{m}_0 such that for any evolution of the system, at least a local T-semiflow belonging to the TSF_1 will be enabled.
2. All local T-semiflows of TSF_1 will be enabled in some evolution of the system.
3. Only those local T-semiflows belonging to the TSF_1 whose input buffers have enough tokens to be fired completely, can start the sequence.

Lemma 3.9. *Given a consistent and conservativeness DSSP structure in which all global T-semiflows are STF_1 , by applying Alg. 2 to all of them, the new net system is structurally live.*

Proof. Let \mathbf{x}_g be a global T-semiflow of type STF_1 defined in Def. 3.8 and let \mathbf{x}_i be the local T-semiflows s.t $\mathbf{x}_g = \alpha \cdot \mathbf{x}_1 + \beta \cdot \mathbf{x}_2 + \dots + \gamma \cdot \mathbf{x}_n$. Applying Alg. 2 to \mathbf{x}_g we ensure that when a pre-assigned transition $t \in \|\mathbf{x}_i\|$ in equal conflict is fired, all transition belonging to $\|\mathbf{x}_i\|$ can be fired. It is due that all restriction of $\|\mathbf{x}_i\|$ has been pre-assigned to transition t . In this way, always a state machine firing a transition in conflict, can be return to the initial marking. In addition, the firing of $\|\mathbf{x}_i\|$ implies that the marking of $b \in \|\mathbf{x}_i\|^\bullet$ increases and consequently other local T-semiflow $\|\mathbf{x}_j\|$ s.t $b \in \bullet\|\mathbf{x}_i\|$ are enabled. \square

Unfortunately, if a global T-semiflow is not TSF_1 , then by applying Alg. 2, the resulting net could remain blocked.

For example, let us consider Fig. 3.6. The structure with the fine and discontinuous arcs is a DSSP modeling the communication protocol between preoperative anesthesia tests and the surgery day. It is formed by two agents, the upper one represents the clinical pathway of the preoperative anesthesia test while the other one models the surgery day. Notice that different anesthesia tests are performed on elective and urgent patients. Furthermore, the test on elective patients depends on the type of anesthesia that the surgery requires: local or general. In the surgery day pathway, elective and urgent surgeries can be performed, moreover, the elective surgeries can be performed under local or general anesthesia.

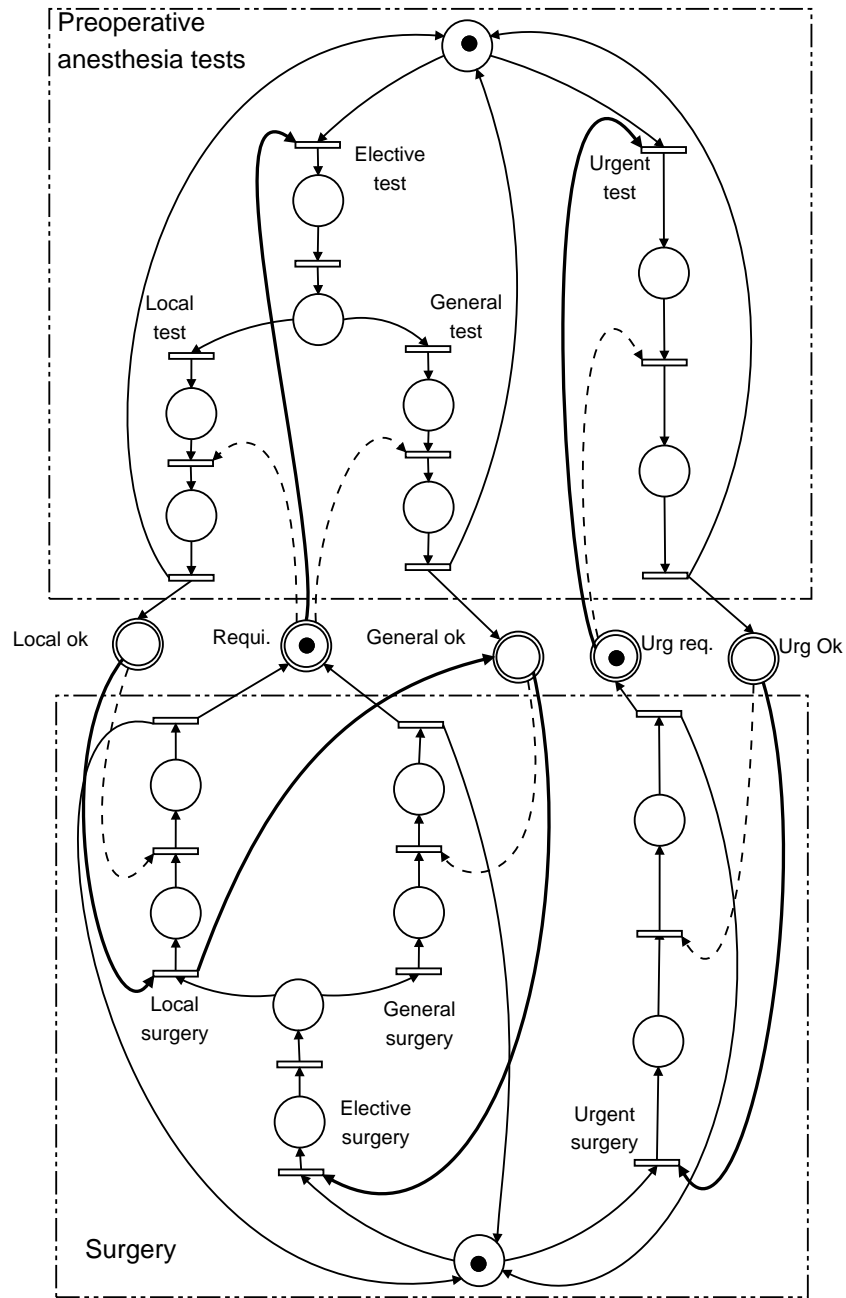


Figure 3.6: Resulting non live system after applying pre-assignment method to a healthcare DSSP system

The DSSP structure has five buffers. Two of them representing the requirement of the anesthesia tests on elective (“Requi”) or urgent (“Urg. req”) patients while the others three represent that a patients has successfully passed the anesthesia test. Particularly, a token in the place “local ok” (“general ok”) means that a patient is apt for an elective surgery under local (general) anesthesia while “Urg. ok” means that an urgent patient is apt for the surgery.

The DSSP system in Fig. 3.6 is not live because each clinical pathway (agent) can take independent decision before they synchronized. For example, from the initial marking, if a preoperative local-anesthesia test for an elective patient is performed in the upper agent and then, in the second agent, a elective surgery requiring general anesthesia try to be perform, the left part of the net (elective surgeries) is blocked.

Applying the pre-assignment method to the DSSP in Fig.3.6, the discontinuous arcs are removed and the grosser ones are added. However, the mentioned method does not force the liveness in this kind of DSSP structure. Particularly, from the initial marking and performing in the upper agent a preoperative local-anesthesia test a token is consumed from “Requi” place and a token is produced in the “local ok” place. However, an elective surgery under local anesthesia cannot be performed in the second agent due to the transition “elective surgery” is conditioned by the “General ok” place which is empty. So, the left part of the net (elective surgery) is blocked. In chapter 4 this problem is analyzed and approached by a new more robust method.

3.3 Discussion

Deterministically synchronized sequential processes (DSSP) are modular subclasses of P/T systems used for modeling and analysis of systems formed by asynchronous cooperating sequential processes. The availability of structural results represents a crucial advantage for the analysis of net models. Specifically, by the rank theorem it is possible to analyze the structural liveness of the DSSP net. In this chapter we studied a synthesis problem of liveness enforcement based on buffers pre-assignment. It is shown that the standard method of controlling bad siphons will introduce non-private buffers in the model. In some cases, in particular when the system is physically distributed, this constraint is desired. We provide an algorithm based on the pre-assignment of buffers in the

local T semiflows. We show that the pre-assignment must be performed from a global T semiflow perspective. For a particular global T-semiflow structure we prove that this pre-assignment ensures structural liveness of the system. However, for a general structure of global T semiflows, it could be necessary to execute additional steps after the pre-assignment. Next chapter proposes a new approach that allows to ensure liveness in more general DSSP systems.

Chapter 4

DSSP liveness enforcement through a control PN

Summary

In this chapter we provide a more general approach (than buffer preassignment) for liveness enforcing in non-live DSSP. The method is formalized on two levels: execution and control. The execution level uses the original DSSP structure while, for the control level we compute a new net system called the *control PN*. This net system is obtained from the original DSSP and has a predefined type of structure. The control PN will evolve synchronously with the non-live DSSP ensuring that the deadlock states will not be reached. The states (marking) of the control PN will enable or disable some transitions in the original DSSP, while some transitions in the control PN should fire synchronously with some transitions of the original DSSP. The results in this chapter are available in [18].

Chapter Nomenclature

$$\begin{aligned} \mathcal{N}^\alpha &= \langle P^\alpha, T^\alpha, \mathbf{Pre}^\alpha, \mathbf{Post}^\alpha \rangle && \text{is a PN} \\ \Sigma^\alpha &= \langle \mathcal{N}^\alpha, \mathbf{m}_0^\alpha \rangle && \text{is a PN system} \end{aligned}$$

Where,

$$\alpha = \begin{cases} d \rightarrow & \text{original (non-live) DSSP system} \\ c \rightarrow & \text{control system} \\ cs \rightarrow & \text{simplified control system} \\ + \rightarrow & \text{simplified control system with} \\ & \text{control buffers} \\ e \rightarrow & \text{equivalent controlled system} \end{cases}$$

p_e	the waiting place of an agent (if exists)
\mathbf{x}	a T-semiflow
t_{ch}^j	the check transition of a T-semiflow \mathbf{x}_j (if exists)
T^e	the set of enabled transitions in \mathcal{N}^d
T^f	the set of transitions that can be fired (control enabled) in \mathcal{N}^d
T_{ck}	the set of check transitions in a subnet \mathcal{N}_i^{cs} of the simplified control net \mathcal{N}^{cs}

4.1 Introduction

In the previous chapter we proposed a technique based on the pre-assignment of the buffers in order to enforce the liveness of DSSP systems. This method keeps the restriction that the buffers are destination private to agents, so it remains a distributed model. Unfortunately, this technique is not applicable for general constructions, where the resulting net after the defined pre-assignment method could be not live.

4.1.1 A two level approach for DSSP liveness enforcement

In order to provide a more general method to ensure structural liveness of DSSP, in this chapter we propose a new approach based on two net-system levels: execution and control.

1. At the execution level, the original DSSP system evolves conditioned by the control level. The firing of the transitions in the DSSP system is constrained by the state (marking) of the control level. In this way, transitions which may lead to total or partial deadlocks are inhibited.
2. At the control level, we use a net system obtained from the DSSP. This is called the *control PN* and its net has a predefined type of structure. The control PN in addition to inhibit the firing of transitions in the DSSP, also will evolve synchronously with the DSSP net.

The control policy proposed in this chapter is initially based on two main ideas: i) a local T-semiflow only can start firing if its input buffers have enough tokens to fire all its transitions and ii) when the firing of a local T-semiflow starts, all its transitions must be fired. Furthermore, we present an algorithm to obtain the control PN from a DSSP. This control PN models the consumption/production relation between buffers and local T-semiflows of the DSSP in such way that:

- for each local T-semiflow in the DSSP there is a sequence $t_a \rightarrow p \rightarrow t_b$ in the control PN.
- for each buffer in the DSSP net there is an homologous buffer in the control PN.
- if a local T-semiflow in the DSSP net consumes/produces items from/to a buffer, then in the control PN the corresponding sequence consumes/produces items from/to the homologous buffer.

In order to prove that the proposed methodology makes the DSSP live, it is necessary to ensure that the control PN is live. Nevertheless, In some cases this is not true, so we provide another algorithm that forces the liveness in the control PN by adding new constraints (places). These new places can be seen as new buffers with only one output transition. Consequently, the distributiveness of the DSSP systems is preserved.

Finally, we deal with the fusion of the DSSP with the control PN in order to obtain a unique controlled net system. The resulting net preserves the distributedness property as well.

4.1.2 How limitation of pre-assignment method are approached

The liveness enforcement technique for DSSP based on buffer pre-assignment is not working for general DSSP structures due to the complex relations that may appear between local and global T-semiflows. In the following the two main problems (**Pb. 1** and **Pb. 2**) of the pre-assignment method are stated and illustrated. Moreover, we give an intuition on how these problems will be approached in the new method.

Pb. 1. If DSSP net \mathcal{N}^d has not-disjoint global T-semiflows, could happen that after applying the pre-assignment of buffers, the firing of a local T-semiflow is conditioned by the marking of a buffer that, in \mathcal{N}^d , was not its input buffer. Most frequently this makes the system too restrictive and consequently non-live.

Example 4.1. Fig. 4.1(a) shows a DSSP \mathcal{N}^d while Fig. 4.1(b) is the resulting net after applying the pre-assignment algorithm (Alg. 2). The global and local T-semiflows of \mathcal{N}^d are given in Tab. 4.1 from where it is possible to check that \mathbf{x}_1 and \mathbf{x}_2 are not disjoint global T-semiflows ($\mathbf{x}_1 \cap \mathbf{x}_2 = \{t_1, t_5\}$).

Table 4.1: Local and Global T-semiflows of \mathcal{N}^d in Fig. 4.1(a)

Id.	Net	Type	Transitions	Global
\mathbf{x}_1	\mathcal{N}	Global	t_1-t_5	-
\mathbf{x}_2	\mathcal{N}	Global	t_1, t_5-t_8	-
\mathbf{x}_3	\mathcal{N}	Global	t_9-t_{12}	-
\mathbf{x}_4	\mathcal{N}_1	Local	t_1, t_2	\mathbf{x}_1
\mathbf{x}_5	\mathcal{N}_1	Local	t_1, t_8	\mathbf{x}_2
\mathbf{x}_6	\mathcal{N}_1	Local	t_9, t_{10}	\mathbf{x}_3
\mathbf{x}_7	\mathcal{N}_2	Local	t_3-t_5	\mathbf{x}_1
\mathbf{x}_8	\mathcal{N}_2	Local	t_5-t_7	\mathbf{x}_2
\mathbf{x}_9	\mathcal{N}_2	Local	t_{11}, t_{12}	\mathbf{x}_3

In the resulting net in Fig. 4.1(b), b_2 has been preassigned to t_5 , so the local T-semiflow $\mathbf{x}_7 = t_5 + t_4 + t_3$ can only start its firing when b_2 has a token. However, this pre-assignment is also conditioning the firing of the local T-semiflow $\mathbf{x}_8 = t_5 + t_6 + t_7$ since t_5 belongs to both \mathbf{x}_7 and \mathbf{x}_8 . So, in the net

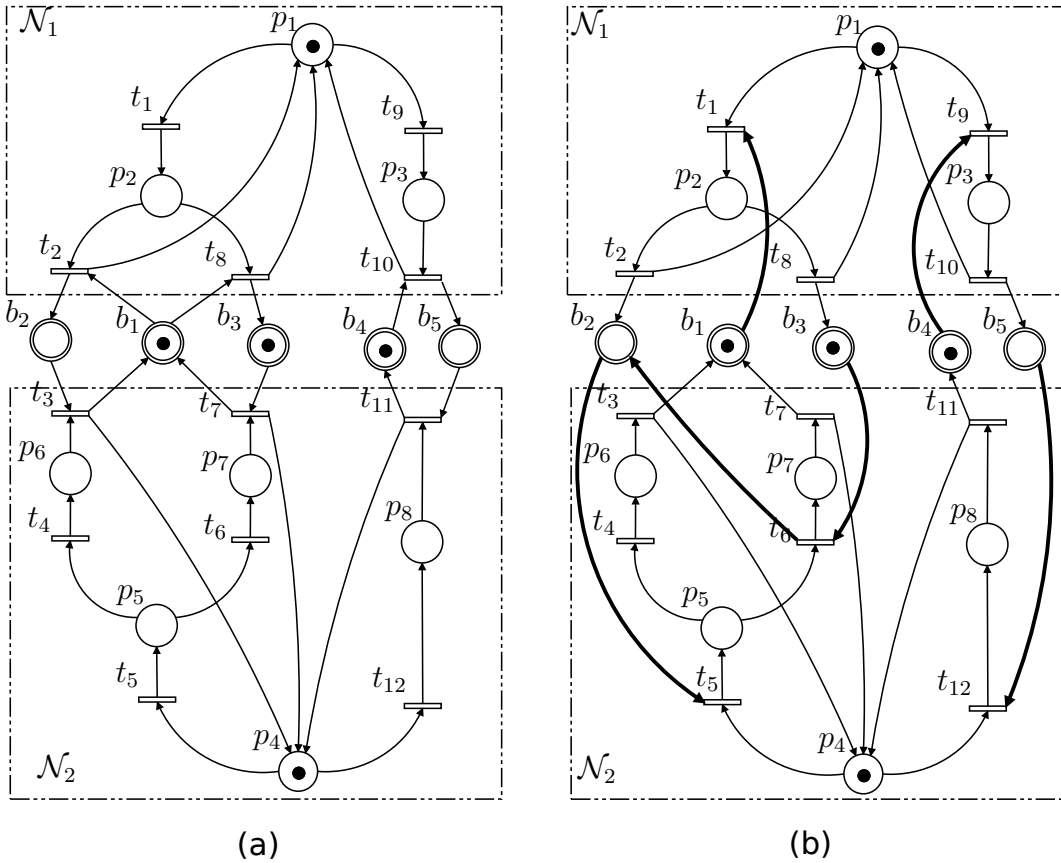


Figure 4.1: Motivation example of problem 1: (a) A non live DDSF net (\mathcal{N}^d). (b) The resulting non live net after applying the pre-assignment method in

in Fig. 4.1(b), the firing of the local T-semiflow \mathbf{x}_8 is conditioned by b_2 that in \mathcal{N}^d (Fig. 4.1(a)) was not its input buffer. This leads to block the execution of the global T-semiflows \mathbf{x}_1 and \mathbf{x}_2 . Particularly, from the initial marking $\mathbf{m}_0 = p_1 + p_4 + b_1 + b_3 + b_4$ and by firing the sequence $t_1 t_8$, $\mathbf{m}_1 = p_1 + p_4 + 2b_3 + b_4$ is reached. From this marking \mathbf{x}_1 and \mathbf{x}_2 cannot fire anymore.

In the approach proposed in this chapter, in order to avoid **Pb 1**, we obtain an equivalent control PN in which the local T-semiflows are made disjoint. In particular, for each local T-semiflow in \mathcal{N}^d , a sequence $t_a \rightarrow p \rightarrow t_b$ is added in the control PN. In this way, after the pre-assignment of the buffers in the control PN, the firing of a local T-semiflow only is conditioned by the buffers

that in \mathcal{N}^d were its input buffers.

Pb. 2. In the DSSP net a buffer has to choose between the firing of different local T-semiflows that subsequently require a synchronization. Only considering the pre-assignment method, the net could remain non live.

Example 4.2. Fig. 4.2(a) shows a \mathcal{N}^d where the pre-assignment method does not work due to **Pb. 2**. This \mathcal{N}^d has a global T-semiflow \mathbf{x}_1 composed by 3 local T-semiflows: $\mathbf{x}_2 = t_1 + t_2$, $\mathbf{x}_3 = t_3 + t_4$ and $\mathbf{x}_4 = t_5 + t_6$. The firing of $\mathbf{x}_2(\mathbf{x}_3)$ consumes a resource from b_1 and produces a resource in $b_2(b_3)$. The firing of \mathbf{x}_4 consumes a resource from b_2 and one from b_3 and produces two resources in b_1 . So, \mathbf{x}_2 , \mathbf{x}_3 and \mathbf{x}_4 should be fired proportionally.

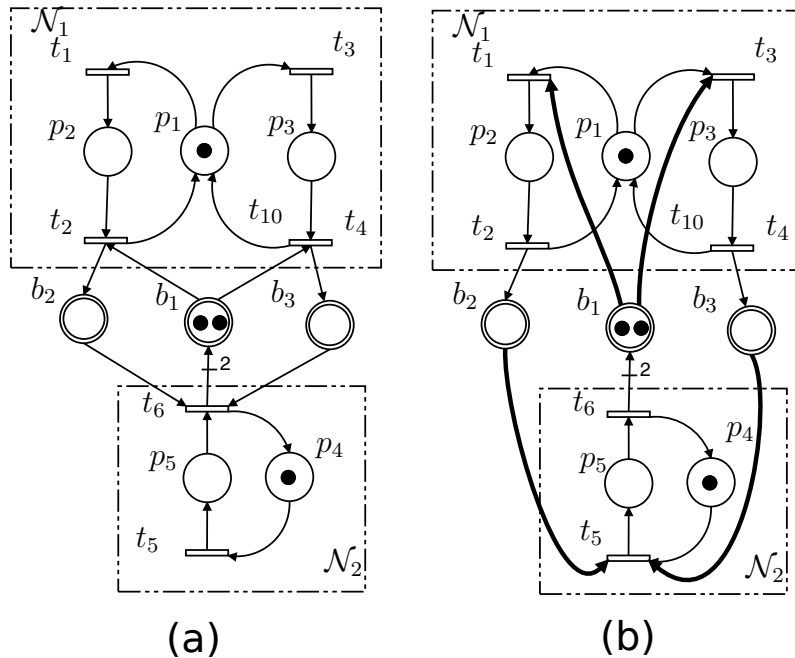


Figure 4.2: Motivation example of problem 2: (a) A non live \mathcal{N}^d . (b) The resulting non live net after applying the pre-assignment method

After applying the pre-assignment method to \mathcal{N}^d in Fig. 4.2(a), the non-live net in Fig. 4.2(b) is obtained. Here it can be seen that \mathbf{x}_1 or \mathbf{x}_2 can be fired twice without a firing of \mathbf{x}_3 . Consequently, the resulting net is not live.

In order to overcome **Pb. 2** in the new liveness enforcement approach, we propose to include in the control PN new buffers that can be seen like information buffers. The main objective of these new buffers is to force some global T-semiflows to fire their local ones in the correct proportion. These new buffers will only have output transitions in one local T-semiflow keeping the distributed property of the system.

4.1.3 Overview of the proposed approach

Let us approach a new method with the idea of *buffer pre-assignment* as the starting point. However, unlike in chapter 3, the pre-assignment here is not performed in \mathcal{N}^d , but it is performed in an *equivalent control PN* denoted as \mathcal{N}^c . This \mathcal{N}^c is obtained from \mathcal{N}^d and has a predefined type of structure in which the local T-semiflows are disjoint. The main advantage of performing the pre-assignment in \mathcal{N}^c with disjoint local T-semiflows is the fact that the firing of local T-semiflows will only be conditioned by buffers that in \mathcal{N}^d were its input buffers. In this way, **Pb 1** of pre assignment method is prevented. Moreover, we propose to include new buffers (seen as information buffers) that force some global T-semiflows to fire their local ones in the correct proportion. These new buffers ensure the liveness of \mathcal{N}^c preventing **Pb. 2**.

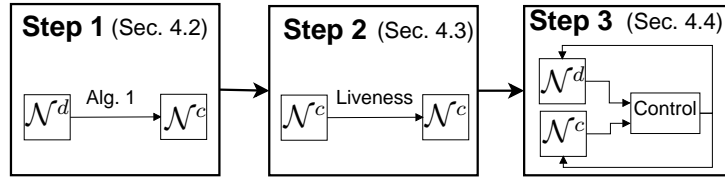


Figure 4.3: Overview of the liveness enforcement methodology

From a high level view (see Fig. 4.3), the proposed liveness methodology for a non-live consistent and conservative DSSP structure (\mathcal{N}^d) consists of:

- **Step 1:** compute the equivalent control PN (\mathcal{N}^c). \mathcal{N}^c has a predefined type of structure and models the consumption/production relation between the buffers and the local T-semiflows of \mathcal{N}^d . Sec. 4.2 explains how \mathcal{N}^c is obtained from \mathcal{N}^d by Alg. 3.

- **Step 2:** *ensure the liveness of \mathcal{N}^c .* To force the liveness in \mathcal{N}^d system through \mathcal{N}^c , the control net \mathcal{N}_c must be live. In Sec.4.3 the possible structures of \mathcal{N}^c obtained after applying Alg. 3 are characterized. In some of them, the liveness holds while in others the liveness is forced by adding some control places (Alg.4).
- **Step 3:** *control policy and systems evolution.* \mathcal{N}^c will evolve synchronously with \mathcal{N}^d disabling transitions that may lead the system to livelock. The methodology and behavior is described in Sec. 4.4, basically consisting in,
 - **3.1** *Label the transitions of the \mathcal{N}^c* with names of transitions of the \mathcal{N}^d . These common labels allow the firing of transitions in the \mathcal{N}^c synchronously with some transitions of the \mathcal{N}^d .
 - **3.2** *Assign guard expressions* to transitions of \mathcal{N}^d . These guard expressions are logical conditions based on the marking of \mathcal{N}^c and will disable the firing of the transitions that may lead to a livelock.
 - **3.3** Events from \mathcal{N}^d are input signals to \mathcal{N}^c .

4.2 Construction of the Control PN

In Section 4.1.2 we showed that buffer pre-assignment in a consistent and conservative DSSP net \mathcal{N}^d with non-disjoint local T-semiflows (**Pb. 1**), could result in non live systems. To prevent this situation, we propose to use an equivalent control PN (\mathcal{N}^c) with a predefined type of structure in which the local T-semiflows are made disjoint. The objective is to pre-assign the buffers in \mathcal{N}^c , so that a local T-semiflow in \mathcal{N}^c can only be fired if its input buffers have enough tokens. Subsequently and by means of guard expressions, the firing of local T-semiflows in \mathcal{N}^d will be conditioned by the state (marking) in \mathcal{N}^c . Both nets will evolve synchronously.

\mathcal{N}^c is obtained from \mathcal{N}^d and models the consumption/production relation that exists between the buffers and local T-semiflows of \mathcal{N}^d . Moreover, \mathcal{N}^c has the same number of agents and the same buffers that \mathcal{N}^d . However, each local T-semiflow in \mathcal{N}^d is modeled by an ordinary sequence $t_a \rightarrow p \rightarrow t_b$ in \mathcal{N}^c .

Where t_a (t_b) models the first (last) transition of the local T-semiflow (defined in the following).

Before giving the methodology to obtain the control PN (\mathcal{N}^c) from a DSSP structure (\mathcal{N}^d), let us state some concepts. For computing \mathcal{N}^c it is necessary that all agents composing \mathcal{N}^d have a waiting place p_e (Def. 3.6) from which the first and last transition of each local T-semiflow can be defined. The existence of a waiting place for each agent ensures that every cycle (local T-semiflow) contains a common input/output place (p_e) such that $p_e \in \bullet\|\mathbf{x}_n^i\| \cap \bullet\|\mathbf{x}_n^i\|$ for all T-semiflows \mathbf{x}_n^i of \mathcal{N}_i .

For example, in the subnet \mathcal{N}_1 of the DSSP net in Fig. 4.1(a) there are three T-semiflows: \mathbf{x}_4 , \mathbf{x}_5 and \mathbf{x}_6 given in Tab. 4.1. $\bullet\|\mathbf{x}_4\| \cap P_1 = \{p_1, p_2\}$, $\bullet\|\mathbf{x}_5\| \cap P_1 = \{p_1, p_2\}$ and $\bullet\|\mathbf{x}_6\| \cap P_1 = \{p_1, p_3\}$. So $\bar{P}_1 = p_1$ and consequently the waiting place is p_1 . However, the net in Fig. 4.4 represents an agent of a DSSP structure without a waiting place. This net have 3 minimal T-semiflow: $\mathbf{x}_1 = t_2 + t_3$, $\mathbf{x}_2 = t_1 + t_4 + t_5$ and $\mathbf{x}_3 = t_6 + t_7$. The input places of this minimal T-semiflow are the following: $\bullet\|\mathbf{x}_1\| = \{p_2, p_3\}$, $\bullet\|\mathbf{x}_2\| = \{p_1, p_2, p_4\}$ and $\bullet\|\mathbf{x}_3\| = \{p_1, p_5\}$. In this case, $\bar{P}_i = \emptyset$ and does not exist a waiting place.

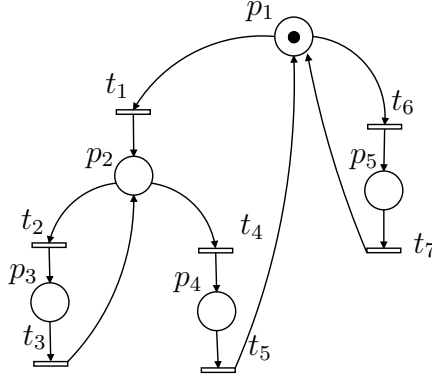


Figure 4.4: Agent without waiting place

The existence of a waiting place (p_e) for an agent \mathcal{N}_i allows us to define the first/last transition of a local T-semiflow \mathbf{x}_i^i as follows:

1. $t_{first}^i = p_e \bullet \cap \bullet\|\mathbf{x}_i^i\|$;
2. $t_{last}^i = \bullet p_e \cap \bullet\|\mathbf{x}_i^i\|$.

Algorithm 3: Obtain a control PN for a non-live DSSP structure

Require: DSSP structure $\mathcal{N}^d = \langle P^d, T^d, \mathbf{Pre}^d, \mathbf{Post}^d \rangle$
Ensure: Control PN $\mathcal{N}^c = \langle P^c, T^c, \mathbf{Pre}^c, \mathbf{Post}^c \rangle$

- 1: **for all** $b_i \in \mathcal{N}^d$ **do**
- 2: Add a place p_{b_i} to P^c
- 3: **end for**
- 4: **for all** agent \mathcal{N}_i of \mathcal{N}^d **do**
- 5: Add a place $p_{\mathcal{N}_i}$ to P^c
- 6: Compute all minimal T-semiflows of \mathcal{N}_i and save them to Γ_i
- 7: **for all** $\mathbf{x}_i^i \in \Gamma_i$ **do**
- 8: Add a transition t_j^l to T^c {representing t_{first}^l }
- 9: Add a transition t_k^l to T^c {representing t_{last}^l }
- 10: Add a place p_{x_i} to P^c {representing \mathbf{x}_i^i }
- 11: $\mathbf{Post}^c[p_{x_i}, t_j^l] = \mathbf{Pre}^c[p_{x_i}, t_k^l] = 1$
- 12: Let $T_i = \|\mathbf{x}_i^i\|$
- 13: **for all** b_i s.t $\mathbf{Pre}^d[b_i, T_i] \neq \mathbf{0}$ **do**
- 14: $\mathbf{Pre}^c[p_{b_i}, t_j^l] = \sum_{t \in T_i} \mathbf{Pre}^d[b_i, t]$ {assign input buffers to t_j^l }
- 15: **end for**
- 16: **for all** b_i s.t $\mathbf{Post}^d[b_i, T_i] \neq \mathbf{0}$ **do**
- 17: $\mathbf{Post}^c[p_{b_i}, t_k^l] = \sum_{t \in T_i} \mathbf{Post}^d[b_i, t]$ {assign t_k^l to output buffers}
- 18: **end for**
- 19: $\mathbf{Pre}^c[p_{\mathcal{N}_i}, t_j^l] = 1$
- 20: $\mathbf{Post}^c[p_{\mathcal{N}_i}, t_k^l] = 1$
- 21: **end for**
- 22: **end for**

Using Alg. 3, a control PN (\mathcal{N}^c) is obtained from a non-live DSSP structure (\mathcal{N}^d) as follows:

- Steps 1-3: for each buffer b_i in \mathcal{N}^d , a place p_{b_i} is introduced in \mathcal{N}^c ;
- Steps 4-5: for each agent in \mathcal{N}^d , a new place is added in \mathcal{N}^c ;
- Step 6-7: for each agent \mathcal{N}_i^d , compute all minimal T-semiflows \mathbf{x}_i^i ;
- Step 8-11: for each local T-semiflow \mathbf{x}_i an ordinary subnet $\{t_j^l \rightarrow p_{x_i} \rightarrow t_k^l\}$ is added in the \mathcal{N}^c ;

- Step 13-15: If a buffer b_i have an output transition in $\|\mathbf{x}_i^i\|$, then, in \mathcal{N}^c , there exists an arc from p_{b_i} to the first transition t_j^l with a weight equal to $\sum_{t \in \|\mathbf{x}_i^i\|} \mathbf{Pre}^d[b_i, t]$;
- Step 16-18: In case that a buffer b_i have an input transition in $\|\mathbf{x}_i^i\|$, then, in \mathcal{N}^c , there exist an arc from t_k^l to p_{b_i} with a weight equal to $\sum_{t \in \|\mathbf{x}_i^i\|} \mathbf{Post}^d[b_i, t]$;
- Step 19-20: the sequences introduced in \mathcal{N}^c for each \mathbf{x}_i^i are connected with the place $p_{\mathcal{N}_i}$ as follows: $\mathbf{Pre}^c[p_{\mathcal{N}_i}, t_j^l] = 1$ and $\mathbf{Post}^c[p_{\mathcal{N}_i}, t_k^l] = 1$

In order to reduce the number of local T-semiflows in \mathcal{N}^d and consequently the computational complexity of Alg.3, some typical reduction rules [64, 51] can be applied to \mathcal{N}^d before computing \mathcal{N}^c . First, fusion of places and fusion of transitions must be considered and then, identical transitions can be reduced to a unique transition.

Definition 4.3. *Two transitions t and t' are identical if $\mathbf{Pre}[P, t] = \mathbf{Pre}[P, t']$ and $\mathbf{Post}[P, t] = \mathbf{Post}[P, t']$.*

Example 4.4. *Let us consider the consistent and conservative DSSP net (\mathcal{N}^d) in Fig. 4.1(a). Applying Alg. 3 to \mathcal{N}^d , the equivalent control PN \mathcal{N}^c in Fig. 4.5 is obtained. First, since \mathcal{N}^d has 5 buffers, in \mathcal{N}^c the places p_{b_i} ($i = 1, 2, \dots, 5$) are added. Furthermore, since \mathcal{N}^d has two agents (\mathcal{N}_1 and \mathcal{N}_2), the places $p_{\mathcal{N}_1}$ and $p_{\mathcal{N}_2}$ are added in \mathcal{N}^c . After this, we apply steps 7-21 to all minimal T-semiflows of each agent. Let us consider for example the local T-semiflow $\mathbf{x}_7 = t_5 + t_4 + t_3$. To this T-semiflow, the ordinary sequence $\{t_5^7 \rightarrow p_{x_7} \rightarrow t_3^7\}$ is added in \mathcal{N}^c . Since in \mathcal{N}^d , b_2 is an input buffer of \mathbf{x}_7 , in \mathcal{N}^c there is an arc from the place p_{b_2} to the input transition t_5^7 . In addition, \mathbf{x}_7 has an output buffer b_1 so in \mathcal{N}^c there exists an arc from the output transition t_3^7 to the place p_{b_1} . Moreover, since $\|\mathbf{x}_7\|$ belongs to \mathcal{N}_2 , we add an arc from $p_{\mathcal{N}_2}$ to t_5^7 and from t_3^7 to $p_{\mathcal{N}_2}$. Finally, by applying Steps 7-21 to all local T-semiflows, we obtain the control PN in Fig 4.5.*

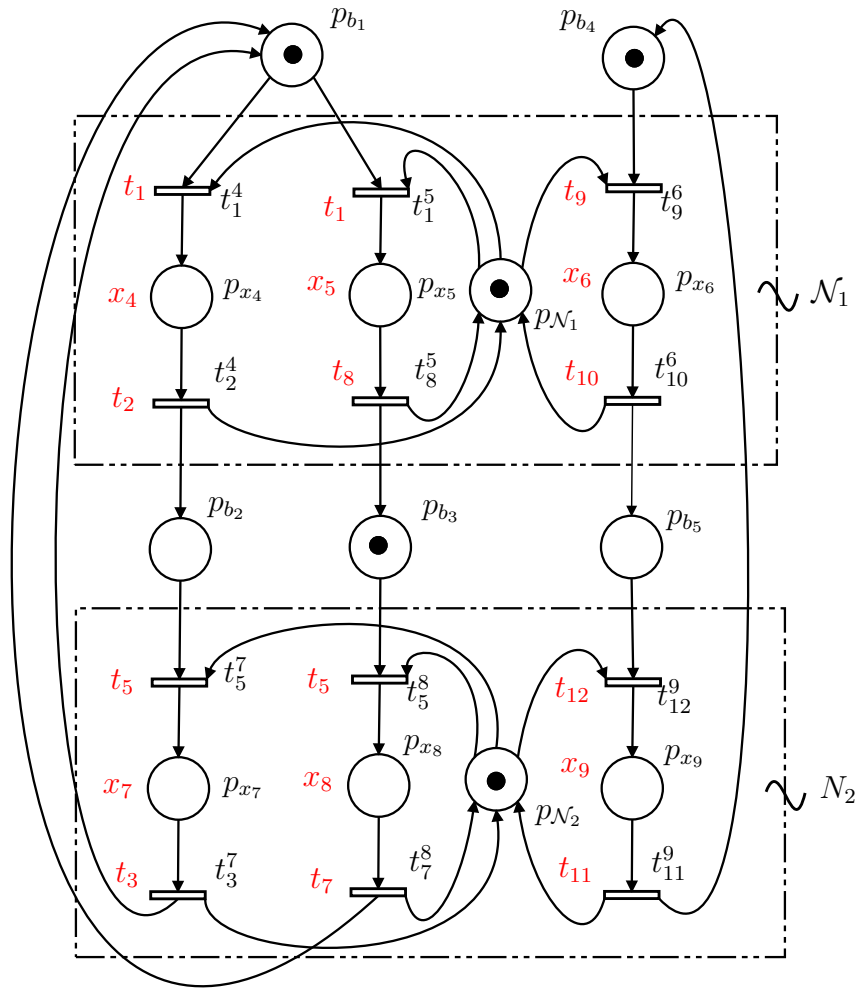


Figure 4.5: Control PN obtained from DSSP structure in Fig. 4.1(a)

4.3 Liveness of the Control PN

The control policy of the proposed liveness enforcement is explained with more details in section 4.4. Mainly, it is based on the idea that each transition in the control PN (\mathcal{N}^c) has associated a transition in the original DSSP structure (\mathcal{N}^d). Moreover, a transition in \mathcal{N}^d can be fired if it is enabled and the associated transition in \mathcal{N}^c is also enabled. So, if \mathcal{N}^d is controlled by a non live \mathcal{N}^c could result in a non live system. For that, the liveness of \mathcal{N}^c is a

necessary condition. In this Section, first we identify some structures of control PNs where the liveness is guaranteed (Sec. 4.3.1). Then, in the structures where the liveness can not be guaranteed, we propose a methodology to force the liveness of the control PN (Sec. 4.3.2). The diagram in Fig 4.6 shows the proposed methodology for ensuring or forcing liveness in the Control PN.

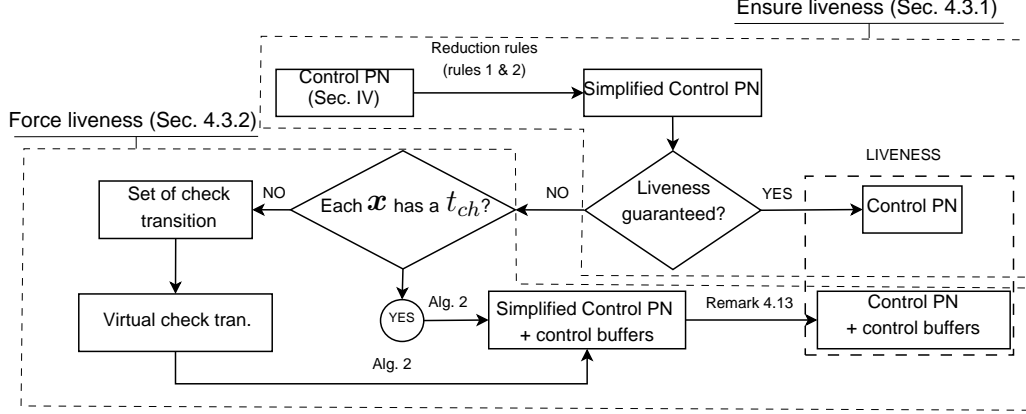


Figure 4.6: Diagram for ensuring liveness in the Control PN

4.3.1 Is it liveness guaranteed in the Control PN?

We identify some particular structures of the control PN where the liveness can be guaranteed. For these structures, after applying some reduction rules (that preserve liveness), a consistent and conservative Equal Conflict net is obtained which always fulfills the sufficient rank condition.

In order to identify if a control PN (\mathcal{N}^c) has the particular structure that ensures its liveness, two reductions rules (1 and 2) are applied sequentially. The resulting net is a *simplified control PN* denoted as \mathcal{N}^{cs} :

Rule 1. *The sequences $\{t_j^l \rightarrow p_{x_l} \rightarrow t_k^l\}$ are reduced to a unique transition t_{x_l} .*

Rule 2. *Once the rule 1 is applied, places p_{N_i} are implicit because $\mathbf{Pre}^c[p_{N_i}, T] = \mathbf{Post}^c[p_{N_i}, T] = \mathbf{m}[p_{N_i}]$ and they are removed.*

Because in \mathcal{N}^c a sequence $\{t_j^l \rightarrow p_{x_l} \rightarrow t_k^l\}$ represents a local T-semiflow of \mathcal{N}^d , in \mathcal{N}^{cs} transition t_{x_l} (rule 1) also represents a local T-semiflow of

\mathcal{N}^d . Moreover, applying rule 2, places p_{N_i} are removed, so all places in \mathcal{N}^{cs} represent buffers. In this way, \mathcal{N}^{cs} is composed of isolated subnets where the places represent buffers and the transitions represent local T-semiflows of \mathcal{N}^d . Moreover, each isolated subnet is formed by T-semiflows that represent global T-semiflow of \mathcal{N}^d which have common input/output buffers.

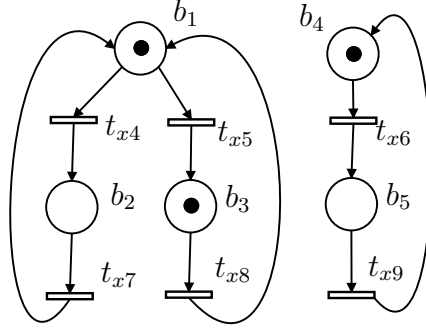


Figure 4.7: Simplified control PN after applying the reduction rules 1 and 2 to the control PN in Fig.4.5

Fig. 4.7 shows the simplified control PN obtained after applying rules 1 and 2 to the control PN in Fig.4.5. It can be seen that this net is composed by two isolated sub-nets. The one at left corresponding to the global T-semiflows \mathbf{x}_1 and \mathbf{x}_2 (given in Tab. 4.1) which have the common input/output buffer b_1 . The subnet at the right corresponds to the global T-semiflow \mathbf{x}_3

Assumption 4.5. *Let \mathcal{N}^d be a consistent and conservative DSSP structure, \mathcal{N}^c the Control PN obtained by applying Alg.3 and \mathcal{N}^{cs} the simplified Control PN obtained by applying the reduction rules 1 and 2.*

Lemma 4.6. *Let us assume nets \mathcal{N}^d , \mathcal{N}^c and \mathcal{N}^{cs} under conditions in assumption 4.5. Each isolated subnet of \mathcal{N}^{cs} is consistent and conservative.*

Proof. Alg.3 preserves in \mathcal{N}^c the number of agents, T-semiflows and buffers of \mathcal{N}^d . Moreover, Alg.3 also preserve in \mathcal{N}^c the consumption/production relation between buffers and local T-semiflows that exist in \mathcal{N}^d . So, if \mathcal{N}^d is consistent and conservative, then \mathcal{N}^c is consistent and conservative. On the other hand, the reduction rules 1 and 2 do not change the number of tokens consumed and produced from and to the buffers, so if \mathcal{N}^c is consistent and conservative, then each subnet in \mathcal{N}^{cs} is consistent and conservative. \square

Lemma 4.7. *Let us assume nets \mathcal{N}^d , \mathcal{N}^c and \mathcal{N}^{cs} under conditions in assumption 4.5. If \mathcal{N}^{cs} is live, then \mathcal{N}^c is live.*

Proof. This holds because the applied reduction rules (1 and 2) preserve liveness property [64]. \square

Lemma 4.8. *Let us assume nets \mathcal{N}^d , \mathcal{N}^c and \mathcal{N}^{cs} under conditions in assumption 4.5. If a subnet \mathcal{N}_i^{cs} of \mathcal{N}^{cs} does not simultaneously have choices and synchronizations is structurally live.*

Proof. If \mathcal{N}_i^{cs} does not simultaneously have choices and synchronizations it is an Equal Conflict (EQ) net. Moreover, \mathcal{N}_i^{cs} is consistent and conservative (Lemma 4.6). An EQ net, which is consistent and conservative, is structurally live iff $\text{rank}(\mathbf{C}) = |\text{SEQS}| - 1$ [65]. These particular EQ nets (without simultaneously choices and synchronizations) always fulfill the rank condition $\text{rank}(\mathbf{C}) = |\text{SEQS}| - 1$, so \mathcal{N}_i^{cs} is structurally live. \square

Lemma 4.9. *Let us assume nets \mathcal{N}^d , \mathcal{N}^c and \mathcal{N}^{cs} under conditions in assumption 4.5. If each one of the isolated subnets composing \mathcal{N}^{cs} does not simultaneously have choices and synchronizations the control PN \mathcal{N}^c is structurally live.*

Proof. By Lemma 4.7 and 4.8 \square

So, it is possible to ensure that a Control PN \mathcal{N}^c is structurally live by its Simplified Control PN \mathcal{N}^{cs} : if each subnet composing the \mathcal{N}^{cs} does not simultaneously have choices and synchronizations, \mathcal{N}^c is live (Lemma 4.9).

The subnets in the simplified control PN of Fig. 4.7 do not simultaneously have choices and synchronizations, so it is possible to ensure that its control PN in Fig. 4.5 is live.

Unfortunately, a subnet of a \mathcal{N}^{cs} could simultaneously have choices and synchronizations and consequently, the liveness of \mathcal{N}^c cannot be guaranteed. Next subsection proposes a methodology to force the liveness in these subnets.

4.3.2 Liveness enforcement of the Control PN

Let \mathcal{N}^d be a consistent and conservative DSSP structure, \mathcal{N}^c the Control PN obtained by applying Alg.3 and \mathcal{N}^{cs} the simplified Control PN obtained by

applying the reduction rules (1 and 2). If one of the subnets (\mathcal{N}_i^{cs}) of \mathcal{N}^{cs} has choices and synchronizations:

1. It can not be guaranteed that \mathcal{N}_i^{cs} is an EQ net.
2. Even assuming that \mathcal{N}_i^{cs} is an EQ net, it can not be guaranteed that the rank condition is fulfilled.

Therefore, the control PN (\mathcal{N}^c) could be not live.

In this subsection, we propose a methodology to force the liveness of the control PN (\mathcal{N}^c). This methodology is applied to those subnets (\mathcal{N}_i^{cs}) of the simplified control PN (\mathcal{N}^{cs}) where the liveness can not be guaranteed. Subsequently, the modifications included in each \mathcal{N}_i^{cs} are refined to \mathcal{N}^c .

Recall that each \mathcal{N}_i^{cs} of \mathcal{N}^{cs} is consistent and conservative (Lemma 12). So, each \mathcal{N}_i^{cs} is composed of one or more T-semiflows covering all transitions. The basic idea to force the liveness, is to ensure that the transitions are fired proportionally according to the T-semiflows. In this way, we prevent the freely chosen of a conflict that subsequently requires a synchronization.

The proposed methodology adds an input place for each transition in conflict. The number of tokens in these new places is equal to the number of times that its output transition can be fired. So, we propose that the initial marking of the new added places must be equal to the number of times that its output transition appear in each T-semiflows.

In order to identify when a T-semiflow has been fired completely, we define the check transition. When this transition is fired, means that the T-semiflow has finished and all its transitions can be fired again. So, the firing of the check transition should update the marking of the new added places.

Before giving formally the methodology to adds this new control places that force the liveness, we show a simple intuitive example.

Example 4.10. *The left part of Fig. 4.8 shows the simplified control PN (\mathcal{N}^{cs}) of the DSSP net (\mathcal{N}^d) in Fig.4.2(a). This net is not live because there exists a conflict (t_{x2}, t_{x3}) and subsequently a synchronization in transition t_{x4} (**Pb 2**).*

In the right part of Fig.4.8 the new control buffers $p_{t_{x2}}$ and $p_{t_{x3}}$ are added for controlling the firing of t_{x2} and t_{x3} . These new buffers constraint to one the times that t_{x2} and t_{x3} can be fired before one firing of t_{x4} .

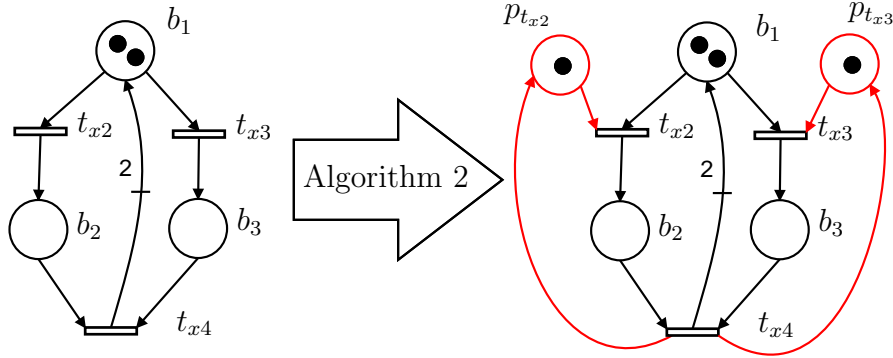


Figure 4.8: Left: simplified control PN of the DSSP net in Fig. 4.2(a); right: simplified control PN with new added buffers

To know when a global T-semiflow of a subnet composing the simplified Control PN has been fired completely, we define its check transition (t_{x4} in the net in Fig. 4.8). The check transition must belong to a unique T-semiflow and its firing must produce enough tokens to fire completely the T-semiflow again. The formal definition is given in Def.4.11.

Definition 4.11. Let \mathcal{N}_i^{cs} be a subnet of a simplified control PN \mathcal{N}^{cs} , let t_{ch}^j be the check transition of a T-semiflow $\mathbf{x}_j \in \mathcal{N}_i^{cs}$ and let $\mathbf{m} = \mathbf{Post}^{cs}[P, t_{ch}^j]$ be the marking produced by firing this check transition (t_{ch}^j). Then the next conditions must be true:

1. $\mathbf{x}_k(t_{ch}^j) = \begin{cases} 1, & \text{if } k = j \\ 0, & \text{if } k \neq j \end{cases}$
2. From \mathbf{m} must be possible to fire the T-semiflow \mathbf{x}_j and come back to the marking \mathbf{m} ($\mathbf{m} \xrightarrow{\mathbf{x}_j} \mathbf{m}$).

If there are more than one transition that fulfill the constraints in Def. 4.11, we should choose a non-conflict transition.

Alg. 4 adds the control places that force the liveness of a subnet \mathcal{N}_i^{cs} . These control places force the liveness by ensuring the proportional firing of the transition belonging to a T-semiflow. Each new added place only have one output transition, so the resulting net preserves the distributed property. It is necessary that all T-semiflows of the sub-nets to which Alg.4 is applied to have

Algorithm 4: Adds control places in a non live subnet \mathcal{N}_i^{cs} (containing both choices and synchronizations) of the simplified control PN (\mathcal{N}^c) and gives the initial marking m_0

Require: A subnet \mathcal{N}_i^{cs} of the simplified control PN \mathcal{N}^{cs}

$$(\mathcal{N}_i^{cs} = \langle P^{cs}, T^{cs}, \mathbf{Pre}^{cs}, \mathbf{Post}^{cs} \rangle)$$

Ensure: A subnet of the simplified control PN with control places

$$(\mathcal{N}^+ = \langle P^+, T^+, \mathbf{Pre}^+, \mathbf{Post}^+ \rangle) \text{ and initial live marking } \mathbf{m}_0.$$

- 1: Compute the set X of minimal T-semiflow \mathbf{x}_i in \mathcal{N}_i^{cs}
 - 2: **for all** T-semiflow \mathbf{x}_i of X **do**
 - 3: Compute the check transition t_{ch}^i
 - 4: **end for**
 - 5: Let T_{ck} be the set of check transitions
 - 6: Let T_{cn} be the set of conflict transitions
 - 7: **for all** $t_i \in (T_{cn} \setminus T_{ck})$ **do**
 - 8: Add a place p_{t_i} s.t. $\mathbf{Pre}^+[p_{t_i}, t_i] = 1$
 - 9: **end for**
 - 10: **for all** T-semiflow \mathbf{x}_i of X **do**
 - 11: **for all** t_j s.t. $\mathbf{x}_i[t_j] > 0$ and $t_j \in (T_{cn} \setminus T_{ck})$ **do**
 - 12: $\mathbf{Post}^+[p_{t_j}, t_{ch}^i] = \mathbf{x}_i[t_j]$
 - 13: **end for**
 - 14: **end for**
 - 15: $\mathbf{m}_0 = \sum_{\mathbf{x}_i \in X} \mathbf{Post}^+[P^+, t_{ch}^i]$
-

a check transition according to Def. 4.11. Alg.4 adds a control place p_{t_j} for each conflict transition t_j that is not a check transition t_{ch}^i . The firing of transition t_j is constrained by the new added place p_{t_j} : $\mathbf{Pre}^+[p_{t_j}, t_j] = 1$ (Steps 7-9). When the check transition t_{ch}^i of T-semiflow \mathbf{x}_i is fired, the marking of the places p_{t_j} that control transition t_j belonging to \mathbf{x}_i are updated: $\mathbf{Post}^+[p_{t_j}, t_{ch}^i] = \mathbf{x}_i[t_j]$ (Steps 10-14). The initial marking \mathbf{m}_0 is given by the marking that would be produced by firing each check transition t_{ch}^i (Step 14).

In Fig.4.9 a possible subnet \mathcal{N}_i^{cs} (containing both choices and synchronizations) of a simplified control PN \mathcal{N}^{cs} is shown in black. This net is composed by two global T-semiflows $\mathbf{x}_1 = 2t_1 + 2t_2 + t_3 + t_4 + t_5 + t_6 + t_8$ and $\mathbf{x}_2 = 2t_2 + t_6 + t_7 + t_9$. Moreover, the control places p_{t_j} added by applying Alg.4 are represented in red color. \mathcal{N}_i^{cs} without the control places is not live because

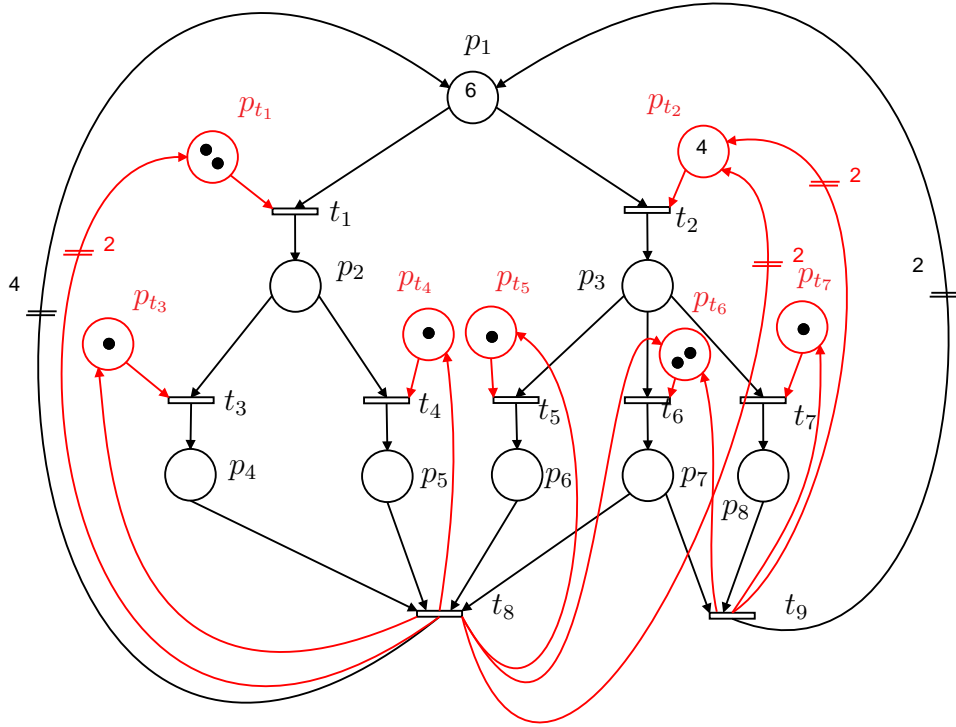


Figure 4.9: A non live subnet of a simplified Control PN plus control places (red)

there are choices between $\{t_1, t_2\}$, $\{t_3, t_4\}$ and $\{t_5, t_6, t_7\}$ and subsequently synchronizations in t_8 and t_9 are required (**Pb.** 2 in Sec.4.1.2). However, after adding the control places, some places become implicit. Particularly, adding p_{t_1} and p_{t_2} with $\mathbf{m}_0[p_{t_1}] = 2$ and $\mathbf{m}_0[p_{t_2}] = 4$ the transition t_1 can be initially fired two times and transition t_2 four times becoming the place p_1 implicit. Removing p_1 (implicit) it is possible to combine p_{t_1} with p_2 and p_{t_2} with p_3 . Now, the control places $p_{t_3} - p_{t_7}$ make the previous combined places implicit. Repeating this process iteratively, it is possible to check that the simplified control PN with the added places is live.

Lemma 4.12. *The net (\mathcal{N}^+) resulting from applying Alg.4 to a non live subnet \mathcal{N}_i^{cs} is live.*

Proof. \mathcal{N}_i^{cs} is a consistent and conservative PN (Lemma 4.6) where each transition represents a local T-semiflow of \mathcal{N}^d and each place represent a buffer.

The initial marking \mathbf{m}_0 of the net is equal to the marking produced by firing of check transitions.

By applying Alg.4, a control place p_{t_j} is added for each conflict transition t_j that is not a check transition. These control places limit the firing of conflict transitions making the original input places of these conflicts to be implicit. This happens because the firing of check transitions produce,

- in its original output places (buffers), enough tokens to fire completely the global T-semiflows.
- in the new added places p_{t_j} , the exact number of tokens for firing the conflict transitions t_j the times that appears in the global T-semiflows.

If a place that receives tokens by firing of check transitions is not a decision (i.e., has only one output transition), it can be fused with the output places of its unique output transition. This procedure can be iterated until the obtained place is a decision. Moreover, the marking of this place is greater or equal to the marking of the added places p_{t_j} (i.e., constraining the firing of the transitions in the conflict). In this way, this place is implicit and can be removed without compromise the liveness of the net. After removing of an implicit place, its output transitions t_j are not anymore in conflict and each one has only one input place p_{t_j} . Starting now with an input place p_{t_j} the full procedure can be iterated and at the end it will be removed being implicit for the next conflict. Finally, the net structure obtained is composed by the check transitions connected with one place by a self-loop. So, the resulting net is live and consequently the original net with the added places (\mathcal{N}^+) is live. \square

Alg.4 adds some control places that force the liveness of the simplified Control PN (\mathcal{N}^{cs}). However, the control PN (\mathcal{N}^c) is the net used on the control level. So, the new added control places in \mathcal{N}^{cs} should be translated to \mathcal{N}^c .

Remark 4.13. *Considering that the sequence $\{t_j^l \rightarrow p_{x_i} \rightarrow t_k^l\}$ in \mathcal{N}^c has been reduced to a unique transition t_{x_i} in \mathcal{N}^{cs} , for each new control buffer (b_s) added in \mathcal{N}^{cs} an homologous buffer (b_h) is added in \mathcal{N}^c as follows:*

- $\mathbf{Pre}^c[b_h, t_j^l] = \mathbf{Pre}^{cs}[b_s, t_{x_i}]$

- $\mathbf{Post}^c[b_h, t_k^l] = \mathbf{Post}^{cs}[b_s, t_{x_l}]$
- $\mathbf{m}_0[b_h] = \mathbf{m}_0[b_s]$

Absence of check transition. The methodology proposed in this subsection uses Alg. 4 to ensure the liveness of the subnets (\mathcal{N}_i^{cs}) of the simplified control PN (\mathcal{N}^{cs}). Alg. 4 assumes that for each T-semiflow in \mathcal{N}_i^{cs} is possible to compute its check transition according to Def. 4.11. Unfortunately, this is not always true, and could exist no check transition for some T-semiflows in \mathcal{N}_i^{cs} . In these situations should be necessary to fix a *set of check transitions* that together fulfill the conditions in Def. 4.11. However, an automatic procedure to find the set of check transitions is difficult to fix. One possible brute force approach for finding the set of check transitions could be to analyze all possible combinations of the transitions that only belong to the analyzed T-semiflow. Once the set of check transitions is fixed, a *new virtual check transition* can be added. This new transition will be the check transition and must synchronize the transitions composing the set of check transitions. Fig. 4.10 shows a possible set of check transition ($\{t_1, t_2, t_3\}$) and the new virtual check transition t_{ch} .

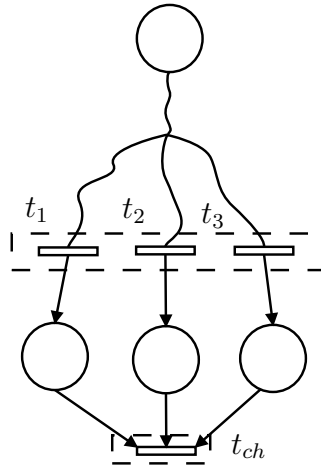


Figure 4.10: Transformation of the set of check transition $\{t_1, t_2, t_3\}$ into a virtual check transition t_{ch}

4.4 Control policy and system evolution

In this section, we propose a control policy that ensures liveness in a DSSP net (\mathcal{N}^d) through its control PN (\mathcal{N}^c). The control policy is initially based on two conditions:

Condition 1. *The firing of a local T-semiflow can only start if all its input buffers have enough tokens to complete the firing of all its transitions.*

Condition 2. *When the firing of a local T-semiflow \mathbf{x}_i^i starts in the agent \mathcal{N}_i , it is not possible to fire transitions that are not support of \mathbf{x}_i^i until all transitions of $\|\mathbf{x}_i^i\|$ are fired.*

To achieve that, \mathcal{N}^c constraint the fireability of the transitions in \mathcal{N}^d (see Fig. 4.11) through guard expressions. Each transition in \mathcal{N}^d has associated a guard expression. These guard expressions are logical conditions related with the state (marking) of \mathcal{N}^c . A transition $t \in T^d$ can be fired only if the associated guard expression is true. Of course, to fire t , it is also necessary to be enabled in \mathcal{N}^d .

Labelling in Control PN: Each transition in \mathcal{N}^c is labelled with the name of a transition of \mathcal{N}^d . In the process of obtaining \mathcal{N}^c , for each local T-semiflow \mathbf{x}_l in \mathcal{N}^d we add a ordinary subnet ($t_j^l \rightarrow p_{x_l} \rightarrow t_k^l$) in \mathcal{N}^c . The transition t_j^l/t_k^l is labelled with the name of the first/last transition of \mathbf{x}_l . In addition, the place p_{x_l} is labelled with the name of the local T-semiflow \mathbf{x}_l . In \mathcal{N}^c , these labels appear in red color. For example, in Fig. 4.5 the ordinary subnet $t_5^7 \rightarrow p_{x_7} \rightarrow t_3^7$ is added in \mathcal{N}^c due to the existence of the local T-semiflow \mathbf{x}_7 in \mathcal{N}^d . Since t_5/t_3 is the first/last transition of \mathbf{x}_7 , then the transition t_5^7/t_3^7 is labelled with t_5/t_3 . In addition since the local T-semiflow is \mathbf{x}_7 , the place p_{x_7} is labelled with x_7

Control policy: A transition t belonging to \mathcal{N}^d can be fired if it is enabled ($\mathbf{m} \geq \mathbf{Pre}^d[P, t]$) and its guard expression is true. The guard expression associated with t becomes true if at least one of the next conditions is fulfilled in \mathcal{N}^c :

- there is a transition labelled as t and it is enabled.
- the marking of a place labelled with the name of a local T-semiflow to which t belongs is equal to one.

First condition prevents to start the firing of a local T-semiflow whose input buffers do not have enough tokens. While second condition ensures that if a local T-semiflow has started, all the guard expressions of its transitions become true.

Evolution of the control PN: The evolution of \mathcal{N}^c is synchronized with the evolution of \mathcal{N}^d . The events in \mathcal{N}^d (firing of transitions) are considered as input signals in \mathcal{N}^c . When a transition t_o is fired in \mathcal{N}^d , then in \mathcal{N}^c :

- if there exists no transition labelled as t_o , no transition is fired.
- if there exist enabled transitions labelled as t_o , one of them must be fired. The transition that will fire corresponds to the global T-semiflow that will start to fire. Different policy could be defined, in this work we fire randomly one transition.

Fig. 4.11 shows the control flow diagram of the proposed liveness enforcement approach for a DSSP structure (\mathcal{N}^d) using a control PN (\mathcal{N}^c). The control policy depends on the markings of \mathcal{N}^d and \mathcal{N}^c . The marking of \mathcal{N}^c inhibits some enabled transitions to be fired in \mathcal{N}^d if a deadlock could appear. A transition t is chosen from the set of control-enabled transitions.

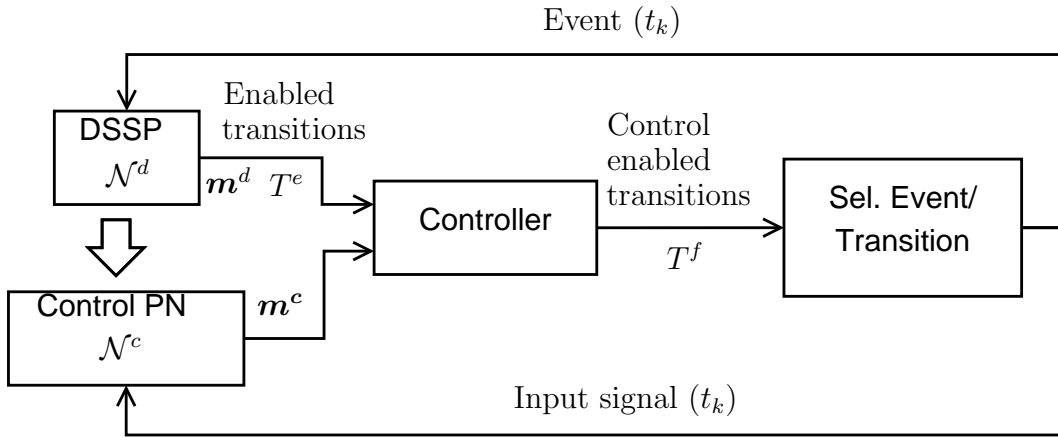


Figure 4.11: Control diagram of a DSSP structure using a control PN

Alg. 5 implements the control policy and the system evolution on a DSSP net and its control PN.

Algorithm 5: Control policy and systems evolution on a DSSP net \mathcal{N}^d and its control PN \mathcal{N}^c

Require: A DSSP net $\mathcal{N}^d = \langle P^d, T^d, \mathbf{Pre}^d, \mathbf{Post}^d \rangle$, its live control PN $\mathcal{N}^c = \langle P^c, T^c, \mathbf{Pre}^c, \mathbf{Post}^c \rangle$ and the initial markings \mathbf{m}_0^d and \mathbf{m}_0^c of both nets.

Ensure: A live evolution of \mathcal{N}^d through \mathcal{N}^c

```

1:  $\mathbf{m}^d := \mathbf{m}_0^d$  {initializing the current state in  $\mathcal{N}^d$ }
2:  $\mathbf{m}^c := \mathbf{m}_0^c$  {initializing the current state in  $\mathcal{N}^c$ }
3:  $T^e := \emptyset$  {initializing the set of enable transition in  $\mathcal{N}^d$  at marking  $\mathbf{m}^d$ }
4:  $T^f := \emptyset$  {initializing the set of transition that can be fired in  $\mathcal{N}^d$ }
5: for all  $t_i \in T^d$  do
6:   if  $\mathbf{Pre}^d[P^d, t_i] \leq \mathbf{m}^d$  then
7:      $T^e := T^e \cup t_i$ 
8:   end if
9: end for
10: for all  $t_j \in T^e$  do
11:   if the guard of  $t_j$  is true at marking  $\mathbf{m}^c$  then
12:      $T^f := T^f \cup t_j$ 
13:   end if
14: end for
15: A transition  $t_k \in T^f$  is fired in  $\mathcal{N}^d$ 
16:  $\mathbf{m}^d := \mathbf{m}^d + \mathbf{C}^d[P^d, t_k]$  { $\mathbf{m}^d$  is updated}
17: if There exist a transition labelled as  $t_k$  in  $\mathcal{N}^c$  then
18:   transition labelled as  $t_k$  is fired in  $\mathcal{N}^c$ 
19:    $\mathbf{m}^c := \mathbf{m}^c + \mathbf{C}^c[P^c, t_k]$  { $\mathbf{m}^c$  is updated}
20: end if
21: go to Step 3

```

Example 4.14. Let \mathcal{N}^d be the DSSP net in Fig. 4.1(a) and let \mathcal{N}^c be its control PN given in Fig. 4.5. Considering the previously explained control policy, the system evolution of \mathcal{N}^d and \mathcal{N}^c is going to be analyzed. To make reading easier, Fig. 4.12 shows \mathcal{N}^d and \mathcal{N}^c together. In \mathcal{N}^d , transitions t_1 , t_9 , t_5 and t_{12} are enabled, but only t_1 , t_9 and t_5 can be fired because in \mathcal{N}^c transition labelled as t_{12} (t_{12}^9) is not enabled. In this way, only the local T -semiflows whose input buffers have enough tokens for completing their firing are allowed to start. Let us assume that t_5 is fired in \mathcal{N}^d :

- automatically the enabled transition t_5^8 (labelled as t_5) has to be fired in \mathcal{N}^c .
- Now in \mathcal{N}^d , the transitions t_1 , t_9 , t_4 and t_6 are enabled.
- In \mathcal{N}^c transition labelled as t_1 (t_1^4 and t_1^5) and t_9 (t_9^6) are enabled, so t_1 and t_9 can be fired in \mathcal{N}^d . Moreover, in \mathcal{N}^c the marking of the place p_{x_8} labelled as x_8 is equal to 1, so transition $t_6 \in ||x_8||$ can be fired in \mathcal{N}^d . However, in \mathcal{N}^c the marking of the place p_{x_7} labelled as x_7 is equal to 0, so transition $t_4 \in ||x_7||$ cannot be fired. Let us assume that t_6 is fired in \mathcal{N}^d .
- The firing of t_6 in \mathcal{N}^d does not imply changes in \mathcal{N}^c because there no exist a transition labelled as t_6 in this net.

The computational complexity of the proposed live enforcement approach is exponential because it is necessary to compute the set of minimal T-semiflows. The other steps necessary to obtain the control PN and to force its liveness can be applied in a polynomial time.

Lemma 4.15. *Given a non-live consistent and conservative DSSP structure \mathcal{N}^d in which there exist waiting places in all agents \mathcal{N}_i^d , by applying Alg. 3 a control PN \mathcal{N}^c is obtained. Ensuring or forcing the liveness of the control PN (Alg. 4) and following the control policy on \mathcal{N}^d and \mathcal{N}^c described in Alg. 5, the original DSSP structure is deadlock free.*

Proof. In \mathcal{N}^c obtained by applying Alg. 3 to \mathcal{N}^d , each local T-semiflow \mathbf{x}_l^i is represented by a ordinary subnet $t_l^i \rightarrow p_{x_i} \rightarrow t_k^i$. In addition the buffers and waiting places are represented in \mathcal{N}^c . All places that represent the input buffers of each local T-semiflow \mathbf{x}_l^i are assigned to t_l^i in \mathcal{N}^c .

The control policy using guard expressions prevents the firing of a transition in \mathcal{N}^d if (i) the corresponding labelled transition is not enabled in \mathcal{N}^c or (ii) the marking of the place in \mathcal{N}^c labelled with the same name as the local T-semiflow (to which belongs) is empty.

First, condition (i) of the guard ensures that only the T-semiflows whose input buffers have enough tokens to finish could fire its first transition t_{first} .

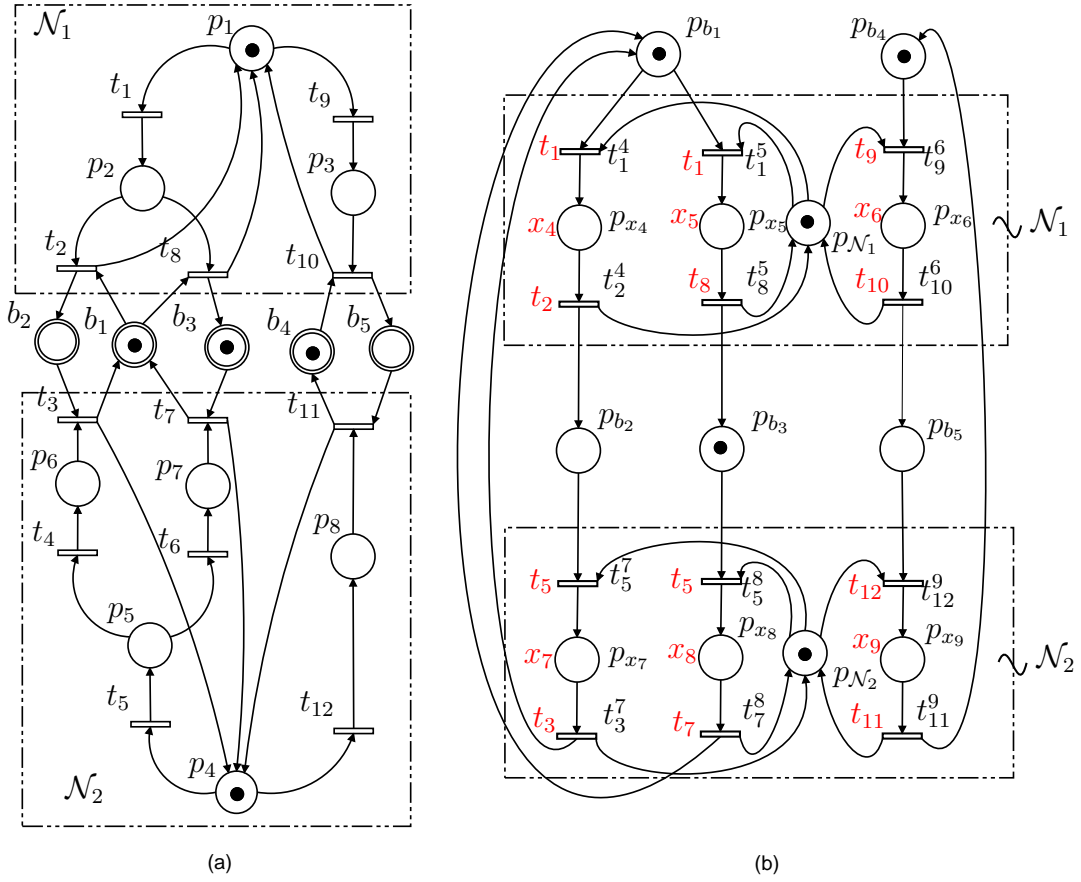


Figure 4.12: Example of the control policy: (a) DSSP net (\mathcal{N}^d) given also in Fig 4.1; (b) its control PN (\mathcal{N}^c) given also in Fig. 4.5

Moreover, once the first transition of a local T-semiflow \mathbf{x}_i^i has been fired in \mathcal{N}_i^d , a transition labelled with the same name is also fired in \mathcal{N}^c , and consequently a token is generated in the place labelled with the name of the local T-semiflow. This token ensures that the second condition (ii) in the guard expression of $t \in \mathbf{x}_i^i$ is true. In this way, only $t \in \mathbf{x}_i^i$ can be fired in \mathcal{N}_i^d . The other transitions in the agent \mathcal{N}_i^d not belonging to the T-semiflow are disabled. Once all $t \in \mathbf{x}_i^i$ are fired, \mathcal{N}_i^d returns to the initial marking by the firing of the output transition.

The firing of an output transition in \mathcal{N}^d , implies the firing of a transition labelled with the same name in \mathcal{N}^c . The firing in \mathcal{N}^c increases the marking

of the places that represents the output buffers of this local T-semiflow. This new marking of the places that represent the buffers enables the firing of other local T-semiflow that composing global one.

In this way for each global T-semiflow, the control PN enables the firing of the local ones that have enough tokens in its input buffers. Moreover when the firing of a local T-semiflow starts, the agent to which belongs has to fire all transitions of the local T-semiflow and returns to the waiting place.

As the net is consistent and conservative, there exist sequences of firing of the local T-semiflow belonging to each global one. Using the live control PN, we are forcing the original DSSP net to follow a correct order and prevent in this way the deadlock of the agent. \square

4.5 Improving permissibility and working with a unique PN

In this section, we propose an easy and intuitive transformation in the control PN for improving the permissibility of the proposed liveness enforcement approach. Moreover, in order to be able to work with a unique PN, we provide the necessary steps to obtain it.

4.5.1 Improving the permissibility

The proposed approach for liveness enforcement is strongly restrictive due to conditions 1 and 2 stated at the beginning of Sec. 4.4.

In order to reduce the restrictiveness of the approach, we partially relax condition 1. This relaxation will allow a local T-semiflow to start firing if is expected that, in a future marking, its input buffers will have enough tokens to complete it. The relaxation is achieved by a transformation in the control PN: the production of tokens from the sequences $t_a \rightarrow p \rightarrow t_b$ to its output buffers is advanced from transition t_b to transition t_a . In this way, from the moment that a sequence $t_a \rightarrow p \rightarrow t_b$ fires the first transition t_a , the tokens are produced in the output buffers. These tokens allow that other T-semiflows can start its firing despite the fact that in the DSSP net the input buffers do not have enough tokens yet.

The advance of the token production in the control PN do not compromise the liveness of the DSSP due to condition 2 stated before is ensuring that if a local T-semiflow starts its firing, it must be fired completely. So, finally, the token will be produced in the original DSSP net.

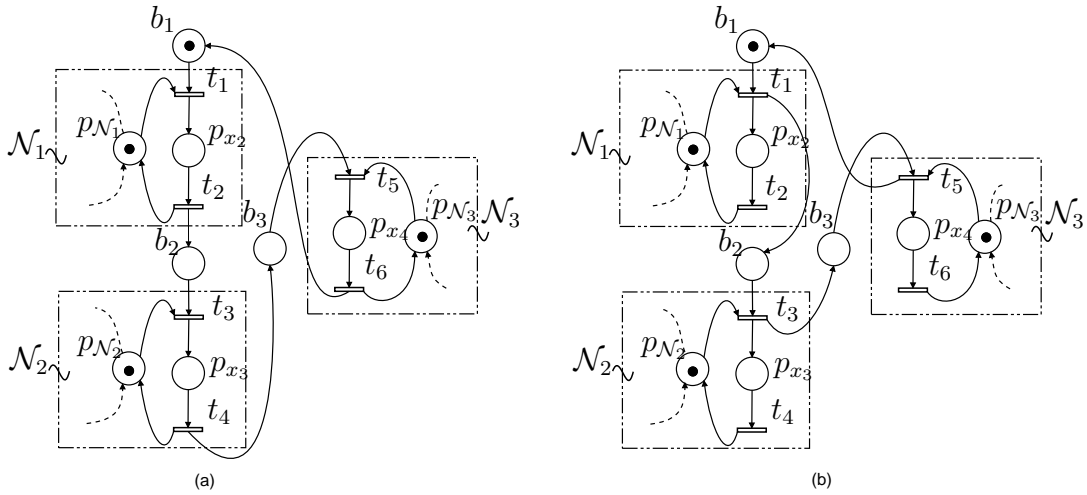


Figure 4.13: A Control PN where: a) the production of tokens is generated from the last transition of the sequences; b) the production of tokens is generated from the first transition of the sequences

In Fig.4.13(a) we show a control PN where the production of token to buffer is generated from the last transition (t_b) of the sequences ($t_a \rightarrow p \rightarrow t_b$) while in Fig.4.13(b) this production has been advanced to the first transitions (t_a).

4.5.2 Composition of DSSP system and control PN system

Our approach is based on obtaining a control PN \mathcal{N}^c that through guard expressions limit the firing of transitions in the DSSP net \mathcal{N}^d . In this way, \mathcal{N}^c can be seen as a high level scheduler that guides the firing of transitions in \mathcal{N}^d . Having a control PN is a great advantage in the case of complex systems, since \mathcal{N}^c gives an overview of what is happening in the system. However, working with smaller systems could be interesting to have a unique net including \mathcal{N}^d and \mathcal{N}^c . The idea is to impose the same initial conditions (1 and 2) that have been imposed through the control policy. To achieve this, some modifications

are made in \mathcal{N}^d . First, in order that each local T-semiflow has its own “first” transition, some transitions are duplicated. Then the input buffers of each local T-semiflow are pre-assinged to its first transitions. Finally, new places are included in order to “guide” the complete firing of a local T-semiflow when its first transition has been fired. In the following the steps to obtain this unique net system are given.

1. *Duplicate first transitions.* Each first transition belonging to more than one local T-semiflow must be duplicated as many times as the number of local T-semiflows to which it belongs.
2. *Pre-assignment of the buffers.* The input buffers of each local T-semiflow must be pre-assinged to its first transition.
3. *Include new places.* For each conflict transition t_i (which is not a first transition), a place p_{t_i} is added such that p_{t_i} limits the firing of t_i ($\mathbf{Pre}[p_{t_i}, t_i] = 1$). Moreover, from each first transition (t_j^k) of the local T-semiflows to which t_i belongs, there is an arc to p_{t_i} ($\mathbf{Post}[p_{t_i}, t_j^k] = 1$).
4. *Obtain the live control PN and ensure or force its liveness.* It is necessary to know if control buffers are needed to force the liveness in the control PN.
5. *Include the control buffers.* If control buffers have been introduced to force the liveness of the control PN, these buffers must be added.

Fig. 4.14 shows the equivalent controlled system \mathcal{N}^e obtained from applying Step 1-5 to the non live DSSP system \mathcal{N}^d in Fig. 4.1(a). The modification performed are showed in different colors. In step 1 (blue) t_1 and t_5 have been transformed in two pairs of transition: $\{t_1^4, t_1^5\}$ and $\{t_5^7, t_5^8\}$. In step 2 (green), the input buffers of the local T-semiflows has been preassigned to its first transitions. For example, b_2 is an input buffer of \mathbf{x}_7 pre-assinged to its first transition t_5^7 . Finally, in step 3 (red) for each transition in conflict relation that is not a first transition ($\{t_2, t_3\}$ and $\{t_4, t_6\}$), a new place ($p_{t_2}, p_{t_3}, p_{t_4}$ and p_{t_6}) has been added. Step 5 is not applied because in Step 4 we obtain a live control PN (Fig. 4.5) without adding control buffers.

In the equivalent controlled net \mathcal{N}^e of Fig. 4.14, a local T-semiflow only can start if its input buffers have enough tokens to fire all its transitions. Moreover,

once a local T-semiflow has started, all its transition are fired. For example, \mathbf{x}_8 can only start when its input buffer b_3 has a token because b_3 has been pre-assigned to transition t_5^8 . Once \mathbf{x}_8 has started its firing by transition t_5^8 , a token is produced in p_{t_6} . This token will enable the conflict transition t_6 and will guide the agent to fire the local T-semiflow \mathbf{x}_8 completely.

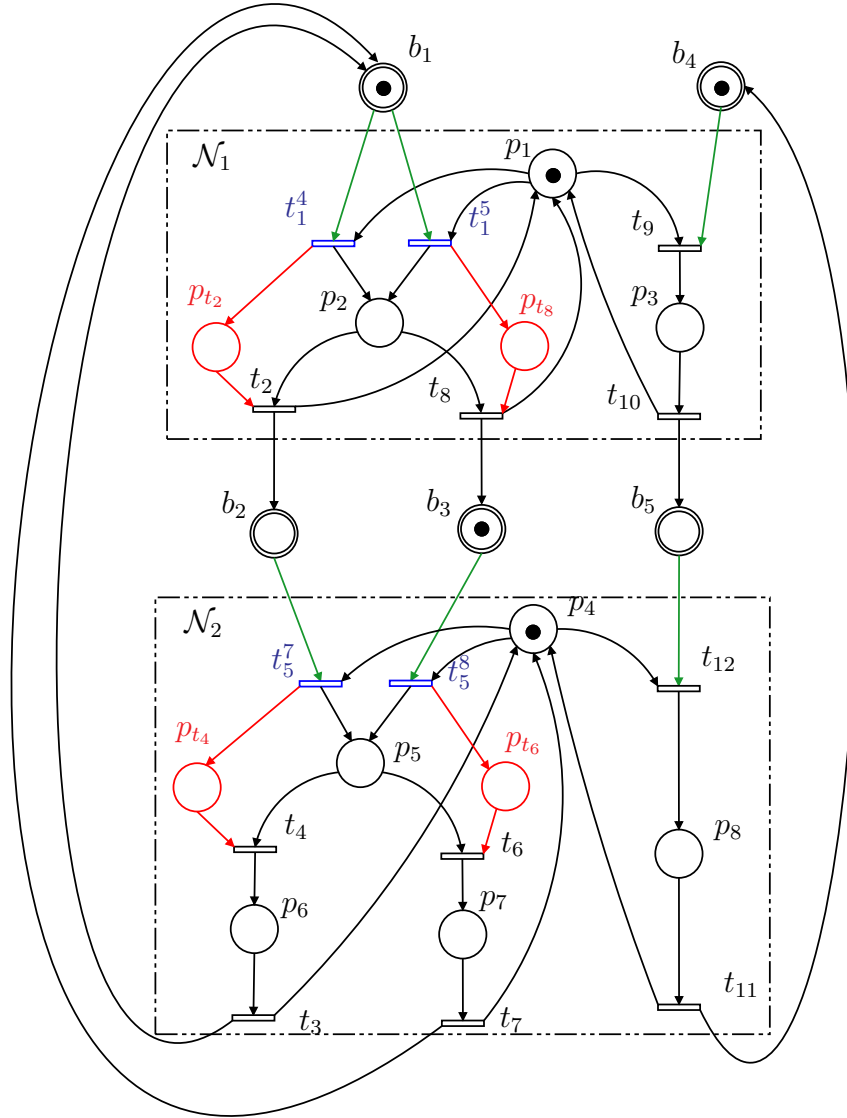


Figure 4.14: Equivalent controlled net (\mathcal{N}^e) of the non live DSSP system in Fig. 4.1(a)

4.6 Discussion

In this chapter we develop a liveness enforcement strategy for DSSP system initially based on two main ideas: i) a local T-semiflow can start firing only if its input buffers have enough tokens and ii) when a local T-semiflow starts firing, then all its transitions must be fired. The permissibility of the method is improved by a partial relaxation in the first condition stated before: a local T-semiflow can start firing if it is expected that its input buffers will have enough tokens in a future marking. Initially, the proposed methodology is formalized on two levels: *execution* and *control*. In the execution level the original DSSP system evolves conditioned by the control level: each transition in DSSP net has associated a guard expression which depends on the state of the control level. In the control level we use a net system obtained from the DSSP structure called the control PN system. This net has a predefined type of structure and models the consumption/production relation between buffers and local T-semiflows of the DSSP net. The disjoint local T-semiflows in the control PN allows that its firing will only be conditioned by buffers that in the original DSSP structure were its input buffers. Both net systems, DSSP and control PN evolve synchronously following a given control/evolution policy. In this chapter we provide an algorithm to obtain the *control PN* and a methodology to ensure or force its liveness. For general DSSP structures in which all agents have a waiting place, the proposed method ensures the structural liveness of the control system. Moreover, from the execution (\mathcal{N}^d) and the control (\mathcal{N}^c) net-system a unique transformed net system can be obtained. In this way, we give the user the possibility to work with a unique controlled net system obtained by a set of compositions rules. The main advantage of working with a single system with respect to the method based on two levels (execution and control) is the reduction of the total number of places and transitions. However, the method based on two levels allows to improve the permissibility of the method by relaxing condition (i) and, on the other hand to check the global state of the DSSP system only with an overview at the control system. This means that only looking at the control system we can check the actual T-semiflows that are executing. This is because the control system is a high-level scheduler of the DSSP system.

Part II

Planning and scheduling of patients in a surgical service

Chapter 5

Elective surgery scheduling under OR block booking

Summary

This chapter considers the scheduling of elective patients in a surgery department of a hospital. We assume an ordered list of patients that should be scheduled for surgery in the available *Operation Rooms* (ORs), each one being possible to be used for a specific duration (time block). Based on the average duration of surgeries, three mathematical programming problems are discussed: 1) Quadratic Assignment Problem (QAP); 2) a Mixed Integer Linear Programming (MILP) model; and 3) Generalized Assignment Problem (GAP). These problems take into account two contradictory objectives: obtain a given occupation rate of the OR and respect as much as possible the preference order of the patients in the waiting list. Then, assuming that surgery duration and cleaning time follow a normal distribution a *New Mixed Integer Quadratic Constrained Programming* (N-MIQCP) model is proposed. This model maximizes the occupation rate of the ORs at the same time that imposes a confidence level of not exceeding the available time. Of course, this model also respects the order of the patients. Different heuristic methods have been used to solve large instances. The results are previously published on [21, 17, 15, 16].

5.1 Background

The *Operation Room* (OR) is one of the most expensive material resources of the hospitals and approximately 60% of patients need it at some point during their hospital stay [1]. So, ORs capacity is a crucial but limited hospital resource. In this way, good management of ORs improves the efficiency of the hospital. The large waiting lists of patients and the uncertain duration of their surgeries make the scheduling task hard and costly for the medical staff.

5.1.1 Introduction

In this chapter, we use mathematical programming to model the scheduling of non-urgent surgeries in a hospital department. Linear programming models were developed during World War II to make plans or proposals of time for training, logistics or deployment of combat units. After the war, many industries began to use it in their daily planning. Subsequently, it was observed that, through proper system modeling, linear programming can be applied to different fields. The scheduling patients problem can be seen as the planning of a production system: (a) there is a waiting list of patients representing the system demand and (b) there exists a limited number of surgeons and a limited number of ORs representing the capacity of the production system. For our application, the bottleneck of the resources are the ORs. In particular, there exists a limited number of ORs for non-emergency surgeries, each one being available only a certain number of hours per day.

Furthermore, surgical costs typically account for approximately 40% of the hospital resource costs [47], while surgeries generate around 67% of hospital revenues [38]. Additionally, because of the aging population, the demand for surgical services is increasing. From this view, Spanish hospitals are an attractive case study having one of the oldest population in Europe: the median age is 42.7 years and 17.95% of the Spanish population is older than 65 years. Moreover, the average life expectancy is 81.8 years being the female life expectancy (84.9 years) one of the highest in Europe [14]. Since 2008, and as a result of the economic crisis, the number of surgical interventions performed annually in Spanish public hospitals has stagnated at 3.5 million. This situation has also contributed to a continuous increase in the number of patients in the surgical waiting lists, reaching 180.000 in 2017. In some cases, these huge

lists lead in waiting times greater than 6-months, which is becoming a serious problem for Spanish health-care system.

Currently is not possible to increase the surgical resources and for this reason, the importance of improving the efficiency of the healthcare system is accentuated. New management techniques, scheduling methods, and specific information systems could help to improve the efficiency of the surgical services and, consequently, could reduce the surgical waiting lists.

In literature, many researchers have studied the problem of planning and scheduling of elective patients. For a state of the art, we recommend the reader to the survey [10] and the references herein. The scheduling and planning of resources have been studied for other problems, as for example home care services [45, 44]. Petri net models have been used for modeling and management of healthcare systems, see for example [3, 27, 5, 48].

Researchers frequently differentiate between *strategic* (long-term), *tactical* (medium term) and *operational* (short term) approaches to situate their planning or scheduling patients problems. Three classical levels are considered in bibliography depending on the time horizon [2]:

1. *case mix planning* is a long-term strategic planning. In this level, considering the hospital's mission, the resource capacity planning is studied.
2. *master surgery schedule* is a medium-term tactical approach that divides the OR time into different blocks (time blocks) which are subsequently assigned to different surgeon groups on each weekday.
3. *scheduling of patients* is a short-term operational approach in which the patients who should be operated in the next time blocks are assigned.

In the studied hospitals, each group of surgeons has its own waiting list of patients. Moreover, time blocks of the ORs are previously booked to each surgeon group. So, the problem to solve is the scheduling of patients from a waiting list to the time blocks (level 3 stated before). This assignment should optimize the use of the ORs, at the same time that respects as much as possible the order of the patients in the waiting list. In this chapter, two different approaches for optimizing the use of the ORs are given:

The first one (Sec. 5.2) considers the average duration of the surgeries and it is based on obtaining a given occupation rate in each time block. The

“appropriate” occupation rate of the time blocks must be established by doctors. A lack as an excess in the occupation rate of OR respect the objective is a system default. Lacks imply the consumption of human resources without their utilization, and excess means that staff could lengthen their working day. However, there are uncertainties due to uncontrollable factors such as unforeseen events or the different nature of each body. Moreover, inexperienced doctors could find difficult to impose an “appropriate” occupation rate. For this reason, a second approach that considers uncertain parameters following a normal distribution is given (Sec. 5.3). In this case, doctors impose a minimum confidence level of not exceeding the available time (which is much more intuitive) and the occupation rate of the time blocks is maximized.

5.1.2 Scheduling Criteria

In this chapter, it is assumed that hospitals work under surgical block booking, i.e., each surgical team has some surgical blocks booked for performing the surgeries of their patients. So, the problem of scheduling of patients can be divided into different independent sub-problems: one for each medical team. In this process, some patients are assigned from the waiting list to the next time blocks previously booked. Currently, the scheduling is performed manually, so the schedulers (normally medical doctors) are guided by their own intuition and experience to assign the patients. This manual scheduling has three main problems:

1. Schedulers need to spend time in administrative tasks (scheduling).
2. Usually, under or overutilization of the ORs is obtained.
3. Because is made by humans, objectivity can be questioned.

The main goal is to develop a method for helping team schedulers to perform automatically the scheduling of their patients. In this way the schedulers would need few time to carry out the scheduling of patients, so the problem (1) of the manual scheduling is solved. In order to approach problems (2) and (3) some scheduling criteria are considered:

C1. To optimize the use of the time blocks. In our problem, the ORs constitute the bottleneck because they can only be open during a given period

of time every working day. This constraint is imposed by budget reasons and cannot be violated. So, it is really important to improve the efficiency of the ORs in order to reduce the waiting time of the patients and consequently to improve the quality of the service. Therefore, the first scheduling criterion is to optimize the use of the time blocks. In this way, we prevent underutilization of the ORs (problem 2 in the manual scheduling). In Sec. 5.2 this optimization is based on obtaining a given occupation rate in each time block while in Sec. 5.3 the optimization is based on maximizing the occupation rate of each time block (constrained by the probability of exceeding the available time).

C2. The total available time in a surgical block should not be exceeded. An over-scheduling of the time blocks is not desirable. This happens because normally, time blocks are scheduled sequentially, that is, there is a small break time between two consecutive blocks. So, if there is not enough time in a block to perform the last scheduled surgery: a) it is canceled and the real occupation rate is lower than the expected or b) it is performed and the allocation time is exceeded, delaying the starting of the next block. In Sec. 5.2 it is assumed that the objective occupation rate fixed by doctors is “appropriate” and consequently is not expected to exceeding the available time. On the other hand, in Sec. 5.3 is assumed that the surgeries duration, as well as other temporal variables (starting delayed, cleaning time) composing the total duration of a time block are not deterministic. So, a probability constraint that ensures with a minimum confidence level that the total available time in a surgical block is not going to be exceeded is imposed. In this way, we prevent overutilization of the ORs (problem 2 in the manual scheduling).

C.3 Respect as much as possible the order of the patients in the waiting list. Due to ethical and common sense reasons, patients with the same urgency level should spend a similar time in the waiting list. For that, once the patients in a waiting list are ordered depending on the urgency level and the admission date, the third scheduling criterion is to respect as much as possible the order of the patients in the waiting list. This means that in the first time blocks should be scheduled the patients who are in the first position in the waiting list. In this way, objective criteria are considered to select the patients (solving problem 3 in the manual scheduling). Criterion C3 is especially important in surgery services where the number of time blocks weekly booked for a given team is low. For example, in the Orthopedic Surgery

Department (OSD) of the “Lozano Blesa” Hospital (LBH), each team has 2 time blocks per week, so the difference of scheduling a patient in the first time block with the sixth one is 21 days.

5.1.3 Problem Statement

In this subsection, the notation, the terminology, and the assumptions of the **scheduling problem** are introduced formally.

Let $\mathcal{S} = \{s_1, s_2, \dots, s_{|\mathcal{S}|}\}$ be the set of surgery types that can be performed in the medical department and let us assume that the duration $d(s_k)$ of each type of surgery $s_k \in \mathcal{S}$ is a random variable with normal probability density function (pdf) $d(s_k) = N(\mu_{d(s_k)}, \sigma_{d(s_k)})$, where $\mu_{d(s_k)}$ is the average and $\sigma_{d(s_k)}$ is the standard deviation.

Furthermore, let us consider $\mathcal{W} = \{w_1, w_2, \dots, w_{|\mathcal{W}|}\}$ an ordered list of patients such that if $w_j \in \mathcal{W}$, j is the preference order number of the patient w_j in the waiting list.

In addition, let us assume an ordered set of time blocks $\mathcal{B} = \{b_1, \dots, b_{|\mathcal{B}|}\}$ to schedule, where $b_{|\mathcal{B}|}$ is the block corresponding to the latest date. Each block $b_i \in \mathcal{B}$ has a fixed duration denoted by $l(b_i)$.

For each time block $b_i \in \mathcal{B}$ to schedule there exist a binary decision vector $\mathcal{S}_i \in \mathcal{S}_i^{|\mathcal{W}|}$ with a dimension equal to the number of the patients in the waiting list. If $\mathcal{S}_i[j] = 1$ then surgery of patient w_j should be performed in time block b_i .

The goal of the scheduling problem is the assignment of the patients from the waiting list \mathcal{W} to the set of time blocks \mathcal{B} considering the scheduling criteria C1, C2, and C3.

Let us define the following notations,

- $\boldsymbol{\mu}_w$ is a row vector containing the duration of surgeries of the patients in the waiting list \mathcal{W} . For example, $\boldsymbol{\mu}_w(j)$ represents the average duration of the surgery corresponding to the patient w_j .
- Similarly, $\boldsymbol{\sigma}_w$ is a row vector containing the standard deviation of surgeries of the patients in the waiting list \mathcal{W} .

5.2 Surgery scheduling problem imposing a given occupation rate

In this section, we approach criterion C1 and C2 by scheduling patients in such way that a given occupation rate is obtained in each time block. For that, three different models for the proposed Scheduling Problem are discussed. These models use a similar penalty in the objective function according to the preference order of the patients. However, each model takes into account in a different way the penalty related with the deviation of the occupation rate with respect to the target. A model with a quadratic cost function called *Quadratic Assignment Problem* (QAP) is explained in Sec. 5.2.1. A *Mixed Integer Linear Programming* (MILP) model is given in Sec. 5.2.2 and the scheduling problem modeled as a *Generalized Assignment Problem* (GAP) is discussed in Sec. 5.2.3.

5.2.1 Scheduling problem modeled as a QAP

In order to respect the preference order of the patients in the surgery scheduling (criterion C3), we assign a cost to each patient depending on the time block in which he/she is scheduled. This cost is given by the matrix \mathbf{P} , where the element $\mathbf{P}[i, j]$ represents the cost of schedule patient w_j in time block b_i . The following value is used: $\mathbf{P}[i, j] = j \cdot (|\mathcal{B}| - i + 1)$. For example, assuming $|\mathcal{W}| = 9$ patients in the waiting list and $|\mathcal{B}| = 3$ time blocks, matrix \mathbf{P} is given in (5.1). The first patients in the waiting list have a lower cost than the patients with higher preference order $\mathbf{P}[\cdot, 1]^* < \mathbf{P}[\cdot, 2] < \dots < \mathbf{P}[\cdot, |\mathcal{W}|]$. Moreover, the first time blocks to perform have a higher cost than the last ones $\mathbf{P}[1, \cdot] > \mathbf{P}[2, \cdot] > \dots > \mathbf{P}[|\mathcal{B}|, \cdot]$. In this way we are giving preference to schedule the first patients of the waiting list in the first blocks.

$$\mathbf{P} = \begin{bmatrix} 3 & 6 & 9 & 12 & 15 & 18 & 21 & 24 & 27 \\ 2 & 4 & 6 & 8 & 10 & 12 & 14 & 16 & 18 \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \end{bmatrix} \begin{array}{l} \rightarrow b_1 \\ \rightarrow b_2 \\ \rightarrow b_3 \end{array} \quad (5.1)$$

$$\mathbf{P}[i, j] = j \cdot (|\mathcal{B}| - 1 + i) \quad \forall j \in \{1, \dots, |\mathcal{W}|\}; i \in \{1, \dots, |\mathcal{B}|\}$$

*In this chapter the following matrix notation is assumed: $\mathbf{P}[i, \cdot]$ represents row i of the matrix \mathbf{P} and $\mathbf{P}[\cdot, i]$ represents the column i of matrix \mathbf{P} .

The cost associated with the order of the patients scheduled in the time block b_i is shown in (5.2)

$$\mathbf{P}[i, \cdot] \cdot \mathbf{S}_i \quad \forall i \in \{1, \dots, |\mathcal{B}|\} \quad (5.2)$$

Let us assume that the occupation rate of a time block is defined as the sum of time in which a patient is within the OR divided by the total available time.

In order to obtain a given occupation rate p of a time block b_i , let us denote by $Obj_i = l(b_i) \cdot p$ the target occupation in minutes of the time block b_i , where $l(b_i)$ is the daily time available in b_i and p is the desired occupation rate. The penalty associated with the occupation rate of the time block b_i is defined as

$$(\boldsymbol{\mu}_w \cdot \mathbf{S}_i - Obj_i)^2 \quad \forall i \in \{1, \dots, |\mathcal{B}|\}. \quad (5.3)$$

In (5.3) the term $\boldsymbol{\mu}_w \cdot \mathbf{S}_i$ is the sum of the durations of the surgeries scheduled in time block b_i . So, the objective is to minimize the square of the difference between the expected surgery time ($\boldsymbol{\mu}_w \cdot \mathbf{S}_i$) and the objective surgery time (Obj_i).

Putting together both costs (5.2) and (5.3), the objective is:

$$\min \sum_{i=1}^{|\mathcal{B}|} (\beta \cdot \mathbf{P}[i, \cdot] \cdot \mathbf{S}_i + (\boldsymbol{\mu}_w \cdot \mathbf{S}_i - Obj_i)^2) \quad (5.4)$$

Where β is a parameter to establish a compromise between the cost associated with the objective of respecting the order of the patients in the waiting list (criterion C3) and the cost associated with the objective of obtaining a target occupation rate (criterion C1 and C2). The full model is shown in (5.5), where the set of constraints $\sum_{i=1}^{|\mathcal{B}|} \mathbf{S}_i[j] \leq 1$ impose that each patient w_j is scheduled no more than one.

$$\begin{aligned} & \min \sum_{i=1}^{|\mathcal{B}|} (\beta \cdot \mathbf{P}[i, \cdot] \cdot \mathbf{S}_i + (\boldsymbol{\mu}_w \cdot \mathbf{S}_i - Obj_i)^2) \\ & \text{Subject to:} \\ & \left\{ \begin{array}{ll} \sum_{i=1}^{|\mathcal{B}|} \mathbf{S}_i[j] \leq 1, & j \in \{1, \dots, |\mathcal{W}|\}, \\ \mathbf{S}_i[j] \in \{0, 1\}, & i \in \{1, \dots, |\mathcal{B}|\}; j \in \{1, \dots, |\mathcal{W}|\}. \end{array} \right. \end{aligned} \quad (5.5)$$

Proposition 5.1. *The proposed scheduling problem (5.5) is NP-hard.*

Proof. We prove this theorem by modeling it as a Quadratic Assignment Problem (QAP). That is proved to be NP-hard [11]. Hence our problem will be NP-hard.

QAP can be described, using the terminology of knapsack problems, as follows: Given n items and m knapsacks, let \mathbf{A}_i be the *cost matrix* of assigning items to knapsack i , $\forall i \in \{1, \dots, m\}$, defined as follows:

$$\mathbf{A}_i[j, k] = \begin{cases} \triangleright \text{cost of assign item } j \text{ to knapsack } i & \text{if } j = k \\ \triangleright \frac{1}{2} \text{ of cost of assign items } j \text{ and } k \\ \quad \text{simultaneously to knapsack } i & \text{if } j \neq k \end{cases}$$

The problem is to assign each item at most to one knapsack so as to minimize the total cost assigned.

In order to model our scheduling problem as a QAP we consider that:

1. The set of time blocks \mathcal{B} are the knapsacks.
2. The set of patients in the waiting list \mathcal{W} are the items.
3. The cost matrices \mathbf{A}_i is obtained from (5.4), in particular from term i of the sum. Since in the second part of the term there is a quadratic term, the following manipulations can be done:

$$(\boldsymbol{\mu}_w \cdot \mathbf{S}_i - Obj_i)^2 = (\boldsymbol{\mu}_w \cdot \mathbf{S}_i)^2 + Obj_i^2 - 2 \cdot Obj_i \cdot \boldsymbol{\mu}_w \cdot \mathbf{S}_i$$

Since Obj_i^2 is a constant, it can be removed from the cost function obtaining,

$$(\boldsymbol{\mu}_w \cdot \mathbf{S}_i)^2 - 2 \cdot Obj_i \cdot \boldsymbol{\mu}_w \cdot \mathbf{S}_i = \mathbf{S}_i^T \cdot \mathbf{D} \cdot \mathbf{S}_i - 2 \cdot Obj_i \cdot \boldsymbol{\mu}_w \cdot \mathbf{S}_i \quad (5.6)$$

where \mathbf{D} is a square and symmetric matrix such that: $\mathbf{D}[i, j] = \boldsymbol{\mu}_w[i] \cdot \boldsymbol{\mu}_w[j]$, $\forall i, j \in \{1, \dots, |\mathcal{W}|\}$. So the term i of the cost function in (5.4) can be written as follow:

$$(\beta \cdot \mathbf{P}[i, \cdot] - 2 \cdot Obj_i \cdot \boldsymbol{\mu}_w) \cdot \mathbf{S}_i + \mathbf{S}_i^T \cdot \mathbf{D} \cdot \mathbf{S}_i \quad (5.7)$$

Finally, because \mathbf{S}_i is a binary vector, we can include the linear term of (5.7) (i.e., $(\beta \cdot \mathbf{P}[i, \cdot] - 2 \cdot \text{Obj}_i \cdot \boldsymbol{\mu}_w) \cdot \mathbf{S}_i$) in the diagonal of the matrix \mathbf{D} resulting the next quadratic cost function:

$$\min \sum_{i=1}^{|\mathcal{B}|} \mathbf{S}_i^T \cdot \mathbf{A}_i \cdot \mathbf{S}_i \quad (5.8)$$

where,

$$\mathbf{A}_i[j, k] = \begin{cases} \triangleright & \mathbf{D}[j, k] + \beta \cdot \mathbf{P}[i, j] - 2 \cdot \text{Obj}_i \cdot \boldsymbol{\mu}_w[j] & \text{if } j = k \\ \triangleright & \mathbf{D}[j, k] & \text{if } j \neq k \end{cases}$$

□

The Scheduling surgery problem can be put has a Binary Quadratic Problem (BQP):

$$\begin{aligned} & \min \sum_{i=1}^{|\mathcal{B}|} \mathbf{S}_i^T \cdot \mathbf{A}_i \cdot \mathbf{S}_i \\ & \text{Subject to:} \\ & \begin{cases} \sum_{i=1}^{|\mathcal{B}|} \mathbf{S}_i[j] \leq 1, & j \in \{1, \dots, |\mathcal{W}|\}, \\ \mathbf{S}_i[j] \in \{0, 1\}, & i \in \{1, \dots, |\mathcal{B}|\}, j \in \{1, \dots, |\mathcal{W}|\} \end{cases} \end{aligned} \quad (5.9)$$

5.2.2 Scheduling problem modeled by a MILP

In this subsection, we propose a MILP model [21] for the scheduling problem. This model is very similar to (5.5). However, the cost associated to the occupation rate (i.e., cost given by (5.3)) is replaced by the absolute deviation in minutes between Obj_i and $\boldsymbol{\mu}_w \cdot \mathbf{S}_i$. This change avoids quadratic terms in the objective function.

$$\|\text{Obj}_i - \boldsymbol{\mu}_w \cdot \mathbf{S}_i\|_2 \Rightarrow \|\text{Obj}_i - \boldsymbol{\mu}_w \cdot \mathbf{S}_i\|_1$$

In order to change the objective function to replace norm 2 to norm 1, in addition to the decision variables $S_i[j]$ (that indicate if surgery w_j is scheduled in time block b_i), let us define the new variable $\boldsymbol{\alpha} \in \mathbf{R}_{\geq 0}^{|\mathcal{B}|}$. It is a vector of dimension equal to the number of blocks $|\mathcal{B}|$ where each element α_i is a real

variable representing the absolute deviation (in minutes) between the surgery time in the block b_i with respect to the objective, i.e., $\alpha_i = |Obj_i - \boldsymbol{\mu}_w \cdot \mathbf{S}_i|$. That is equivalent with the minimum α_i that fulfill the following constraints

$$\begin{cases} \boldsymbol{\mu}_w \cdot \mathbf{S}_i - Obj_i \leq \alpha_i \\ \boldsymbol{\mu}_w \cdot \mathbf{S}_i - Obj_i \geq -\alpha_i \end{cases} \quad \forall i = 1, 2, \dots, |\mathcal{B}| \quad (5.10)$$

Two constraints will be necessary to define each variable α_i . Since the number of variable α_i is equal to the timing horizon (i.e., number of time blocks to schedule $|\mathcal{B}|$), we need $2 \times |\mathcal{B}|$ constraints to define all variables α_i .

Notice that the QAP has $|\mathcal{W}| \cdot |\mathcal{B}|$ binary variables and $|\mathcal{W}|$ constraints while the MILP model has $|\mathcal{W}| \cdot |\mathcal{B}|$ binary variables plus $|\mathcal{B}|$ continuous variables and $|\mathcal{W}| + 2 \cdot |\mathcal{B}|$ constraints. Although the new model is larger, it is a MILP model without quadratics term which is in general computationally more efficient. The full MILP is as follows:

$$\begin{aligned} & \min \sum_{i=1}^{|\mathcal{B}|} (\beta \cdot \mathbf{P}[i, \cdot] \cdot \mathbf{S}_i + \alpha_i \cdot (|\mathcal{B}| - i + 1)) \\ & \text{Subject to:} \\ & \begin{cases} \boldsymbol{\mu}_w \cdot \mathbf{S}_i - Obj_i \leq \alpha_i, & \forall i = 1, 2, \dots, |\mathcal{B}| \\ -\boldsymbol{\mu}_w \cdot \mathbf{S}_i + Obj_i \leq \alpha_i, & \forall i = 1, 2, \dots, |\mathcal{B}| \\ \sum_{i=1}^{|\mathcal{B}|} \mathbf{S}_i[j] \leq 1, & \forall j = 1, 2, \dots, |\mathcal{W}| \end{cases} \end{aligned} \quad (5.11)$$

Now the objective function is a *linear cost function* composed by two terms. Variable α_i of the second term penalizes the deviation of the occupancy of the time block with respect to the objective. Since $(|\mathcal{B}| - i + 1)$ is multiplying α_i , it gives more importance to obtain a better occupancy rate in the first time blocks. In this way, if there are not enough patients for all time blocks, the last ones remain free.

5.2.3 Scheduling problem modeling by a GAP

In this subsection, the scheduling problem is modeled as a Generalized Assignment Problem (GAP). Using the terminology of the patients scheduling problem, the GAP problem can be described as follow:

Given $|\mathcal{W}|$ patients and $|\mathcal{B}|$ time blocks, with

- $\bar{P}[i, j] = \text{profit}$ of patients w_j if assigned to time block b_i ,
- $\mu_w[j] = \text{duration}$ of the surgery to perform in patient w_j ,
- $c_i = \text{maximum capacity}$ (in minutes) allowed in the time block b_i ,

assign each patient to exactly one time block in such way that: (i) the maximum capacity allowed c_i in each time block b_i is not exceeding; and (ii) the maximum total profit is obtained.

Since the objective in the GAP is to maximize the profit, the new \bar{P} profit matrix is defined as follow:

$$\bar{P}[i, j] = (|\mathcal{W}| + 1 - j) \cdot (|\mathcal{B}| - 1 + i) \quad \forall j \in \{1, \dots, |\mathcal{W}|\}; i \in \{1, \dots, |\mathcal{B}|\}$$

Assuming $|\mathcal{W}| = 9$ patients in the waiting list and $|\mathcal{B}| = 3$ time blocks, matrix \bar{P} for the GAP problem is given in (5.12).

$$\bar{P} = \begin{bmatrix} 27 & 24 & 21 & 18 & 15 & 12 & 9 & 6 & 3 \\ 18 & 16 & 14 & 12 & 10 & 8 & 6 & 4 & 2 \\ 9 & 8 & 7 & 6 & 5 & 4 & 3 & 2 & 1 \end{bmatrix} \begin{array}{l} \rightarrow b_1 \\ \rightarrow b_2 \\ \rightarrow b_3 \end{array} \quad (5.12)$$

The patients with lower preference order have a greater profit than the patients with higher preference order: $P[\cdot, 1] > P[\cdot, 2] > \dots > P[\cdot, |\mathcal{W}|]$. Moreover, the first time blocks to perform have a higher profit than the last ones $P[1, \cdot] > P[2, \cdot] > \dots > P[|\mathcal{B}|, \cdot]$. In this way we are forcing to schedule patients with lower preference order in the first time blocks.

The problem can be put as Binary Integer Linear Programming (BILP) as follows:

$$\begin{array}{l} \max \sum_{i=1}^{|\mathcal{B}|} [\bar{P}[i, \cdot] \cdot \mathbf{S}_i] \\ \text{Subject to:} \\ \left\{ \begin{array}{l} \boldsymbol{\mu} \cdot \mathbf{S}_i \leq c_i, \quad \forall i = 1, 2, \dots, |\mathcal{B}| \\ \sum_{i=1}^{|\mathcal{B}|} \mathbf{S}_i[j] = 1, \quad \forall j = 1, 2, \dots, |\mathcal{W}| \end{array} \right. \end{array} \quad (5.13)$$

Note that GAP model (5.13) does not control directly the occupation rate obtained because there is no penalty in the cost function related to it. However, in this model, all patients in the waiting list should be scheduled. Therefore,

for a given waiting list of patients, the model should be solved with enough number of time blocks. If a solution of (5.13) is obtained, the number of time blocks is decreased in one unit and the problem (5.13) is solved again. This process is repeated until no solution is obtained. The last obtained solution uses the smallest number of blocks, and consequently, their occupation rates are near to the maximum allowed capacities (c). For this reason, the capacity assigned to each OR is a little greater (2 %) than the objective p . In this way, the average value of the occupation rate will be near the target.

5.2.4 Heuristic methods

In this section, different heuristics solution methods are proposed for solving the mathematical programming problems (5.5), (5.11) and (5.13). Particularly, a meta-heuristic *Genetic Algorithm* (GA) for the QAP (5.5) is explained in Sec. 5.2.4, a *Receding Horizon Strategy* (RHS) used to solve the MILP model (5.11) is given in Sec. 5.2.4. Finally, a heuristic *Steepest Descent Multiplier Adjustment Method* (SDMAM) for the GAP (5.13) is provided in Sec. 5.2.4.

Genetic Algorithm for solving QAP

It has been proved that the QAP (5.5) is NP-hard. Moreover, there are no efficient approximation algorithms in polynomial time to solve QAP [11]. In Sec. 5.2.5 it will be shown that very small instances of problem (5.5) containing maximum 10 patients and 3 time blocks can be solved optimally using CPLEX in a “reasonable” time. This small size of the resolvable instances it is not appropriate to consider a heuristic iterative method using receding horizon strategy. On the contrary, we propose a meta-heuristic *genetic algorithm* (GA) which is a nature-inspired strategy for optimization problems. The basic idea is to adapt the evolutionary mechanisms acting in the selection process in nature to combinatorial optimization problems. The first genetic algorithm for optimization problems was proposed by Holland [35] in 1975. The function “ga” of the globaloptim toolbox of Matlab is used for its implementation. The principal parameters of this function are the following:

- *PopulationSize*: Size of the population.

- *EliteCount*: Positive integer specifying how many individuals in the current generation are guaranteed to survive to the next generation.
- *CrossoverFraction*: The fraction of the population at the next generation, not including elite children, that is created by crossover.
- *FunctionTolerance* and *MaxstalGenerations*: The algorithm stops if the average relative change in the best fitness function value over MaxStall-Generations generations is less than or equal to FunctionTolerance.

Heuristic iterative method to solve MILP

In order to solve large instances of MILP model (5.11), we propose a heuristic approach called *Receding Horizon Strategy* (RHS) which obtains sub-optimal solution sequentially (similar to [8]). Fig. 5.1 shows the methodology of the RHS. Mainly, this method is as follow: from the waiting list of patients, the first k time blocks are scheduled. From these k time blocks, we consider the scheduling of the first s (where $s \leq k$) to be included in the final scheduling. The patients with surgeries scheduled in these s time blocks are removed from the waiting list and the process is repeated until all desired time blocks have been scheduled.

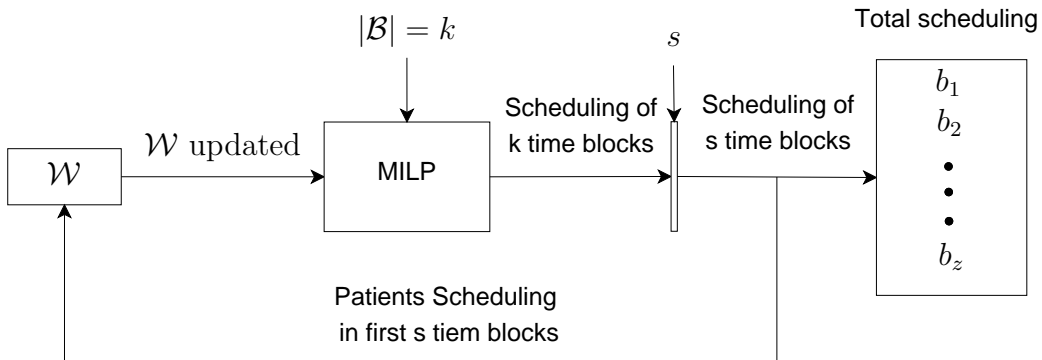


Figure 5.1: Receding horizon Strategy

A Steepest Descent Multiplier Adjustment Method for the GAP

The GAP problem is NP-complete [58], however, unlike QAP [11], there exists efficient approximation algorithms in polynomial time [61, 41]. A constructive heuristic method called *Steepest Descent Multiplier Adjustment Method for the Generalized Assignment Problem* [41] is used to solve large instances of model (5.13). This method uses a branch-and-bound algorithm that incorporates an improved Multiplier Adjustment Method (MAM) as a bounding tool. MAMs are heuristic algorithms for solving Lagrangian duals of the Generalized Assignment Problem (GAP) exploiting their special structure. The proposed algorithm uses an improved version of the traditional MAM by incorporating a post-optimality analysis for the 0-1 knapsack subproblems based on a dynamic programming formulation. The algorithm guarantees a steepest descent required by the traditional MAM for calculating a step length.

5.2.5 Simulation results

In this section, the different models and their heuristic approximation methods solutions are numerically analyzed. First, the computational efficiency of exact and heuristic solution methods is shown, then the scheduling obtained using the proposed heuristics approaches are compared according to the criteria of occupation rate and order of the patients.

The numerical input data of the models (waiting list) has been randomly generated using realistic data from the Orthopedic Surgery Department (OSD) of the “Lozano Blesa” Hospital (LBH) in Zaragoza. In particular, we used a lot of average durations for different pathologies that have been operated in the mentioned department during the last two years. We have used some probabilities computed based on the real data to generate “randomly” the type of pathologies of the patients. All solutions have been obtained using a computer with an Intel Core i5 and 8 GB of memory.

Computational efficiency of the methods

The purpose of this sub-section is to analyze the computational efficiency of the models and their heuristic approximation methods. For each one, the computational times to solve instances of different sizes are shown, which al-

lows knowing the limitations (due to computational time) of each model or heuristic approximation method. The exact solutions of the problems have been obtained by using the IBM ILOG CPLEX Optimization Studio which is often referred as CPLEX [36] which is a commercial solver designed to tackle (among others) large scale (mixed integer) programming problems. CPLEX is now actively developed by IBM and it is one of the fastest software solution for MILP problems [30].

Exact solution method of QAP (5.9). A large number of solutions of model (5.9) with a different number of patients in the waiting list and a different number of time blocks to schedule have been obtained using CPLEX. In these simulations, the duration of time blocks is 390 [minutes], the desired occupation rate is 78% and a value of parameter β of 20 is considered. After performing some simulations has been observed that this is the best value of β from the medical point of view in order to balance the two objectives (occupation rate and preference order). Table 5.1 shows the average computational time necessary to solve different instances of problem (5.9). The first column represents the number of time blocks to schedule, the second column indicates the number of patients in the waiting list and the last column is the average computational time necessary to solve the corresponding instance.

Table 5.1: Computational time to solve optimally different instances of the QAP (5.9)

$ \mathcal{B} $	$ \mathcal{W} $	Avg. Time [s]
3	10	3,63
	11	11,03
	12	39,56
4	12	463
	13	1933
5	15	-

Note that only small instances composed of 3 time blocks and 10-12 patients can be solved in a reasonable time. Moreover, in instances with more than 4-time blocks to schedule, the solution cannot be obtained.

Meta-Heuristics Genetic Algorithm of model (5.9). The function “ga” of the “globaloptim” toolbox of Matlab has been used to solve the scheduling problem. In particular we solve the model with quadratic cost function (5.9) assigning a value of $\beta = 20$. After analyzing the results obtained with different values of the parameters using the “ga” function, we have fixed the next values:

- PopulationSize = Nvar · 10 where *Nvar* is the number of variables ($n \cdot m$)
- EliteCount = 0.3 · PopulationSize
- CrossoverFraction = 0.6
- FunctionTolerance = $1e^{-15}$
- MaxstalGenerations = 20

Table 5.2 shows the average computational time obtained for different instances of the ga.

Table 5.2: Computational time to solve different instances of the QAP (5.9) using GA

# Blocks	# Patients	Avg. Time [s]
5	20	83
10	35	426
15	50	2176
20	65	7629
25	80	21542

In all instances studied in Tab. 5.2 the GA converged to a solution. The computational times necessary are very big and, in addition, as it is shown in Section 5.2.5, the solutions obtained according to the order of the patients are very poor.

Exact solution method of the MILP model (5.11). After solving different instances of MILP problem using CPLEX, it has been observed that the most influential variable in the computational time is the number of blocks to schedule. Considering the duration of time blocks of 390 [minutes], the desired occupation rate of 78%, a value of parameter β of 2 and waiting lists composed

by 70 patients, we have computed the average computational time necessary to schedule different number of time blocks. Tab. 5.3 shows the results obtained. The value of $\beta = 2$ has been chosen from the medical point of view as the better.

Table 5.3: Computational time to solve optimally different instances of model (5.11)

# Blocks	Avg. Time [s]
3	0,1421
4	0,1965
5	0,3953
6	0,8311
7	3,083
8	5,97
9	22,31
10	749
12	-

It can be seen that this model is computationally more efficient than (5.9). However, in order to schedule more than 9-time blocks in a reasonable time is necessary to use the heuristic approach based on RHS. Furthermore, the memory necessary to solve some instance of 12-time blocks is not enough and consequently, the simulation stops being out of memory after 6 hours of computation.

Iterative method (RHS) solving (5.11). The expected computational time (ET) necessary to schedule m time blocks using the iterative method explained in Section 5.2.4 is the following:

$$ET = \left\lceil \frac{m}{s} \right\rceil \cdot Avg(k)$$

where $\lceil \cdot \rceil$ is the rounding to the next integer, $Avg(k)$ is the average computational time to schedule a time horizon of k time blocks, and s is the number of blocks to select from the k scheduled. For example, assuming $m = 20$ time blocks to schedule, $k = 7$ and $s = 5$, the expected computation time to

schedule the 20-time blocks is:

$$ET = \left\lceil \frac{20}{5} \right\rceil \cdot Avg(7) = 4 \cdot 3.08 = 12.32[s]$$

Exact solution method for solving GAP (5.13). Tab. 5.4 shows the average computational time to solve optimally different instances of the GAP (5.13). Time blocks of 390 minutes with a maximum capacity (c) of 80% has been considered. As mentioned in Sec. 5.2.3 the GAP schedules all patients in the waiting list, so a sufficiently large number of time blocks must be considered initially to get a solution. In order to maximize the occupation rate, if a solution is found, the same problem is solved again but decreasing in one unit the number of blocks to schedule. The last instance in which a solution can be found has the highest average occupation rate. The computational time to solve these last instances has been taken into account to obtain the average computational time. Note that the previous iteration where some blocks remain (partially) free are obtained in few second.

Table 5.4: Computational time to solve optimally different instances of scheduling problem (5.13)

# Patients	Avg. Time [s]
10	0.05
20	0.7
30	-

It can be seen that some instance with more than 30 patients to schedule can not be solved being out of memory.

Steepest Descent MAM for the GAP. The implementation of the Steepest Descent MAM for the GAP proposed by Nejat Karabakal in [40] is used to solve the scheduling problem (5.13). We consider again time blocks of 390 minutes with a maximum capacity of $c = 80\%$. Tab. 5.5 shows the computational times necessary to schedule a different number of patients.

The heuristic Steepest Descent MAM algorithm for GAP is computationally very efficient. Moreover, as it is shown in Section 5.2.5, the solution

Table 5.5: Computational time to solve different instances of GAP (5.13) using Steepest Descent MAM for the GAP

# Patients	Avg. Time [s]
10	0.27
20	0.71
30	2.07
40	10.24
50	15.69
60	19.32

obtained according to the occupation rate and order of patients are very interesting.

The computational times to solve the QAP (5.9), MILP (5.11) and the GAP (5.13) optimally are very high and only small instances can be solved in a reasonable time. In order to solve larger instances, we have tested and compared three heuristic approximation methods (one for each model): a meta-heuristic GA for solving the QAP, an RHS for solving the MILP and an SD-MAM for the GAP. The results show that using the RHS and the SD-MAM it is possible to schedule 22 blocks in a reasonable time: 15.41 [s] and 19,32 [s] respectively. However, the meta-heuristic GA spends 7629 [s] in the scheduling of 20 blocks.

Comparing the quality of the solutions

Now, the quality of the scheduling obtained using the different heuristic approximation methods are compared. Particularly, we use the *RHS to solve the MILP*, the *SDMAM for the GAP* and the meta-heuristic *GA for the QAP*. It has been considered instances of three different sizes, which are shown in Tab. 5.6.

Table 5.6: Instances to evaluate

INSTANCE	PATIENTS ($ \mathcal{W} $)	TIME BLOCKS ($ \mathcal{B} $)
1	30	11
2	45	18
3	60	22

The duration of the blocks is 390 minutes and the target occupation rate is fixed to 78%. The RHS schedules 8-time blocks in each iteration ($k=8$) and selects the first six blocks ($s=6$). Moreover, a value of $\beta = 2$ is fixed in the MILP problem (5.11) for each iteration. Using the SDMAM for the GAP a maximum capacity (c) of 80% (2% larger than the target) is imposed. Finally, the solution for the meta-heuristic GA has been obtained with the same values of the parameters as those used in the computational time simulations.

For each one of the three instances and for each model, 200 replications has been performed for computing the average values. In order to be able to compare two different scheduling from the point of view of the order of the patients, the indicator Ω is defined. This indicator measures the disorder of the patients in the obtained scheduling, so the smaller it is, the more orderly are the patients in the scheduling. To compute this value, for each time block scheduled b_i we define an interval $[f_i, l_i]$. If the preference order of the surgeries scheduled in the time block b_i belong to the interval $[f_i, l_i]$ do not increase the value of Ω . On the contrary, each patient with a preference order outside the interval increases the value of Ω . The formal calculation of Ω is given in Algorithm 6 where Np is the total number of patients scheduled and Pd is the average number of patients scheduled per time block.

Algorithm 6: Calculation of Ω parameter in a scheduling of $|\mathcal{B}|$ time blocks

```

1:  $\Omega := 0$ 
2:  $Np := \sum_{i=1}^m (sum(S_i))$ 
3:  $Pd := \frac{Np}{|\mathcal{B}|}$ 
4: for all  $b_i \in \mathcal{B}$  do
5:    $f_i := \min(1, \lfloor Pd \cdot (i - 1) \rfloor - 3)$ 
6:    $l_i := \lceil Pd \cdot i \rceil + 4$ 
7:   for all  $w_j$  scheduled the time block  $b_i$  do
8:     if  $j \notin [f_i, l_i]$  then
9:        $\Omega := \Omega + \min(|j - f_i|, |j - l_i|)$ 
10:    end if
11:  end for
12: end for

```

In order to compare different scheduling two aspects have been taken into

account:

1. *Occupation rate*: a static analysis of the occupation rate.
2. *Order of the patients from the waiting list*: The parameter related to the order of the patients Ω (given in Alg. 6).

Tab. 5.7 shows a comparison of the solution obtained by using the three proposed heuristic approximation methods for the 3 instances considered. The first column indicates the instance to solve, the second column represents the heuristic approximation method (model) used to obtain the scheduling, the next fourth columns show a static analysis of the occupation rate including the average, the extreme values and the standard deviation. Finally, the last column evaluates the order of the patients with the indicators Ω .

Table 5.7: Comparing of surgery scheduling using the RHS, SDMAM and Meta-Heuristic GA

Instance	Model	Occupation Rate				Order
		Avg.	Max.	Min.	Std. Dev.	Ω
1 ($ \mathcal{B} $)=11 ($ \mathcal{W} $)=30	RHS	77.8	78.5	76.15	0.76	25
	SDMAM	77.5	79.7	72.56	2.38	34
	GA	72.6	90	60	8.46	324
2 ($ \mathcal{B} $)=17 ($ \mathcal{W} $)=45	RHS	77.9	78.46	58.79	4.46	60
	SDMAM	77.5	79.7	58.79	5.91	93
	GA	63.6	90.25	0	23.33	1023
3 ($ \mathcal{B} $)=22 ($ \mathcal{W} $)=60	RHS	77.75	78.94	63.3	3.11	97
	SDMAM	77.47	79.83	66	3.19	237
	GA	67.5	90	45.1	11.7	2700

It can be seen that the *RHS solving MILP* obtains the best scheduling in terms of occupancy rate and patient order. For each instance, using the RHS, the lowest standard deviation of the occupancy rate is obtained (0.76, 4.46 and 3.11). Moreover, the average occupation rate is the closest to the target value (77.8, 77.9 and 77.75). According to the preference order of the patients, the lowest values of Ω (25, 60 and 97) are obtained for each instance using the RHS. So, using this approach the best scheduling are obtained according to both objectives (occupation rate and preference order).

The solutions obtained using the heuristic *SDMAM for the GAP* are good according to the occupation rate. The averages values for the instances are 77.5, 77.5 and 77.47 respectively, very similar to those obtained using the iterative method. However, the standard deviations are a little larger (2.38, 5.91 and 3.19). On the contrary, the scheduling obtained using the heuristic *SDMAM for the GAP* respects less the order of the patients than those obtained with the iterative method. It can be seen in the values of the indicator Ω (34, 93 and 237). This disorder of the scheduling is acceptable from the clinical point of view. So, the scheduling obtained by using this model is not as good as that obtained by using the iterative one, however, it is acceptable from the medical point of view.

Finally, using the meta-heuristic *GA*, very bad solutions are obtained according to both criteria of occupation rate and order of the patients. The average values of the occupation rate are far to the objective (72.6, 63.6 and 67.5). Moreover, the standard deviations are very high (8.46, 23.33 and 11.7). The scheduling obtained using *GA* do not respect the order of the patients in the waiting list and consequently, very high values of the indicators Ω (324, 1023 and 2700) are obtained. The scheduling obtained using the meta-heuristic *GA* are not acceptable from the medical point of view. A general genetic algorithm implemented in the “globaloptim” toolbox of Matlab has been used. More specific heuristics rules should be considered in order to improve the solutions obtained.

Effect of parameter β in MILP problem

In order to fix, from a medical point of view, the best value of the parameter β , some simulations has been performed and subsequently analyzed by medical doctors. Remember that β parameter in the cost function of MILP (5.11): $\min \sum_{i=1}^{|\mathcal{B}|} (\beta \cdot \mathbf{P}[i, \cdot] \cdot \mathbf{S}_i + \alpha_i \cdot (|\mathcal{B}| - i + 1))$, establish a compromise between the objective of obtain a given occupation rate and the objective of respect as much as possible the order of the patients in the waiting list. So, greater is the parameter β , the more importance is given to the order of the patients.

Considering a same waiting list of $|\mathcal{W}| = 300$ patients and different values of β we have used the RHS to plan $|\mathcal{B}| = 60$ time blocks. The duration of each time block is $l(b_i) = 390$ minutes and a objective occupation rate of $p = 80\%$

has been imposed. A statistic analysis of the occupation rate is shown in Tab. 5.8. The average, standard deviation and the extreme values of the occupation rate are shown.

Table 5.8: MILP: Statistic analysis of occupation rate depending of β

β	Average	Deviation	Minimum	Maximum
3	79.278	1.554	70.952	81.667
2	80.340	1.033	77.619	81.905
1.5	80.340	0.932	77.857	81.905
1	80.340	0.719	78.333	81.667
$\frac{2}{3}$	80.183	0.703	78.333	81.667
0.5	80.238	0.624	78.333	81.190
$\frac{1}{3}$	80.238	0.621	78.571	81.905

It can be seen that by decreasing the parameter β better results of occupation rate are obtained: the standard deviation decreases and consequently the data are more concentrated around the average value. Unfortunately, this improvement is achieved by allowing a greater disorder in the scheduling. Tab. 5.9 shows the planning done for the first 5 time blocks with different values of β . The first column represents the value of β and the second one indicates the time block b_i . The next three columns represent the preference order of the patients scheduled in the corresponding block, i.e., a value of j indicates that patients w_j has been scheduled. Notice that if lower than 3 patients are scheduled in a block, a 0 value appears in some of these three columns. Finally, the last column shows the occupation rate of the block.

At most 3 surgeries per time block are scheduled but this is not always true and more surgeries could be scheduled. After performing several simulations with different waiting lists, a value of parameter $\beta = 2$ has been considering optimum for the OSD of the LBH

5.3 Surgery scheduling under uncertain durations

The scheduling solution of the proposed MILP (5.11) problem are obtained based on the average durations of each type of surgery which can be computed

Table 5.9: Patients scheduling in the 5 first time blocks for a same waiting list with different values of parameter β

β	Time block	Sur1	Sur2	Sur3	Occupation
3	b_1	1	2	4	77.857
	b_2	3	5	6	80.714
	b_3	7	8	10	80.714
	b_4	9	17	0	70.952
	b_5	12	13	14	80.714
2	b_1	1	2	6	80.714
	b_2	3	5	9	81.666
	b_3	7	8	10	80.714
	b_4	4	13	18	80.238
	b_5	11	12	15	79.524
1	b_1	1	2	6	80.714
	b_2	3	7	10	80.714
	b_3	5	8	9	81.666
	b_4	4	13	18	80.238
	b_5	11	12	15	79.524
$\frac{2}{3}$	b_1	1	2	6	80.714
	b_2	3	7	10	80.714
	b_3	5	8	9	81.666
	b_4	4	13	18	80.238
	b_5	11	12	15	79.524
$\frac{1}{3}$	b_1	1	2	6	80.714
	b_2	3	7	10	80.714
	b_3	5	11	15	79.524
	b_4	4	13	18	80.238
	b_5	8	9	29	80

by using historical data. However, two problems may appear,

- P_1 - the obtained scheduling could be not robust enough if the surgery durations have large standard deviations. This uncertainty could result in uncomfortable situations for the medical management staff, either the doctors that usually may lengthen their working day either low utilization of the ORs is obtained.
- P_2 - a target occupation rate p is an input parameter in the optimization problem and in some cases it is difficult to select a good value for it in order to get solutions not exceeding the available time but having a good OR utilization.

In order to overcome P_1 , we propose a *New Mixed Integer Quadratic Constrained Problem* (N-MIQCP) that uses not only the average duration of the surgeries but also their standard deviations. In this way, each type of surgery has a pair of values (mean and standard deviation) that define its duration. Additionally, it considers the cleaning time between surgeries and the delayed in the starting time as random variables (with mean and standard deviation). These new assumptions allow us to introduce some chance constraints allowing to impose a *Minimum Confidence Level* (Cl) not exceeding the available time in the blocks.

To tackle the problem P_2 stated before we change the objective function. Instead of trying to obtain a given occupation rate (as an input parameter) we consider in N-MIQCP problem the objective of maximizing this occupation rate (making it a variable) keeping the chance constraints. In this way, the efficiency of the ORs is improved not only by maximizing the theoretical occupation rate but also by minimizing the risk of having overtime which could result in the cancellations of surgeries and consequently reduces the real occupation rate in the ORs.

5.3.1 Related work

In our N-MIQCP approach, it is assumed that the uncertain parameters (surgery duration and cleaning time) follow a normal distribution and consequently, the expected total duration of an OR working day also follows a normal distribution. In this way, it is possible to require that the probability of exceeding

the available time be no more than a given scalar by introducing a capacity chance constraint. The idea of using a chance constraint for the scheduling of ORs is also used in [62, 34]. These approaches require to set in advance the patients that are going to be scheduled in the next blocks, therefore in these approaches all considered patients must be scheduled in one of the available blocks. For this, both approaches start with initial scheduling obtained through the scheduling rule: *first-fit probabilistic* (Π_{FFP}). Following this rule, sequentially each surgery is assigned to the first available block for which the probabilistic capacity constraint is satisfied after the assignment. Once the initial scheduling is obtained the expected occupation rate of the first blocks are improved by rescheduling the surgeries. In this way, the last blocks are totally or partially released.

In our case, an important criterion is the quality of the service, so the order of patients in the waiting list must be respected as much as possible (criterion C3). That is, first patients should be scheduled in the first surgical block, while last patients should be preferably scheduled in the last block. This consideration is not taken into account in the previously explained approaches because:

1. When obtaining the initial solution, patients who are far behind on the waiting list, but are suitable to complete a surgical block, may be scheduled.
2. Once the initial scheduling has been obtained following the first-fit probabilistic rule, the patients are rescheduled, and in the final solution, any patient can be assigned to any surgical block.

Unlike [62, 34], our approach does not require to know in advance the set of patients that should be scheduled in the next surgical blocks, since any patient on the waiting list may or may not be scheduled. Alternatively, we propose a linear cost function composed by two balanced terms that favor patients to be scheduled in an orderly manner at the same time that maximizes the expected occupation rate of the OR. Using realistic data, a comparison between the approaches proposed in the related works [62, 34] and the one explained in this work is performed and analyzed. The average results show that similar occupation rate and confidence level are obtained. However, using

our approach, the scheduling obtained respects more the preference order of the patients due to this criterion is considered in the definition of the problem.

5.3.2 Surgery scheduling under uncertain durations

In this section, a New Mixed Integer Quadratic Constrained Programming (N-MIQCP) model is proposed to solve the scheduling problem. This model assigns patients from the waiting list to the time blocks in such a way that *the occupation rate of the ORs is maximized* (criterion C1) at the same time that *a minimum confidence level of not exceeding the available time* is guaranteed (criterion C2). Moreover, it *respects as much as possible the order of the patients* in the waiting list (criterion C3)

In order to respect the preference order of the patients in the surgery scheduling (criterion C3) we consider exactly the same cost that was imposed in the MILP model (5.11). This cost is given by (5.2), where $P[i, j] = j \cdot (|\mathcal{B}| - i + 1)$ represents the cost of schedule patient w_j in time block b_i .

However, while in the MILP model the objective is to obtain a given occupation rate, in the N-MIQCP the objective is to maximize the occupation rate. So, criterion C1 is formalized by (5.14). This equation maximizes the expected surgery time in each block ($\mu_w \cdot S_i$) giving more importance to the first ones ($|\mathcal{B}| + 1 - i$).

$$\max \sum_{i=1}^{|\mathcal{B}|} [\mu_w \cdot S_i \cdot (|\mathcal{B}| + 1 - i)]. \quad (5.14)$$

Putting together both costs (5.2) and (5.14), the objective function in the N-MIQCP model is:

$$\min \sum_{i=1}^{|\mathcal{B}|} [\beta \cdot P[i, \cdot] \cdot S_i - \mu_w \cdot S_i \cdot (|\mathcal{B}| - i + 1)] \quad (5.15)$$

This is a linear cost function composed by two balanced terms where the parameter β establish a compromise between the first term ($P[i, \cdot] \cdot S_i$) and the second one ($\mu_w \cdot S_i \cdot (|\mathcal{B}| - i + 1)$).

The second term in the cost function tries to maximize the surgery time in each block. So, some constraints limiting the utilization of the blocks are

necessaries in order to prevent over-utilization. Before giving these constraints, some assumptions and notations are stated.

Let Dt be the delay with respect to the starting time of a block and let Ct be the cleaning time duration of a OR after a surgery is performed, we assume that Dt and Ct are random variables with normal pdf, i.e., $Dt = N(\mu_{Dt}, \sigma_{Dt})$ and $Ct = N(\mu_{Ct}, \sigma_{Ct})$.

Let us assume the following notation:

- $\boldsymbol{\mu}_c$ is a row vector containing the average cleaning time after each surgery in a waiting list is performed. So, $\boldsymbol{\mu}_c(j)$ represent the average cleaning time of the OR after performing the surgery of patient w_j .
- Similarly, $\boldsymbol{\sigma}_c$ is a row vector containing the standard deviation of the OR cleaning time after a surgery in the waiting list is performed.

Let S_{b_i} the set of surgeries scheduled in the time block b_i , then the total duration T_{b_i} of the time block b_i can be computed by eq. (5.16). That is the sum of: i) the delay with respect to the starting time (i.e., Dt), ii) the duration of the surgeries scheduled in the time block b_i , and iii) the cleaning time duration Ct .

$$T_{b_i} = Dt + \sum_{s_k \in S_{b_i}} [d(s_k) + Ct]. \quad (5.16)$$

Because all these variables (Dt , $d(s)$ and Ct) are assumed to follow a normal distribution, the total duration T_{b_i} of the time block b_i also follows a normal distribution $T_{b_i} \sim N(\mu_{T_{b_i}}, \sigma_{T_{b_i}})$. Considering the solution vector \mathbf{S}_i of a time block b_i then,

- $\mu_{T_{di}} = \mu_{Dt} + (\boldsymbol{\mu}_w + \boldsymbol{\mu}_c) \cdot \mathbf{S}_i$
- $\sigma_{T_{di}} = \sqrt{\sigma_{Dt}^2 + (\bar{\boldsymbol{\sigma}}_w^2 + \bar{\boldsymbol{\sigma}}_c^2) \cdot \mathbf{S}_i}^\dagger$

The N-MIQCP includes a set of chance constraints ensuring that the scheduled blocks have a confidence level not exceeding the total time $l(b_i)$ greater than a threshold $0 \leq Cl \leq 1$, i.e.,

$$P(T_{di} \leq l(b_i)) \geq Cl \quad (5.17)$$

[†]In this thesis $\bar{\boldsymbol{x}}^2$ is a vector such that $\bar{\boldsymbol{x}}^2(i) = \boldsymbol{x}(i) \cdot \boldsymbol{x}(i)$

By using statistic concepts, this set of constraints (5.17) is given by the following inequality

$$\frac{l(b_i) - \mu_{T_{di}}}{\sigma_{T_{di}}} \geq V_{Cl}, \quad \forall i = 1, 2, \dots, |\mathcal{B}|, \quad (5.18)$$

where V_{Cl} is the value corresponding to a normal variable ($x \sim N(0, 1)$) with an accumulative probability Cl , i.e., $P(x \leq V_{Cl}) = Cl$.

Developing inequality (5.18),

$$\begin{aligned} \frac{l(b_i) - \mu_{T_{di}}}{\sigma_{T_{di}}} \geq V_{Cl} &\Rightarrow l(b_i) - \mu_{T_{di}} \geq V_{Cl} \cdot \sigma_{T_{di}} \Rightarrow \\ l(b_i) - \mu_{Dt} - (\boldsymbol{\mu}_w + \boldsymbol{\mu}_c) \cdot \mathbf{S}_i &\geq V_{Cl} \cdot \sqrt{\sigma_{Dt}^2 + (\bar{\sigma}_w^2 + \bar{\sigma}_c^2) \cdot \mathbf{S}_i}. \end{aligned}$$

Let us assume: $\mathbf{A} = \boldsymbol{\mu}_w + \boldsymbol{\mu}_c$; $\mathbf{B} = \bar{\sigma}_d^2 + \bar{\sigma}_c^2$; $C = l(b_i) - \mu_{Dt}$.

Therefore if $C - \mathbf{A} \cdot \mathbf{S}_i > 0$ then

$$\begin{aligned} [C - \mathbf{A} \cdot \mathbf{S}_i]^2 &\geq \left[V_{Cl} \cdot \sqrt{\sigma_{Dt}^2 + \mathbf{B} \cdot \mathbf{S}_i} \right]^2 \Rightarrow \\ C^2 + [\mathbf{A} \cdot \mathbf{S}_i]^2 - 2 \cdot C \cdot \mathbf{A} \cdot \mathbf{S}_i &\geq V_{Cl}^2 \cdot \sigma_{Dt}^2 + V_{Cl}^2 \cdot \mathbf{B} \cdot \mathbf{S}_i \Rightarrow \\ [V_{Cl}^2 \cdot \mathbf{B} + 2 \cdot C \cdot \mathbf{A}] \cdot \mathbf{S}_i - [\mathbf{A} \cdot \mathbf{S}_i]^2 &\leq C^2 - V_{Cl}^2 \cdot \sigma_{Dt}^2 \Rightarrow \end{aligned}$$

Let us assume: $\mathbf{K} = V_{Cl}^2 \cdot \mathbf{B} + 2 \cdot C \cdot \mathbf{A}$; $X = C^2 - V_{Cl}^2 \cdot \sigma_{Dt}^2$,

then the previous inequality becomes:

$$\mathbf{K} \cdot \mathbf{S}_i - [\mathbf{A} \cdot \mathbf{S}_i]^2 \leq X.$$

The set of chance probability constraints imposing a minimum confidence level of not exceeding the available time Cl are shown in (5.19).

$$P(T_{di} \leq l(b_i)) \geq Cl \sim \begin{cases} \mathbf{K} \cdot \mathbf{S}_i - [\mathbf{A} \cdot \mathbf{S}_i]^2 \leq X \\ C - \mathbf{A} \cdot \mathbf{S}_i \geq 0, \end{cases} \quad \forall i = 1, 2, \dots, |\mathcal{B}|. \quad (5.19)$$

Note that $C - \mathbf{A} \cdot \mathbf{S}_i > 0$ is a constraint imposing that the average expected duration $\mu_{T_{di}} = \mu_{Dt} + (\boldsymbol{\mu}_w + \boldsymbol{\mu}_c) \cdot \mathbf{S}_i$ of block b_i is lower than the total available time $l(b_i)$:

$$C - \mathbf{A} \cdot \mathbf{S}_i > 0 \Rightarrow l(b_i) - \mu_{Dt} - \mathbf{A} \cdot \mathbf{S}_i > 0 \Rightarrow \mu_{Dt} + (\boldsymbol{\mu}_w + \boldsymbol{\mu}_c) \cdot \mathbf{S}_i < l(b_i)$$

In this way, the possible symmetric solutions obtained due to $[C - \mathbf{A} \cdot \mathbf{S}_i]^2$ are prevented. However, the model can only schedule working days with a confidence level not exceeding total time greater than 50%. In order to impose a confidence level lower than 50%, the constraint $C - \mathbf{A} \cdot \mathbf{S}_i > 0$ should be changed with $C - \mathbf{A} \cdot \mathbf{S}_i < 0$. In this thesis, we consider $Cl \geq 50\%$.

The full N-MIQCP problem is obtained as,

$$\begin{aligned} & \min \sum_{i=1}^{|\mathcal{B}|} [\beta \cdot \mathbf{P}[i, \cdot] \cdot \mathbf{S}_i - \boldsymbol{\mu}_w \cdot \mathbf{S}_i \cdot (|\mathcal{B}| - i + 1)] \\ & \text{Subject to:} \\ & \left\{ \begin{array}{ll} \mathbf{K} \cdot \mathbf{S}_i - [\mathbf{A} \cdot \mathbf{S}_i]^2 & \leq X, \quad \forall i = 1, 2, \dots, |\mathcal{B}| \\ C - \mathbf{A} \cdot \mathbf{S}_i & \geq 0, \quad \forall i = 1, 2, \dots, |\mathcal{B}| \\ \sum_{i=1}^{|\mathcal{B}|} \mathbf{S}_i[j] & \leq 1, \quad \forall j = 1, 2, \dots, |\mathcal{W}| \\ \mathbf{S}_i \in \{0, 1\}^{|\mathcal{W}|}, \alpha_i \in \mathbb{R}, & \forall i = 1, 2, \dots, |\mathcal{B}|. \end{array} \right. \end{aligned} \quad (5.20)$$

Regarding the size of N-MIQCP (5.20), the problem has

- $n \cdot m$ binary variables;
- $n + m$ linear inequality constraints;
- m quadratic inequality constraints.

Even if the number of variables and of constraints is smaller than in MILP (5.11), the computational complexity of N-MIQCP (5.20) is in general higher due to the existence of quadratic constraints. In order to obtain schedulings in a reasonable time, heuristic approaches are needed.

5.3.3 Heuristics approaches

Two heuristic approaches have been discussed for solving the proposed N-MIQCP. The first one is a *Receding Horizon Strategy* (RHS) which obtains sub-optimal solution sequentially (similar to [8]), while the second one is a *Specific Heuristic Algorithm* (SHA) based on list scheduling techniques [39, 52, 4]. Considering historical data from the Orthopedic Department in the HCU a

one-year scheduling simulation is performed in Sec. 5.3.4 using the two heuristic approaches (RHS and SHA). The results show that similar schedulings are obtained with both approaches, however, the computational time is smaller using the SHA. In addition, the RHS needs the use of CPLEX (an expensive commercial solver) for obtaining sub-optimal solutions of the N-MIQCP problem.

Receding Horizon Strategy to solve the N-MIQCP

The proposed N-MIQCP problem has high computational complexity. So in order to schedule a large number of time blocks, it is necessary to do this sequentially using Receding Horizon Strategy (RHS). The RHS was explained and used in Sec. 5.2.4 for solving large instances of the MILP problem. The methodology of the RHS is shown in Fig. 5.1. Mainly, it is based on obtaining the solution for the next k blocks and select the first s for the final scheduling. This process is iterated until all required blocks have been scheduled. In Sec. 5.3.4 different instances of N-MIQCP are solved optimally, concluding that no more than 3-time blocks can be scheduled at once in a reasonable time. So, the RHS is considered with $k = s = 3$ blocks per iteration.

Specific Heuristic Algorithm

To avoid the use of an expensive commercial solver (CPLEX), an SHA is proposed for solving the scheduling problem. By simulations, it is observed that the obtained solutions using the SHA are very similar to the ones obtained using RHS, but with a lower computational time.

The SHA schedules patients from the waiting list to the next $|\mathcal{B}|$ time blocks in a sequential way. It is inspired from list scheduling techniques [39], [52] and [4], where the items to schedule are ordered according to a given priority. In our case, the items are the patients, and they are ordered according to the inclusion date. The algorithm is based on the idea that the first patients on the waiting list should be scheduled in the first blocks.

The proposed SHA is divided into three main parts: i) a previous data analysis; ii) the scheduling of the time blocks; and iii) the re-assignment of time blocks to dates.

In the *data analysis*, first we classify the set of possible surgeries \mathcal{S} in t (design parameter) different subsets $\{S_1, S_2, \dots, S_t\}$ according with their average duration. Then, considering the surgery with the lowest average duration of each subset, for each time block duration we obtain the surgeries combinations that fulfill the chance probability constraint (5.17). In this way, we compose the combination of subsets to evaluate.

The *scheduling of the time blocks* is performed sequentially by running the second part of the algorithm once for each time block to schedule. In this second part, for each combination of subsets previously generated we obtain from the waiting list, the first real scheduling. These real scheduling, if fulfill the chance probability constraint (5.17), are evaluated by a fitness function. The scheduling with the lowest value of the fitness function is selected for the next time block.

Has been observed that the average preference order of patients scheduled in a time block b_i could be greater than in a time block $b_{i'}$ being $i < i'$. In order to avoid this situation, the three step in the SHA is proposed. In this step, the *time blocks scheduled are re-assigned to the available dates* considering the average preference order of the patients and considering also their available time $l(b_i)$.

The algorithm's input data is:

- The set of surgery types performed in the department, i.e., \mathcal{S} , and their average occurrence percentages.
- The number of time blocks to schedule, i.e., $|\mathcal{B}|$.
- The duration of the time blocks, i.e., $l(b_i)$.
- The minimum confidence level of not having overtime denoted as Cl .
- An ordered waiting list of patients $W = \{w_1, \dots, w_{|W|}\}$ where, for each patient w_j , we know the preference order j , the average duration of its surgery $\mu_w(j)$ and the standard deviation of its surgery $\sigma_w(j)$
- t and β are two input parameters of the algorithm. The number of subsets in which the set of surgeries S is going to be divided is given by t while β is a parameter in the fitness function.

The SHA is composed by 8 steps divided in the 3 mentioned main parts:

A previous data analysis: composed from the first three steps which are running once at the beginning of the scheduling.

Step 1: Classify the surgeries of \mathcal{S} in t disjoint subsets such that $S = \bigcup_{k=1}^t S_k$ and the average duration of all surgeries in S_{k1} are less than the average duration of surgeries in S_{k2} if $k1 \leq k2$, i.e., $\mu(s_a \in S_{k1}) \leq \mu(s_b \in S_{k2})$. Furthermore, the partition such that the expected number of surgeries belonging to each subset S_k in a real waiting list is same.

Step 2: Obtain the set of possible scheduling types (SPS), each type being defined by the subset to which the surgeries belong. For example, a possible scheduling type (st_p) could be $\{k1, k1, k2\}$ and it is composed by two surgeries belonging to the subset S_{k1} and another one belonging to the subset S_{k2} . The set of all st combinations is composing the set SPS , where all elements of the set SPS satisfies (5.17), the chance constraint of not having overtime. This constraint is evaluated considering the surgery with the lowest average duration from the sets S_k .

Step 3: Classify the patients on the waiting list. To each patient $w_i \in W$ we assign a certain type through $ty(w_i) = k_{\in\{1,2,\dots,t\}}$ according with the subset (S_1, S_2, \dots, S_t) to which the surgery belongs.

Scheduling of a block: composed by four steps (4, 5, 6 and 7) which are run sequentially $|\mathcal{B}|$ times, one for each time block to schedule.

Step 4: For each scheduling type ($st_p \in SPS$), obtain a set of real scheduling (SRS_p). Given a scheduling type, a real scheduling is composed by patients from the waiting list who have the same types of surgeries. For example, considering the scheduling type $st_p = \{k1, k1, k2\}$ a real scheduling belonging to SRS_p would be composed by two patients w' and w'' with $ty(w') = ty(w'') = k1$ and other patient w''' with $ty(w''') = k2$). As first patients in the waiting list should be scheduled first, the first real scheduling of type st_p founded in the waiting list is added to their corresponding SRS_p . Moreover, if there are other real scheduling of type st_p ending with the same patient preference order than the first real scheduling, they are also added in SRS_p .

Step 5: For each set of real scheduling (SRS_p) evaluate their elements and select the best real scheduling. First, the real scheduling that do not fulfill the

chance constraint (5.17) are removed from SRS_p . After this, for each real scheduling R_i in SRS_p , the expected occupation rate Or_i and the average preference order Apo_i are computed. According with these values, each real scheduling i of SRS_p is evaluated by the next fitness function (H):

$$H(i) = (Apo_i - Min_{Apo}) * \beta + (Max_{Or} - Or_i), \quad (5.21)$$

where Min_{Apo} is the minimum Apo value between all real scheduling in SRS_p and Max_{Or} is the maximum Or value between all real scheduling of SRS_p . The real scheduling with the lower fitnesses value of each SRS_p is selected.

Step 6: Between the previously selected scheduling of each SRS_p , the final scheduling decision is chosen. Using again the fitness function (5.21) the previously chosen schedulings (Step 5) are evaluated. The scheduling with lower fitness function is assigned to the next block.

Step 7: Remove scheduled patients. The patients scheduled in Step 6 are removed from the waiting list.

Re-assignment of time blocks to dates: composed by last step (8) is running once at the end.

Step 8: Sorting of scheduling. The scheduling obtained are sorted depending on their Average Preference Order Apo_i and considering the available time of the blocks ($l(b_i)$). This means that two schedulings can be interchanged if the duration of their blocks is the same.

The parameter t in *Step 1* fix the number of surgery types. On the other hand, the fitness function (5.21) is composed of two terms. The first one is related with the objective of respecting the preference order of the patients while the second one is related with the objective of maximizing the occupation rate of the OR working days. These two terms are balanced by the value of the parameter β . After some simulations, it has been observed that a value of $t = 3$ and $\beta = 2.6$ are appropriate in the hospital department. Note that we are assuming that all time blocks have the same daily working time $l(b_i)$. In other cases, step 2 should be executed one time for each different time block duration $l(b)$. Consequently, in Steps 4 and 5, the SPS_p corresponding with the duration $l(b_i)$ of the current time block b_i should be used.

5.3.4 Simulation results

In this subsection, the scheduling obtained by solving the N-MIQCP (5.20) are analyzed and compared. For that, considering realistic data from the Orthopedic Surgery Department (OSD) of the “Lozano Blesa” Hospital (LBH), different instances are generated and subsequently scheduled. From these results the computational efficiency, as well as other quality indicators (occupation rate, confidence level, the order of the patients), are compared.

First, the computational efficiency of the exact solution method for solving the N-MIQCP (5.20) problem is analyzed. The results show that only small instances with no more than 3 blocks can be solved optimally. Then, the two heuristic approaches (RHS and SHA) proposed in Sec. 5.3.3 are analyzed and compared. The scheduling obtained are really similar, however, the computational time is lower by using the SHA. Moreover, the SHA does not require an expensive commercial solver.

In order to compare the scheduling obtained considering the SHA with other approaches in bibliography, a one-year scheduling simulation has been proposed. The results conclude that our SHA respects much more the preference order of the patients. Finally, the scheduling obtained by imposing different minimum confidence level in the SHA are compared.

All simulations in this subsection has been performed by using Matlab in a computer with an Intel Core i3 and 4 GB of memory. Moreover, the IBM ILOG CPLEX optimization Studio has been used for the simulation that requires to solve optimization problems.

Exact solution method for the N-MIQCP problem

Let us analyze the computational time to solve optimality different instance of N-MIQCP (5.20) problem. Waiting list composed by $|\mathcal{W}| = 100$ patients are considered. This size of the lists is realistic in Spanish hospitals. Moreover, the time blocks to scheduled are assumed to have a duration of $l(b_i) = 6.5$ [hours]. It could be a morning time block from 8:30 to 15:00. Furthermore, a value of parameter $\beta = 4$ has been considered. It is appropriate from a medical point of view. Simulations with different values of minimum confidence level Cl and different number of blocks to schedule $|\mathcal{B}|$ has been performed.

Notice that the computational time using N-MIQCP problem increase exponentially with the number of time blocks to schedule $|\mathcal{B}|$. Particularly, it has been observed that the optimal solution of some instances with $|\mathcal{W}| = 100$ patients and $|\mathcal{B}| = 4$ time blocks to scheduled cannot be obtained, being the computer out of memory after 6 hours.

However, it always possible to solve instances of N-MIQCP problem with $|\mathcal{B}| = 3$ time blocks to scheduled. Tab. 5.10 shows the average computational time and the average occupation rate. For each minimum confidence level, 200 replication has been performed.

Table 5.10: Comparing exact solution method for different instance of N-MIQCP problem with $|\mathcal{W}| = 100$ patients and $|\mathcal{B}| = 3$ time blocks to scheduled

<i>Input</i>	<i>Output</i>	
Min. <i>Cl</i> [%]	Avg. Ocu. rate [%]	Avg. Time [s]
68	78.9	5.1
70	78.3	7.6
72.5	77.6	9.3
75	76.7	13.2
78	75.57	20
80	74.5	24.03

The result shows that reasonable average computational time (between 5 and 24 [s]) are required to solve optimally instance of N-MIQCP problem with $|\mathcal{W}| = 100$ patients and $|\mathcal{B}| = 3$ time blocks to scheduled. It can be observed that increasing the minimum confidence level, the average computational time also increase. It could be because against greater is the minimum confidence level, more are the solutions that become not feasible due to the quadratic constraints. Moreover, as it was expected, the average occupation rate increase when the minimum confidence level decrease.

Comparing RHS and SHA approaches

In the following, the proposed heuristic approaches (RHS and SHA) for solving large instances of the N-MIQCP problem are compared. For that, a *discrete*

event simulation model of the scheduling is implemented. Particularly, a one year (52 weeks) scheduling simulation for a surgical team is considered. It is assumed that the mentioned team uses 3-time blocks per week, so 156-time blocks are scheduled. Moreover, the duration $l(b_i)$ of each one of these time blocks is assumed to be 6.5 hours. The initial waiting list is composed of $|\mathcal{W}| = 100$ patients. Moreover, the list is updated each week assuming the arrival of new patients who are added at the end. The new patients needing surgery are assumed to arrive according to a Poisson distribution with a mean of 9 per week. During the last two years, considering historical data in the OSD of the LBH, the occurrence of each surgery type and their durations have been computed. These values are considered in the simulation for generating the type of surgery of the (arrival) patients.

Let x be the current simulation week, the steps in the simulation algorithm of the scheduling process are described as follow:

Step 1. Generate the initial waiting list and initialize the current simulation week. An initial waiting list composed of 100 patients is generated randomly using realistic data of the studied department. Moreover the current week “ x ” is initialized to $x = 1$

Step 2. Scheduling the time blocks. Surgeries from the current waiting list are assigned to time blocks booking in the week “ $x+2$ ”.

Step 3. Generate a new set of arrival patients. A number a of new arriving surgeries is generated based on Poisson distribution with a fixed arrival rate considering realistic data from the historical data.

Step 4. Process all scheduled blocks for the current week. We process 2000 replications of each block available in the current week obtaining different average metrics (overtime, utilization, confidence level).

Step 5. Update the waiting list. The patients scheduled in the current week “ x ” (they should be surgically operated week “ $x+2$ ”) are removed from the waiting list. Moreover, the new arrival patients (Step 3) are included at the end of the waiting list.

Step 6. Increment the current week (next week) and stop the simulation if the current week exceeds the end week of the simulation, otherwise proceed to Step 2.

Some events as changes in surgery dates, cancellation of surgeries or the addition of emergency/urgent surgeries are not considered in our simulation

model because are difficult to predict and they are managed in real time based on expert opinions.

Considering different values of minimum confidence level (70%,75%,80%), 200 replications of one-year scheduling have been performed using the RHS and SHA approaches. For the RHS $k = 3$ time blocks are scheduled in each iteration and then, the $s = 3$ time blocks are selected for the final scheduling. Furthermore, a value of parameter $\beta = 4$ has been considered. In the SHA the parameter t is fixed to 3 while the a value of $\beta = 1.2$ is fixed. These values have been approved from a medical point of view.

The average results of the scheduling have been compared from different points of view: utilization efficiency, the confidence of not exceeding the total time, order of the patients and computation efficiency.

Table 5.11: Comparing one year scheduling using N-MIQCP approaches: RHS vs HSA.

Min. Cl [%]	Approach	Utilization		Confidence	Order	Time
		Occu.[%]	# Treated	Avg. Confi.	Ω	Avg.[s]
70	RHS	78.3	439	77.3	406.1	394
	SHA	77.7	437	78.6	318	57
75	RHS	76.7	430	81.5	343.1	685
	SHA	76.1	428	82.7	336	57.4
80	RHS	74.5	415	86.3	313.2	1250
	SHA	73.9	414	86.9	353	58.2

Tab. 5.11 shows the results obtained. The first column imposes the minimum confidence level, this is an input parameter of the problem. The second column indicates the heuristic approach used (RHS or SHA). The third and fourth columns are related to the utilization: occupation rate and the number of treated patients respectively. The fifth column shows the confidence level of not exceeding the available time $l(b_i)$. The sixth column is the parameter Ω related to the order of the patients. Finally, the seventh column shows the computational time to schedule one year (156-time blocks).

The results show that similar utilization efficiency of the ORs is achieved with both, the RHS and the SHA approach. Maybe, a little better occupation rate is obtained and a few more patients are treated using the RHS. However,

the confidence level is better using the heuristic approach and the scheduling obtained is respecting more the order of the list. Finally, it can be checked that the computational time is much lower using the SHA than using the RHS. Moreover, the same computational time is used for performing one-year scheduling using the SHA independently of the minimum confidence level (time ~ 58 [s]). However, the computational time using the RHS increases (394, 685 and 1250 [s]) when the minimum confidence level increases (70, 75, 80 [%]).

Comparing the SHA with other chance-constrained approaches

Using the one-year simulation model described in the previous subsection, the proposed SHA approach is compared with the approaches proposed in [62, 34]. These approaches need a base scheduling of the blocks to obtain the final assignment. The *first-fit probabilistic rule* (Π_{FFP}) under the probabilistic constraints (5.17) is used to obtain this base scheduling. By using Π_{FFP} sequentially each surgery is assigned to the first available block for which the probabilistic capacity constraint (5.17) is satisfied after the assignment.

Considering a minimum confidence level of not exceeding the available time of $Cl = 70\%$, 200 replications of one-year scheduling have been performed for each one of the approaches. Tab. 5.12 shows a comparison between the scheduling obtained by using: (1) the Π_{FFP} rule (commonly used in hospitals), (2) the batch scheduling approach in [62], (3) the constructive algorithm proposed in [34] and (4) our SHA approach. Particularly, column 2 shows the average annual confidence level of not exceeding the available time. In column 3 the total overtime [min/year] is given. Column 4 indicates the average occupation rate of the time blocks [%] while column 5 shows the number of treated patients per year. Finally, column 6 gives the value of the parameter (Ω).

The 3 approaches analyzed in Tab. 5.12 improve the occupation rate of the time blocks with respect to the obtained by using the Π_{FFP} scheduling rule. However, the improvement in the occupation rate of the time block implies a decreasing in the confidence level. For example, the Batch Scheduling approach [62] achieves the highest occupation rate (79.06 %) and the highest number of treated patients (447.78), and consequently, the lower confidence level (75%) and the highest total overtime (1183[min]) is obtained. Taking into account the pairs of values occupation rate and confidence level, the Batch Scheduling

Table 5.12: Comparison of one year scheduling using the SHA and other chain-constrained approaches ($Cl = 70\%$)

Approach	Conf. [%]	Overtime (Year) [min]	Occu. [%]	# Treated (Year)	Ω
Π_{FFP} rule (commonly used)	81.98	806.41	76.12	429.9	1935.9
Constructive Alg. [34]	80.43	922	76.69	432.02	2840
Batch Scheduling [62]	75	1183	79.06	447.78	3993.1
SHA (here proposed)	77.31	1059	78.28	438.3	395.4

approach obtains the better solution with 1) the highest occupation rate and 2) a confidence level within the allowed.

Our SHA approach obtains a little worse occupation rate (78.28 %) than the Batch Scheduling (79.06 %). However, considering the order of the patients by the value of parameter Ω , it can be checked that our SHA approach obtains the best scheduling ($\Omega = 395$). Doctors in the studied hospital department consider that the scheduling obtained using the other approaches are not suitable from a medical point of view because of the great disorder of the patients.

Analysis of different confidence level in the SHA

In order to set an appropriate minimum confidence level Cl using the SHA, one-year scheduling simulations imposing different values of Cl has been performed. Tab. 5.13 shows the results obtained. It can be seen that imposing a little lower Cl in the SHA than in the Batch scheduling approach, similar occupation rates and total overtime are obtained. For example, using Batch Scheduling imposing a minimum confidence level of $Cl = 70$ (Tab. 5.12) and using the RHS approach with $Cl = 67.7$ (Tab. 5.13) the average occupation rates are 79.06[%] and 79.08[%] respectively. Moreover, the total overtime is also very similar: 1183[min] using Batch Scheduling and 1190[min] considering the SHA. However, considering the order of the patients, the scheduling is much more

ordered using the SHA approach ($\Omega = 386.64$) than using the Batch Scheduling approach ($\Omega = 3993.1$).

Table 5.13: Analysis of the scheduling obtained by using the SHA approach with different Cl [%]

Cl [%]	Conf. [%]	Overtime (Year) [min]	Occu. [%]	# Treated (Year)	Ω
51	59.49	2425.1	83.9	469.64	446.58
55	63.14	2093.2	82.80	463.92	411.78
60	67.75	1712.0	81.4	455.56	426.52
65	72.99	1340.8	79.75	446.56	410.44
67.7	75.18	1190.07	79.08	441.42	386.64
70	77.31	1059	78.28	438.3	395.38
75	81.57	805.86	76.72	428.08	380.62
80	86.1	562.33	74.76	418.962	378.82
85	89.82	376.84	72.89	408.14	366.42

5.4 Discussion

The elective surgery scheduling problem under block booking in a Hospital surgery Department has been considered in chapter.

First, three different mathematical models that try to obtain a given occupation rate of the time blocks have been proposed. However, the computational times necessary to solve them optimally are very high and only small instances can be solved in a reasonable time. In order to solve larger instances, we have tested and compared three heuristic approximation methods (one for each model).

- The first one solves a *Mixed Integer Linear Programming* (MILP) model iteratively by using *Receding Horizon Strategy* (RHS).
- The second one considers the heuristic *Step Descent Multiplier Adjustment Method* (SDMAM) for the *Generalized Assignment Problem* (GAP).
- Finally, the last one uses the meta-heuristic *Genetic Algorithm* (GA) for solving the *Quadratic Assignment Problem* (QAP).

The results according to the computational time show that using the RHS and the heuristic SDMAM it is possible to plan 22-time blocks in a reasonable time: 15.41 and 19,32 [s] respectively. However, the meta-heuristic GA spends 7629 [s] in the scheduling of 20 OR workings days.

Regarding the quality of the solutions, two criteria have been taken into account: the occupation rate and the preference order of the patients. Using RHS and the heuristics SDMAM very similar and good occupation rates have been obtained: the average values are not more than 2 points of percentage away from the target values and the standard deviations are lower than 6. The meta-heuristic GA obtain very bad solutions according to the occupation rate: the average values are up to 14.4 points away from the target value and standard deviations are up to 23.3. Taking into account the preference order of the patients, the RHS approach solving the MILP gets the most ordered scheduling. A little more disordered, although acceptable from the medical point of view, are the schedules obtained by using the heuristic SDMAM for the GAP. Again, the meta-heuristic GA obtains a very bad solution with a great disorder of the patients, being unacceptable from the medical point of view.

However, the proposed MILP could be not robust enough because it uses only the average duration of the surgeries. In order to overcome this problem a *New-Mixed Integer Quadratic Constraint Programming* (N-MIQCP) problem is proposed. The N-MIQCP problem considers that the different durations in a block (surgery time, cleaning time, etc) follow a normal distribution. This assumption allow us to introduce some chance constraints that impose a *Minimum Confidence Level* (Cl) not exceeding the available time.

The N-MIQCP problem is computationally very complex being only possible to solve small instances of 3 blocks in a reasonable time. For this reason, two heuristics methods have been proposed and compared. The better schedulings are obtained by using the *Specific Heuristic Algorithm* (SHA). Considering the *Orthopedic Surgery Department* (OSD) in the “Lozano Blesa” Hospital, a one-year scheduling simulation has been used to compare the SHA with other approaches in bibliography. The results show that similar occupation rates and similar confidence levels are obtained, however, our approach respect much more the order of the patients in the waiting list.

Chapter 6

A three step approach for surgery scheduling of elective and urgent patients

Summary

This chapter considers the problem of surgery planning with elective and urgent patients. The proposed solution follows three steps: 1) *scheduling of elective patients*: given the ordered waiting lists of elective patients, schedule them for a target *Elective Surgery Time* (EST) in the ORs; 2) *scheduling of urgent patients*: one day in advance, schedule the urgent patients in the remaining time after step 1; and 3) *sequencing of the surgeries*: the elective and urgent patients scheduled in each OR are sequenced. Using real data from the OSD of the LBH in Zaragoza, the scheduling obtained by the three steps approach are analyzed by simulations. The influence of different EST, i.e., different time reservation policies of the ORs are analyzed in terms of performance and quality of the surgical service. This chapter is based on [22].

6.1 Introduction

Two major patient classes are considered in the literature on the OR planning and scheduling [10], namely elective and non-elective patients. *Elective patients* class represents patients for whom the surgery can be planned well in advance while *non-elective patients* represent patients for whom surgery is unexpected and hence needs to be performed urgently. When considering non-elective patients, a distinction can be made between *urgent* and *emergent* surgery based on the responsiveness to the patients arrival (i.e., the waiting time until the start of the surgery). Whereas the surgery of emergent patients (emergencies) has to be performed as soon as possible, urgent patients (urgencies) refer to non-elective patients that are sufficiently stable so that their surgery can possibly be postponed for a short period (48 hours).

In this chapter, the surgery scheduling of both elective and urgent patients in a surgical department is considered. Moreover, after the scheduling, they are sequenced. The idea is to minimize the maximum time that a new potential emergent patient who just arrives at the service during the working day should wait. It is assumed that the department is composed by medical doctors divided in *surgeon groups*. Moreover, each surgeon group has its own waiting list of elective patients that must be operated by them. In contrast, urgent patients can be operated by any surgeon group. It is also assumed that during a working day each OR is used by one unique surgeon group for a specific duration of time given by the *Daily Working Time* (DWT). Furthermore, the *Daily Surgery Time* (DST) OR is defined as the part of the DWT in which a surgery (elective or urgent) is being performed. Similarly, the *Elective Surgery Time* (EST) OR is defined as the time in which the OR is used for elective surgeries during a day. If the pre-allocated EST is too high, there might not be enough time for urgent patients and consequently, some elective patients should be canceled resulting in a poor quality service. On the other hand, a low pre-allocated EST will result in a waste of time if there are not enough urgent patients. This scenario decreases the utilization rate of the ORs and consequently the efficiency of the service.

Assuming that the daily assignment of the surgeon groups to the ORs are known in advance, in classical scheduling of elective surgeries, the medium term tactical approach *master surgery schedule* would be known. However,

in this chapter, we propose mixed scheduling of elective and urgent patients, where it is necessary to fix the EST of the OR. That is, the time allocated for performing elective surgeries in each OR during a working day. The remaining time will be reserved for surgeries of urgent patients and also for OR cleaning.

A three steps approach for the combined scheduling of elective and urgent patients is proposed in this chapter. In the first step, by solving a *Mixed Integer Linear Programming* (MILP) problem, the elective patients in the waiting lists of the surgeon teams are scheduled in the next OR days. This OR scheduling tries to obtain a target EST in the OR respecting as much as possible the order of the patients in the waiting lists. Second and third steps are performed the day before. In the second one, by using a *Mixed Integer Quadratic Programming* (MIQP) problem the urgent patients are scheduled in the unoccupied times of the available ORs. These patients are scheduled to reach a target DST in each OR and ensuring that an urgent patient does not wait for more than 48 [hours]. The DST is the time allocated for the scheduling of elective and urgent patients in one OR. This time should be lower than the DWT because it is necessary to leave some time for the OR cleaning after each surgery is performed. Finally, in the third step, using another MILP problem, the elective and urgent patients scheduled in each OR are sequenced. The sequencing is performed in such a way that the expected completion times of the surgeries (the time when surgery should finish) are distributed as uniformly as possible during the working day. In this way, the maximum time that an emergent patient should wait to be operated is minimized.

From a strategical point of view, here, we examine whether it is preferred to reserve a dedicated operating room or to reserve some time in all elective operating rooms for urgent patients in order to improve the efficiency of the ORs without compromising the service quality of elective patients.

Different scheduling strategies based on the assignment of different EST to each OR can be used. For example, assuming three available ORs during each working day, a possible strategy could be to reserve two ORs for elective surgeries and the third one to be used only for urgent surgeries. Another possible strategy could be to reserve one OR to the elective patients and the other two ORs could be used for both elective and urgent surgeries. Finally, another possibility is to use all three ORs for elective and urgent surgeries with different EST. Moreover, for each strategy, a different target DST could

be fixed, DST that is mainly related to the utilization of the ORs.

In order to fix a good strategy for the OSD of the LBH of Zaragoza, a realistic simulation methodology is proposed. The scheduling obtained under different conditions using the three steps approach are analyzed and compared from a surgical performance point of view, as well as, from a quality of the service point of view.

Some works combine planning and scheduling problems of elective patients with the urgent ones by using stochastic models (see, for example, [42, 43]). In [7] the balance between maximizing the utilization of one OR, avoiding too many overruns and ensuring a reasonable quality of care in a typical hospital in the United Kingdom was explored by simulation. These simulations examine a policy of including elective patients within one urgent OR session of 7 hours. However, in our chapter, the simultaneous use of ORs is considered (3 in our case of study). In this way, different strategies based on the assignment of different EST to each OR can be explored by simulation. The contributions of this chapter with respect to the previous results are: (1) the application of a three-step approach to the surgery planning problem in the studied hospital; (2) a time simulation methodology for the surgical activity in the three ORs of the studied hospital; and (3) comparison, analysis and synthesis of the simulation result using different strategies for the reservation of the ORs with real data from the hospital.

The chapter is organized as follows. Sec. 6.2 sets the problem. Sec. 6.3 present the proposed mathematical programming problems used in the 3 Steps approach. In Sec. 6.4 the methodology used in the simulations performed is given. Using real data, in Sec. 6.5 some simulations results are analyzed and compared. Finally, in Sec. 6.6, we provide some conclusions.

6.2 Problem Statement

In this section, the terminology and notation are fixed and the problem is introduced formally.

Surgery duration of elective patients

Let $\mathcal{S} = \{s_1, s_2, \dots, s_{|\mathcal{S}|}\}$ be the set of all elective surgeries that can be performed in a hospital department. Moreover, let $d : \mathcal{S} \rightarrow \mathbb{R}_{>0}$ be the *duration*

function: $d(s_i)$ is the duration of the surgery s_i (the time from the moment when the patient enters in the OR until she/he leaves the OR).

For each elective surgery $s_i \in \mathcal{S}$, let us assume that its duration $d(s_i)$ is a normally distributed random variable (pdf) $d(s_i) = N(\mu_{d(s_i)}, \sigma_{d(s_i)})$, where $\mu_{d(s_i)}$ is the mean and $\sigma_{d(s_i)}$ is the standard deviation. The mean and the standard deviation are computed by using historical data from the hospital (for our case study we use the data of the last two years).

Surgeon groups and waiting lists

Let $\mathcal{T} = \{t_1, t_2, \dots, t_{|\mathcal{T}|}\}$ be the set of all surgeon teams in a hospital department. Moreover, let $\mathcal{W}^k = \{w_1^k, w_2^k, \dots, w_{|\mathcal{W}^k|}^k\}$ be the ordered waiting list of patients belonging to the surgeon team t_k such that if $w_j^k \in \mathcal{W}^k$, j is the order number of the patient w_j in the waiting list of team t_k . A patient $w_j^k \in \mathcal{W}^k$ should be operated by team t_k . Let $surg : \mathcal{W}^k \rightarrow \mathcal{S}$ be the function that for a given patient $w_j^k \in \mathcal{W}^k$ gives the surgery that should be performed. For example, if the surgery that should be performed on patient w_j^k is s_l , then $surg(w_j^k) = s_l$.

Operating Rooms

Let $\mathcal{O} = \{o_1, o_2, \dots, o_{|\mathcal{O}|}\}$ be the set of ORs available in a hospital department and let *Daily Working Time* (DWT) be the specific duration of time that each OR can be used every day. Let us consider that all $o_i \in \mathcal{O}$ have the same DWT X . Moreover, let $est : \mathcal{O} \rightarrow \mathbb{R}_{>0}$ be the *Elective Surgery Time* (EST) function: $est(o_i)$ is the specific duration of time per day that the OR o_i is used for performing elective surgeries. For sake of clarity, the EST and the DST are defined as a percentage of the DWT.

Urgent patients

Let $\mathcal{U} = \{u_1, u_2, \dots, u_{|\mathcal{U}|}\}$ be the list of urgent patients that should be operated in the following 48 hours. Moreover, let $d : \mathcal{U} \rightarrow \mathbb{R}_{>0}$ be the *urgent duration function*: $d(u_i)$ is the surgery duration of the urgent patient u_i estimated by the doctors. Notice that, unlike the duration of the elective surgeries, which are calculated based on historical data, the duration of the urgent ones are estimated by the doctors. This is due to the great differences in the duration of a same urgent pathology depending on several factors. Finally, let $wt : \mathcal{U} \rightarrow \{0, 1\}$ be the *waiting time function*: $wt(u_i)$ is the number of days that an urgent patient has been waiting for the surgery.

The **combined scheduling and sequencing problem** of elective and

urgent patients considered in this chapter is:

Given,

- a set \mathcal{T} of surgeon groups;
- for each team $t_k \in \mathcal{T}$ a set \mathcal{W}^k containing an ordered waiting list of patients;
- a set \mathcal{S} of possible elective surgeries;
- a set \mathcal{O} of ORs;
- a duration X of the DWT;
- for each OR $o_i \in \mathcal{O}$ a target elective surgery time: $est(o_i)$;
- the master surgery schedule of the department (set the group $t_k \in \mathcal{T}$ that use each OR $o_i \in \mathcal{O}$ during each day).

1. Schedule the available ORs in the next m working days with the objectives:
 - O_1 - Obtain the target EST of each OR: $est(o_i)$;
 - O_2 - Respect the order of the patients in the waiting list \mathcal{W}^i .
2. Assuming the daily arrival of new urgent patients, schedule them in the unoccupied times of the ORs available in the next day with the objectives:
 - O_3 - Obtain the target DST of the ORs;
 - O_4 - Ensuring that an urgent patient does not wait more than 48 hours.
3. Sequence the elective and urgent patients scheduled in each OR with the objectives:
 - O_5 - Minimize the maximum time that an emergency patient (that arrives during a OR working day) should wait to be surgically operated;

- O_6 - The urgent patients are sequenced before than the electives ones.

Example 6.1. *The Orthopedic Department of the "Lozano Blesa" Hospital is composed by 5 surgeon teams ($|T| = 5$). Moreover there are 3 OR ($|\mathcal{O}| = 3$) available from 8:30 to 15:00 ($DWT=390$ [min]) every weekday (Monday-Friday). Each OR is used by one unique surgeon team during a working day. Tab. 6.1 is the Master Surgery Schedule for weekly OR-to-surgeon groups assignment.*

During the weekend (Saturday and Sunday) only one OR is available and it is used to operate urgent patients. So, urgent patients arrived on Thursday and Friday should be operated on Saturday and the ones arrived on Friday and Saturday should be operated on Sunday. However, to simplify the simulations, in this work the effect of the weekend is not considered, i.e., it is assumed that Friday and Monday are consecutive days.

Table 6.1: Master Surgery Schedule in the Orthopedic Department of the "Lozano Blesa" Hospital

	OR 1	OR 2	OR 3
Mon	t_1	t_2	t_3
Tue	t_4	t_5	t_1
Wed	t_2	t_3	t_4
Thu	t_5	t_1	t_2
Fri	t_3	t_4	t_5

Let us assume the previously described Orthopedic department and its master surgery schedule in Tab. 6.1. Moreover, a time horizon of 10 days (2 weeks) it is also assumed. From the master surgery schedule (Tab. 6.1) it is defined the sequence Sq of ORs that each team will use during the mentioned time horizon. For example, during a week, team t_1 uses first OR 1 (on Monday), then OR 3 (on Tuesday) and finally OR 2 (on Thursday). So in a time horizon of two weeks the sequences of ORs used for t_1 is $Sq = o_1 o_3 o_2 o_1 o_3 o_2$.

Assuming a strategy in which the DST is 75% of the DWT and all three ORs are used for elective and urgent patients with the following EST: $est(o_1) = 60\%$,

$est(o_2) = 50\%$ and $est(o_3) = 40\%$. The target EST in minutes of the i^{th} OR in the sequence Sq of t_1 is given by (6.1).

$$Obj_i = X \cdot \frac{est(Sq(i))}{100} \tag{6.1}$$

For example, the second OR in the sequence of t_1 is o_3 ($Sq(2)=o_3$) and it has an EST in minutes of $Obj_2 = 390 \cdot \frac{40}{100} = 156$.

Moreover, the theoretical time in minutes to perform urgent patients in the i^{th} OR of the sequence Sq is given by (6.2)

$$Ur_i = X \cdot \frac{DST - est(Sq(i))}{100} \tag{6.2}$$

For example, the second OR in the sequence of t_1 ($Sq(2)=o_3$) has a theoretical time to perform urgent patients in minutes of $Ur_2 = 390 \cdot \frac{75 - 40}{100} = 136.5$.

Fig. 6.1 shows the time distribution in the o_3 of the example 6.1.

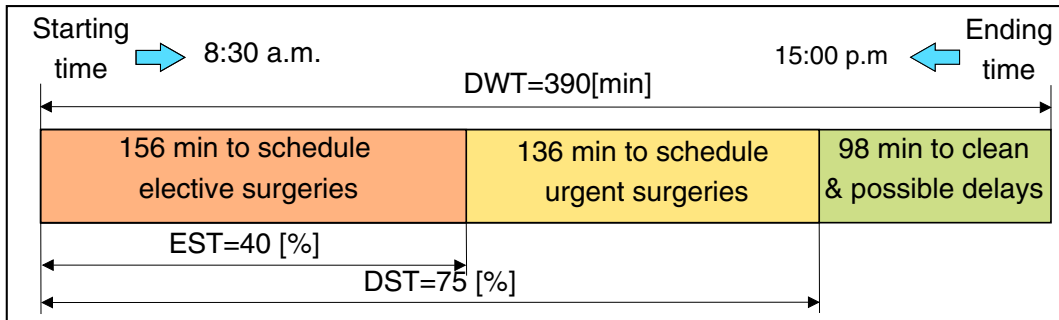


Figure 6.1: Time distribution in an OR with a DWT of 6.5 hours, a DST of 75% and a EST of 40%

6.3 Scheduling and sequencing of patients

In order to solve the combined scheduling and sequencing problem proposed in Sec. 6.2, a three step approach is discussed. Fig. 6.2 shows a flowchart overview of the proposed approach.

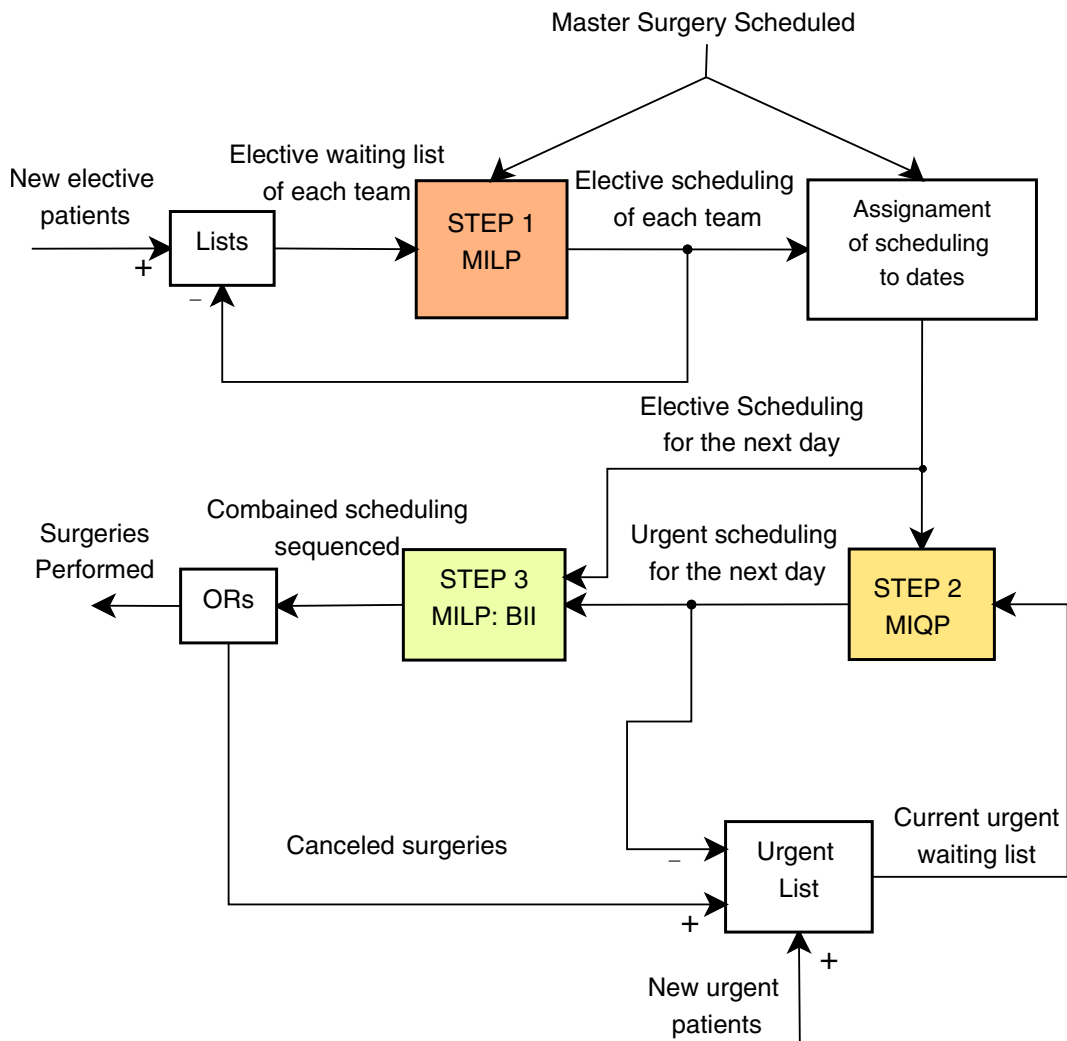


Figure 6.2: Flowchart overview for scheduling and sequencing of urgent and elective patients

6.3.1 Scheduling elective patients using MILP problem

In the first step, the elective patients in the waiting lists are scheduled in the available ORs in the following m working days. Assuming that during a given day, each OR must be used by one unique team, it is possible to divide the scheduling problem in $|\mathcal{T}|$ different scheduling problems, one for each team $t_k \in \mathcal{T}$. So, the next steps should be done independently for each team $t_k \in \mathcal{T}$.

Knowing the master surgery scheduling, it is possible to fix the sequence Sq of ORs that will be used by a team t_k during these m days, the problem to solve is reduced to schedule patients from the waiting list \mathcal{W}^k to the next $|Sq|$ time blocks with the objectives: O_1 - Obtain the target EST of each OR in the sequence and O_2 - Respect as much as possible the order of the patients in the waiting list.

The scheduling of elective patients is obtained by solving the MILP problem (5.11) proposed in section 5.2.2. This problem schedules the elective patients from the waiting list with two contradictory criteria: 1) to minimize the absolute deviation between the target EST and the scheduled EST; and 2) to minimize the sum of the preference order of patients scheduled each day, giving more weights to the first days. Moreover, the MILP problem has a set of constraints to ensure that each patient is scheduled at most once. In this approach, the target EST depends on the OR and it is given by Obj_i in (6.1).

Using the MILP problem, for each team $t_k \in \mathcal{T}$, the following $|Sq|$ time blocks are scheduled with the elective patients from their corresponding waiting list. After this, the $|Sq|$ time block scheduled for each list, are assigned to the corresponding day and to the corresponding OR in the master surgery schedule (Tab. 6.1). In this way, the scheduling of elective patients is obtained.

6.3.2 Scheduling urgent patients

In the second step, the urgent patients in the waiting list \mathcal{U} are scheduled daily in the unoccupied time of the available ORs in the next day. Two objectives must be taken into account in the urgent patients scheduling: O_3 - obtain the target DST and O_4 - an urgent patient should not wait more than 48 hours. In order to solve this problem, a MIQP is proposed. Notice that if the number of urgent patients is high, the target DST will be overcome. In these ORs, during the time simulation, some elective patients may be canceled.

The input data of the MIQP problem is the urgent waiting list of patients \mathcal{U} and the scheduled EST obtained in step 1. Related with the urgent waiting list, the row vector $\boldsymbol{\mu}_u$ of dimension $|\mathcal{U}|$ represents the estimated duration of the urgent surgeries: $\boldsymbol{\mu}_u(j)$ is the estimated duration of the j^{th} urgent patient u_j in the urgent list \mathcal{U} . Regarding the scheduled EST, the vector *Elective occupation* \mathbf{Eo} of dimension $|\mathcal{O}|$ represents (as a percentage respect to the DWT) the scheduled EST in each $o_i \in \mathcal{O}$: $\mathbf{Eo}(i)$ is the percentage of the DWT that the scheduled elective patient should spend in o_i .

In order to not exceed the total DWT but at the same time obtain a good performance of the OR, a target DST is proposed for all $o_i \in \mathcal{O}$. Let us assume that the DWT is denoted by X (in minutes) then,

$$Urg_i = X \cdot \frac{DST - \mathbf{Eo}(i)}{100} \quad (6.3)$$

is the target time in minutes of the OR o_i for urgent patients. A variable γ_i is defined as the difference (in minutes) between the total scheduled time of urgent patients in o_i and the time available for urgent patients Urg_i . This can be written as,

$$\gamma_i = \boldsymbol{\mu}_u \cdot \mathbf{V}_i - Urg_i \quad \forall i = 1, 2, \dots, |\mathcal{O}|, \quad (6.4)$$

where \mathbf{V}_i is the binary vector defining the urgent surgeries scheduled in o_i . There are $|\mathcal{O}|$ binary vectors $\mathbf{V}_1, \mathbf{V}_2, \dots, \mathbf{V}_{|\mathcal{O}|}$, each vector having dimension equal to the number of urgent patients in the waiting list $|\mathcal{U}|$, i.e., $\mathbf{V}_i \in \{0, 1\}^{|\mathcal{U}|}$. If $\mathbf{V}_i[j] = 1$ then surgery of patient u_j should be performed in the OR $i \leq |\mathcal{O}|$.

Variable γ_i in the cost function (6.5) of the proposed MIQP penalize the square difference between the scheduled DST and the target DST in the OR o_i .

$$\min \sum_{i=1}^{|\mathcal{O}|} \gamma_i^2 \quad (6.5)$$

In this way, after step 1, the unoccupied times available in each $o_i \in \mathcal{O}$ are used for the scheduling of urgent surgeries trying to obtain their DST. Moreover, the urgent patients are assigned to the ORs in such a way that its occupation rates are balanced, i.e., similar deviation from the DST for all $o_i \in \mathcal{O}$ are obtained each day.

A new urgent patient $u_j \in \mathcal{U}$ who has been waiting for the surgery 0 days ($wt(u_j) = 0$) could be scheduled at most once:

$$\sum_{i=1}^{|\mathcal{O}|} \mathbf{V}_i[j] \leq 1, \quad \forall j \text{ s.t. } wt(u_j) = 0. \quad (6.6)$$

However, an urgent patient $u_j \in \mathcal{U}$ who has been waiting for the surgery 1 day ($wt(u_j) = 1$) must be mandatory scheduled:

$$\sum_{i=1}^{|\mathcal{O}|} \mathbf{V}_i[j] = 1, \quad \forall j \text{ s.t. } wt(u_j) = 1. \quad (6.7)$$

Putting together, the following MIQP is obtained:

$$\begin{aligned} & \min \sum_{i=1}^{|\mathcal{O}|} \gamma_i^2 \\ & \text{Subject to:} \\ & \left\{ \begin{array}{ll} \boldsymbol{\mu}_u \cdot \mathbf{V}_i - U r g_i & = \gamma_i, \quad \forall i = 1, 2, \dots, |\mathcal{O}| \\ \sum_{i=1}^{|\mathcal{O}|} \mathbf{V}_i[j] & \leq 1, \quad \forall j \text{ s.t. } wt(u_j) = 0 \\ \sum_{i=1}^{|\mathcal{O}|} \mathbf{V}_i[j] & = 1, \quad \forall j \text{ s.t. } wt(u_j) = 1 \\ \mathbf{V}_i \in \{0, 1\}^{|\mathcal{U}|}, \gamma_i \in \mathbb{R}, & \forall i = 1, 2, \dots, |\mathcal{O}|. \end{array} \right. \quad (6.8) \end{aligned}$$

6.3.3 Sequencing the surgeries

In the third step of the approach, the surgeries scheduled (electives and urgent) to each OR are sequenced in order to minimize the maximum time that an emergency patient, who arrive at the hospital during the working day, should wait to be operated. Remember that emergency patients should be operated as soon as an operating room is free. Moreover, urgent patients should be sequenced first. In this way, if a surgery must be canceled in order to not exceed the total DWT, the canceled surgery should correspond to an elective patient. To sequence the surgeries, a modified version of the *Break In Moments* (BIM) problem proposed in [69] is used. The BIM problem focuses on sequencing surgeries which are assigned to specific ORs. The set of these surgeries is given by set $I = \{1, \dots, M\}$. In this work, we include the subset $E \subset I$ representing

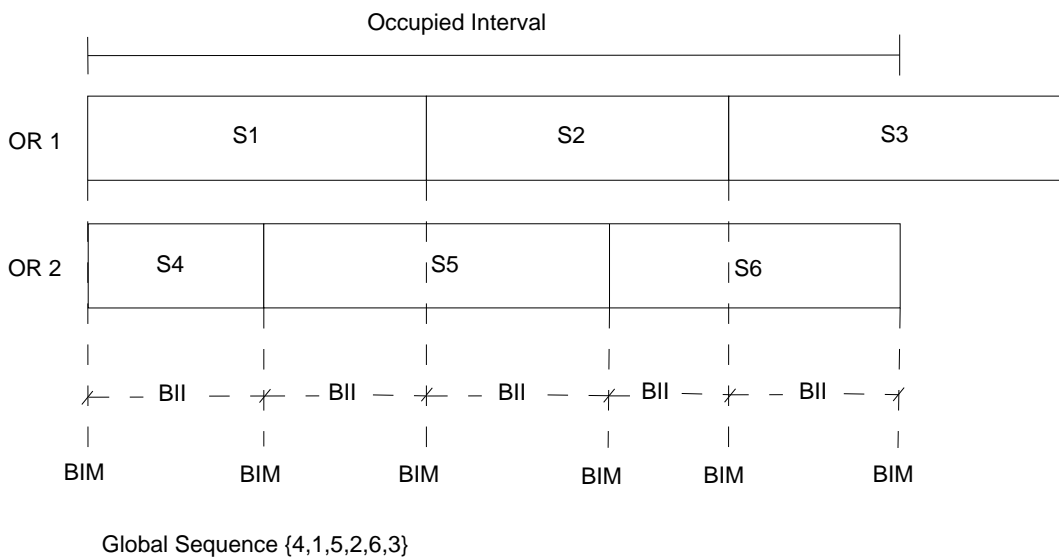


Figure 6.3: Example of the BIM problem terminology showing the occupied interval, the BIMs, the BIIs, and the global sequence.

the elective surgeries and the subset $U \subset I$ containing the urgent surgeries. These subsets allow us to impose by a new constraint that elective surgeries are sequenced after the urgent ones. In the BIM problem, it is assumed that all surgeries have to be scheduled successively with no breaks in between. So, in order to take into account the cleaning time we define the duration of a surgery i as the expected duration of the surgery $i \in I$ plus a cleaning time of 15 [minutes]. In the BIM problem, the interval for which all operating rooms are occupied is called the “occupied interval”. Fig. 6.3 shows an illustrative example in which the surgeries scheduled in two ORs are sequenced.

The start and end time of the occupied interval, as well as all completion times of surgeries within the occupied interval are defined as an event BIM. These events are used to start the emergency surgeries that can arrive at the hospital. Ordering the completion times of the surgeries in a non-decreasing way, it is possible to determine the length of the intervals between two BIMs that is called *Break In Interval* (BII). The timing of the BIMs, and thus the length of the BIIs, depends on the sequence of the surgeries in each OR. In practice, BIMs appear to be distributed unevenly over the occupied interval,

which results in lengthy BIIs that increase the expected waiting time for emergency surgery and therefore, increase the risk of medical complications or even mortality. The objective of the sequencing step is to minimize the expected waiting time for emergency surgery by sequencing surgeries in such a way that the BIMs are spread as evenly as possible over the day. For this, the minimization of the maximum BII per day is proposed.

To solve the BIM problem in an optimal way a Mixed Integer Linear Program is proposed [69]. This problem fixes the sequence of the surgeries (previously scheduled) in each OR. Several ORs are considered simultaneously. Moreover, the patients scheduled are not allowed to be swapped between ORs. These sequences are “local”, however, in order to determine the BIIs and the maximum BII, it is necessary to determine the sequence of the BIMs overall ORs, i.e., the completion times of all surgeries. This sequence is called the ‘global sequence’.

In order to impose that the elective surgeries are sequenced after the urgent ones, the next set of constraint should be added to the original BIM problem:

$$Z_i < Z_k, \quad \forall i \in U, \forall k \in E, \quad \text{with } O(i) = O(k) \quad (6.9)$$

where Z_i are integer variable that represents the position of surgeries $i \in I$ in the global sequence and $O(i)$ denote the OR to which surgery $i \in I$ is assigned.

6.4 Simulation Approach

In this Section, the simulation methodology used for surgical activity in the OSD of the LBH in Zaragoza is explained.

The main objective of these simulations is to check the influence that different EST assigned to each OR, i.e., $est(o_i)$, have in the:

- number of elective scheduled patients (# Sp)
- number of elective treated patients (# Tp)
- number of elective canceled patients (# Cp)
- number of OR in which DWT is exceeded (# Et)

6.4.1 Patients generation and scheduling

In this subsection, the organizational structure and the size of the department are recalled. Furthermore, the simulation time horizon is fixed. Moreover, it is shown how both waiting lists of elective patients and the arrival of the urgent patients per day are generated.

Size of the Department. As in example 6.1, the hospital department has 3 ORs each day from 8:30 to 15:00, so the DWT is defined as $X = 390$ [minutes] in all ORs. The department is composed by 5 medical teams and the master surgery schedule is the same every week, shown in Tab. 6.1.

Generation of elective waiting lists. Using historical data of the performed surgeries, the probability of appearance of each surgery is computed as the total number of patients with the same surgery divided by the total number of surgeries. Based on these probabilities, a large random waiting list of elective patients is generated for each team.

Time horizon. The temporal horizon in these simulations is fixed to 200 days. Moreover, knowing the master surgery scheduling (see Tab. 6.1), the number of OR that each team is going to use in the 200 days is computed. As 3 out of 5 days each team uses an OR, the total number of OR to schedule for each team is $|Sq| = 200 \cdot \frac{3}{5} = 120$.

Using the MILP model, next $|Sq| = 120$ OR are scheduled for each waiting list. Due to the complexity of the problems, the MILP problem is solved iteratively, using the RHS approach.

Once the $|Sq| = 120$ OR has been scheduled for all five waiting lists, each one is assigned to the corresponding day using again the master surgery scheduling.

Generation of urgent patients. Assuming that initially the waiting list of urgent patients is empty, it is updated each day: the patients who has been scheduled the previous day are removed and the new urgent patients who arrive to the hospital are included. The arrival of urgent patients each working day is generated as follow:

1. Based on historical data of the performed urgent surgeries in the studied department, we have analyzed the *urgent daily duration* (Tu). It is defined as the duration of a working day used for urgent surgeries. After

applying the following Johnson Transformation [13]:

$$Z = -1.499 + 2.1028 \cdot \sinh^{-1} \left(\frac{Tu - 63.467}{51.992} \right), \quad (6.10)$$

it is proved that follows a Normal distribution such that $Z \sim N(-0.05092, 1.00603)$. So, in order to generate the total time of a working day for urgent patients, first $Z = N(-0.05092, 1.00603)$ is generated randomly and then, using (6.10) Tu is obtained.

2. The urgent patients duration are generated sequentially until the error e_{j+1} is greater than e_j . The error e_j is defined as the absolute difference between the sum of the expected duration of the first j urgent patients generated and Tu .

$$e_j = \left| Tu - \sum_{i=1}^j d(u_i) \right| \quad (6.11)$$

Based on historical data of the performed urgent surgeries, the duration of an urgent patient $d(u_i)$ has been analyzed. It has been observed that this duration, after applying the Johnson Transformation [13]:

$$Z = -0.0455 + 0.5280 \cdot \sinh^{-1} \left(\frac{d(u_i) - 299.254}{19.759} \right), \quad (6.12)$$

follows a Normal distribution such that $Z \sim N(-0.06289, 0.93346)$. So, in order to generate the expected duration of an urgent patient, first $Z \sim N(-0.06289, 0.93346)$ is generated randomly and then, using (6.12) $d(u_i)$ is obtained.

The urgent patients are scheduled using the MIQP (6.8) and after this, the elective and urgent patients scheduled in each $o_i \in \mathcal{O}$ are sequenced using the BIM problem described in Sec. 6.3.3.

6.4.2 Surgery activity simulation in a OR

Once the elective and urgent surgeries are scheduled and sequenced, a time simulation of the activity surgeries is performed for each OR. The following considerations are taken into account:

Delayed OR opening. Based on historical data, it has been observed that there is a delay Dt with respect to the starting time. This delay follows a normal distribution $Dt = N \sim (10, 12)$. That is considered at the beginning of each surgery day.

Performing urgent surgeries. For each urgent surgery scheduled, a duration of the surgery and the cleaning time are generated. It is assumed that the duration of an urgent surgery u_i follow a normal distribution $Ut_i = N(d(u_i), 10)$ in which the average value is the expected duration $d(u_i)$ and the standard deviation is 10 minutes. The cleaning time Ct after a surgery (elective or urgent) is performed assuming to follow a normal distribution such that $Ct = N(20, 10)$. So the total time after the last urgent patient has been surgically operated is as follow:

$$T_t = Dt + Ut_1 + \sum_{i=2}^k (Ct + Ut_i),$$

where k is the total number of urgent patients scheduled in the OR. Notice that the cleaning time after the last urgent surgery is not included yet.

Performing or canceling elective surgeries. A scheduled elective surgery (s_i) is performed only if it is expected that the DWT is not exceeded. For this, before starting an elective surgery, it is necessary to check if the expected duration of the cleaning time (20 minutes) plus the average duration of the elective surgery ($\mu_{d(s_i)}$) plus the actual total time T_t is lower than the DWT. So for each one of the elective surgery scheduled in the OR:

- if $DWT \geq T_t + 20 + \mu_{d(s_i)}$ the surgery is performed and the total time T_t is updated by adding the cleaning time and the duration of the elective surgery s_i :

$$T_t := T_t + Ct + d(s_i).$$

Notice that Ct and $d(s_i)$ are random variables with normal pdf and consequently, adding these durations to the T_t it is possible that the DWT is exceeded although it was not expected considering their average values.

- if $DWT < T_t + 20 + \mu_{d(s_i)}$ it is expected that performing the surgery s_i the DWT X will be exceeded and the surgery is canceled.

Finally, the number of elective scheduled patients (# Sp), the number of elective treated patients (# Tp), the number of elective canceled patients (# Cp) and the number of ORs in which DWT is exceeded (# Et) are computed. To obtain more robust results, 150 replications are performed for each solution, computing the average values of: # Sp #Tp, #Cp and #Et.

6.5 Numerical Results

In this section, following the methodology explained in Sec. 6.4, different simulation results are showed and compared. They are shown in Tab. 6.2, which is divided in 3 parts for easier analysis and for simple reading. The first column shows the number of corresponding simulations. The next three columns represent the target EST of each OR. The fifth column indicates the sum of the time percentage reserved for elective patients in each OR: $\sum_{i=1}^3 est(o_i)$. The sixth column shows the target DST. Finally, the last four columns indicate the #Sp, #Tp, #Cp and #Et respectively.

Table 6.2: Simulation results of 200 days in the studied department with different $est(o_i)$ and DST

Sim	$est(o_1)$ [%]	$est(o_2)$ [%]	$est(o_3)$ [%]	$\sum_{i=1}^3 est(o_i)$	DST [%]	#Sp	#Tp	#Cp	#Et
1	75	75	0	150	75	1067	975	92	64
2	75	42.5	32.5			1060	960	100	43
3	60	50	40			1064	981	83	43
4	60	55	35			1068	966	102	43
5	60	55	40	155	75	1098	981	117	49
6	60	50	45			1091	982	109	46
7	60	45	40	145		1030	964	66	38
8	60	50	35			1038	967	71	39
9	60	50	40	150	78	1064	973	91	46
10					72	1064	972	92	42

In the first part of Tab. 6.2 (simulation 1 to 4), the influence of different target EST $est(o_i)$ adding together the same elective surgery total time is checked. Moreover, the same target DST= 75% is fixed. In the

first simulation, the ORs o_1 and o_2 are reserved only for elective patients ($est(o_1) = est(o_2) = 75\%$) and the OR o_3 is reserved only for urgent patients: ($est(o_3) = 0\%$). In the second simulation, the strategy is to reserve OR o_1 for elective patients ($est(o_1) = 75\%$) while ORs o_2 and o_3 combine the elective patients ($est(o_2) = 42.5\%$ and $est(o_3) = 32.5\%$) with the urgent ones ($75 - 42.5 = 32.5\%$ and $75 - 32.5 = 42.5\%$). Finally, the last two strategies use the 3 ORs in a combined way scheduling elective and urgent patients as is shown in Tab. 6.2. Because the total time assigned to the elective surgeries is the same for all distribution ($\sum_{i=1}^3 est(o_i) = 150$), as it can be expected, the number of elective patients scheduled ($\#Sp$) is really similar in each simulation (between 1060-1068). The first strategy, with the OR o_3 reserved for urgent patients, has the greatest number of ORs working days in which the DWT is exceeded ($\#Et=64$). It happened because in OR o_3 there are only urgent patients who cannot be canceled. Regarding the number of canceled patients ($\#Cp$), the strategy in the simulation 3 ($est(o_1) = 60\%$, $est(o_2) = 50\%$ and $est(o_3) = 40\%$) that combines the scheduled of elective and urgent patients in the three ORs have the lowest number of canceled patients ($\#Cp=83$), and consequently, the highest number of treated patients ($\#Tp=981$). So, the best option according to this simulations is to combine the surgeries of elective patients and urgent patients in all the three ORs such that $est(o_1) = 60\%$, $est(o_2) = 50\%$ and $est(o_3) = 40\%$.

In the second part of Tab. 6.2 (simulations 5-8), the influence of the total time assigned to elective patients ($\sum_{i=1}^3 est(o_i)$) is studied. In all simulations of the second part, the three ORs are scheduled in a combined way (elective and urgent patients). In the first two simulations of the part 2 ($\#5$ and $\#6$), the time assigned to elective patients is increased ($\sum_{i=1}^3 est(o_i) = 155\%$) while in the two last simulations ($\#7$ and $\#8$) this time is decreased ($\sum_{i=1}^3 est(o_i) = 145\%$).

Analyzing simulations $\#5$ and $\#6$, it can be observed that increasing the time assigned to the elective patients from 150% to 155% the number of scheduled patients $\#Sp$ also increase (from 1064 to 1091 and 1098 respectively). However, this increase is not reflected in the number of treated patients $\#Tp$ (from 981 to 981 and 982 respectively), since the number of canceled surgeries $\#Cp$ is also increased (from 83 to 117 and 109 respectively). Regarding the number of days in which the DWT is exceeded $\#Et$, it is increased (from 43 to

49 and 46 respectively). So, increasing the time reserved for elective patients above 150 % the solutions are worst: the number of treated patients does not increase. Nevertheless, the number of canceled patients and the number of OR working days in which the DWT is exceeded are increased.

According to simulations #7 and #8, it can be seen that decreasing the time assigned to elective patients from 150% to 145%, the number of elective scheduled patients #Sp, elective treated patients #Tp and elective canceled patients #Cp decrease. From the point of view of elective treated patients, the solutions obtained are worse. However taking into account the quality of the service, it could be reflected in the number of days in which the DWT is exceeded (#Et) or in the number of elective patients treated for each one canceled ($\frac{\#Tp}{\#Cp}$), the solutions obtained are better.

Finally, in the two last simulations (#9 and #10) the influence of the target DST is showed. For that, two simulations with the same cumulative target EST than in the simulation 3 but with different target DST are showed. In the ninth simulation, a target DST=78% is considered while in the tenth one a target DST=72% is chosen. In both cases, the results obtained are worse: with DST=78%, a higher probability of exceeding the total time exists, and consequently, a more number of patients are canceled; with DST=72%, although the probability of exceeding the DWT is lower, the ORs are scheduled with a lower total capacity. This means that lower is the urgent capacity scheduled and the urgent list increase. Consequently, when the arrival of urgent patients has a peak, greater is the number of elective patients canceled.

6.6 Discussion

A three steps approach to surgery planning of elective and urgent patients has been proposed in this chapter. The first step uses a MILP. The purpose is to schedule the elective patients with a given *Elective Surgery Time* ($est(o_i)$) in each OR, respecting as much as possible the order of the patients in the waiting list. In the second step, the day before surgeries, the urgent patients are scheduled in the unoccupied time of the ORs by solving MIQP (6.8). This model tries to obtain a target *Daily Surgery Time* (DST) of the OR, but ensuring that the urgent patients do not wait more than 48 hours. Finally, in the third step, by solving a Break In Moments (BIM) problem, the scheduled

(elective and urgent) are sequenced in such way that the maximum time that an emergency patient eventually arrives at the hospital waits is minimized.

The problems have been tested and compared using data derived from the Orthopedic Surgery Department of the “Lozano Blesa” Hospital in Zaragoza, Spain. In order to check the influence of the target $est(o_i)$ in each OR and the target DST realistic simulations have been performed. In these simulations, the performance of the service has been analyzed by the *number of treated elective patients* (#Tp). Moreover, the quality of the service is also analyzed considering the *number of canceled patients* (#Cp) and the number of days in which the *Daily Working Time* (DWT) is exceeded (#Et).

The simulation results show that the strategy of scheduling elective and urgent patients in all ORs is better than the strategy of reserving some ORs exclusively for elective or urgent patients. More patients are scheduled, lower is the number of canceled patients and lower is the number of blocks that exceed the available time. Moreover, for the studied department, it has been observed that increasing the sum of the time assigned for elective patients in each OR ($\sum_{i=1}^3 (DST(o_i))$) above 150, the performance of the service does not increase and the quality decreases, so worse solution are obtained. However, up to around 150, the performance of the service is improved, but the quality is decreased, being necessary to establish a compromise between efficiency and quality. Finally, it has been observed that a target DST around 75% is appropriate while a target DST above 75% implies a high risk of exceeding the DWT. Furthermore, a target DST below 75% implies smaller utilization of the ORs.

A target ESTs of 60%, 50% and 40% (Sim. #3) for the three ORs have been considered as a good option for the Orthopedic department in the “Lozano Blesa” Hospital of Zaragoza.

Chapter 7

Decision Support System and Software tool for surgery scheduling

Summary

Medical doctors should spend their working time taking care of patients. However, they also play the role of decision makers being involved, for example, in scheduling surgeries in ORs. Although some doctors might be doing this for a long time, which gives them experience in such decisions, the efficiency of the operation room can be improved with the help of some technical solutions that are the result of advanced research in planning and scheduling surgeries. Such technical solutions can also decrease the overtime hours and reduce the under/over utilization of some operating rooms. This chapter's focus is to create such a solution that will help medical doctors to spend less time in managing tasks and have a better occupancy rate within the available capacity of the operation room. It will also decrease the stress level and allow medical doctors to offer more of their time to healthcare. First, a *Decision Support System* (DSS) for surgery scheduling is proposed. Then, based on this DSS, a software solution with a friendly interface is developed. Finally, a real case study is discussed. The DSS and the software tool are explained in [15, 16].

7.1 Decision Support System

A DSS for OR scheduling was proposed in [26]. It uses as core a MILP problem that takes into account the preference order of the patients. However, this approach does not consider the variability on the uncertain parameters (surgery duration, cleaning time, etc) and consequently can not manage the risk of exceeding the total time. The DSS proposed in this work uses the SHA (Sec. 5.3.3) that allows doctors to impose a minimum confidence level of not exceeding the total time in a surgical block. In this way, doctors not only can manage the risk but also they can know in advance the probability that each block exceed the available time.

The DSS proposed in [26] allows to perform medium term estimation and short term scheduling as well as manual modifications. The DSS proposed here, in addition to the aforementioned, can also perform the management of medical teams and OR time table. Moreover, it is possible to perform iterative scheduling which is useful if, for example, after obtaining a first scheduling, some patients inform that they can not attend the day of their surgeries. Furthermore, after a surgery is performed, the DSS automatically updates and customizes the surgeries duration depending on the surgeon. Finally, in order to provide a safe environment to work and better security of the medical data, different access level are provided to every user.

For the DSS is assumed that each doctor in the hospital department has his own list of patients and the waiting list of a team is composed by the merged list of doctors belonging to the team. Each team has a coordinator who must schedule the patients from the waiting list of his/her team during the time blocks previously booked by the head of the department. So, the main objective of the DSS is to help medical doctors to perform a rapid efficient and dynamic scheduling of elective patients.

Fig. 7.1 shows the organizational structure of a hospital department composed by 5 surgeon groups and 2 ORs. Each strong color person represents a coordinator and the fade colors ones represent team members. The small squares are each doctors waiting list. The circles are the available ORs. The strong red person is also the head of the department so, he/she books OR for teams. The coordinators (in strong colors) take the days booked for their teams and create the scheduling of the patients from the team waiting list,

represented by the big squares.

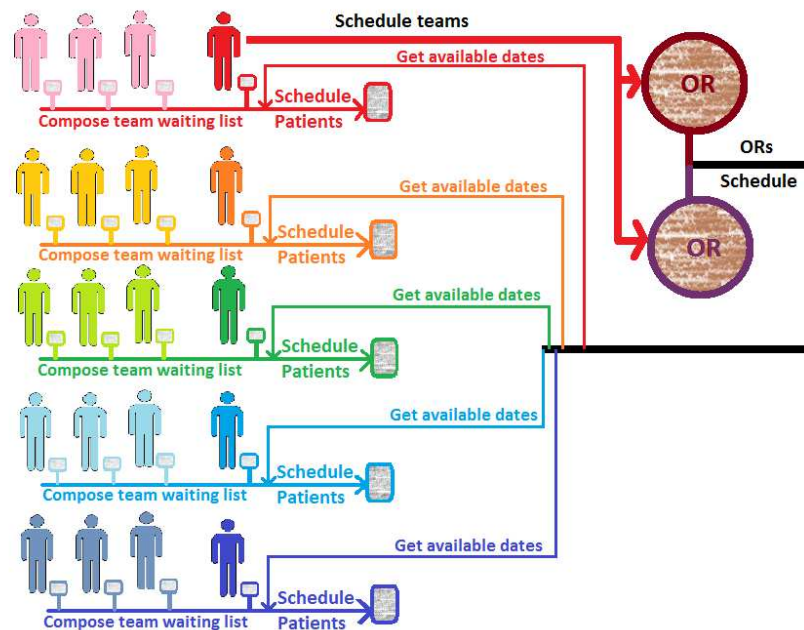


Figure 7.1: Organizational structure in the studied hospital departments

The DSS uses as a core for the scheduling of patients the SHA. But, also other features are included which enable a) manage medical teams and OR time-table b) updating the waiting list c) dynamic planning and d) improving the input data by updating the surgeries duration.

7.1.1 Manage medical team and OR time-table

Normally the medical teams are composed by the same medical doctors. However, sometimes medical doctors could move from one team to another. The head of the department is responsible for updating the medical teams and including this information in the DSS. Moreover, the head of the department should define the ORs time-table in the DSS. The OR, the date, the starting time and the ending time of each time block assigned to each team is the information that must be included in the DSS.

7.1.2 Updating the waiting list

Generally, the new arrival patients are included at the end of the waiting list. However, depending on medical criteria, the surgeon could decide to advance the patient into a higher position in the waiting list. The DSS, automatically orders the waiting list of patients. There are 2 parameters influencing directly the position of the patient in the waiting list.

1. The first parameter is related to the waiting time for surgery. A score S_1 of 10 is given to the patient with the highest number of waiting days, while the newest patient has a score of 0. A proportional score between 10 and 0 is given to other patients. In the calculation of total score (denoted S_T) the score S_1 have a weight of p_1 .
2. The second parameter has to do with the surgery priority. Although non-urgent surgeries are scheduled with the DSS, 3 levels of priority 1, 2 and 3 are considering by doctors depending on the urgency. A corresponding score (S_2) of 0, 5 and 10 is associated respectively. The weight of S_2 in the computation of S_T is p_2 .

Assuming $p_1 + p_2 = 1$, the final score is obtained as follow:

$$S_T = p_1 \cdot S_1 + p_2 \cdot S_2 \quad (7.1)$$

Finally the patients are ordered according to their total score. The patient with the highest total score will be the first, while the patient with the lowest punctuation will be the last one.

7.1.3 Iterative planning

The scheduler of each medical team performs the scheduling for the next $|\mathcal{B}|$ time blocks (this is done by using the SHA). The secretary calls the patients scheduled and ask them for their availability on the scheduled date. The secretary gives back this information to the team scheduler who should schedule again the empty gaps. This process is repeated until the $|\mathcal{B}|$ time blocks are completely scheduled with all patients confirmed.

During the iterative scheduling, if a patient confirms the attendance, she/he is fixed in the corresponding time block. However, in case that a patient

cannot be contacted or he/she cannot be hospitalized. However, in case that a patient cannot be contacted or he/she cannot be hospitalized, then the patient is not considered for the scheduling of the next $|\mathcal{B}|$ time block.

So, in the next iteration, the patients previously confirmed and the patients who cannot be hospitalized are not considered in the waiting list. The SHA schedule new patients in the gaps of the time block uncompleted.

7.1.4 Updating and customizing the averages durations

The average duration and standard deviation of each type of surgery should be computed using historical data from the hospital department. Considering a period of two years, a sufficiently high number of surgeries of each type is obtained and the average durations are representative. However, depending on the skills and experience of the surgeons, significant differences could exist between their average durations. Moreover, in general, for the same medical doctor, a continuous decreasing of the average durations occurs after performing the same surgery several times. So, a dynamic update of these input values is really important.

When a surgery is finished, the surgeon who has performed it and the time spent are introduced in the DSS which registers this information in a database and updates the average duration database.

7.1.5 Overview of the DSS

All the previously explained features are integrated into the DSS whose flowchart is given in Fig. 7.2.

When a new patient arrives at the service, he/she is introduced into the waiting list of a medical doctor. Each surgeon is responsible for including their patients in the system. The DSS recognizes the surgeon (using a personal password) and he/she must enter some personal information of the patient, the pathology and the priority of the surgery. Additionally, the DSS saves the actual date (to compute the waiting time in the list) and the surgeon to whom the patient belongs. When a team scheduler wants to perform scheduling, he/she is recognized by the DSS which, considering the doctors currently belonging to his/her team, automatically composes the ordered waiting list of patients.

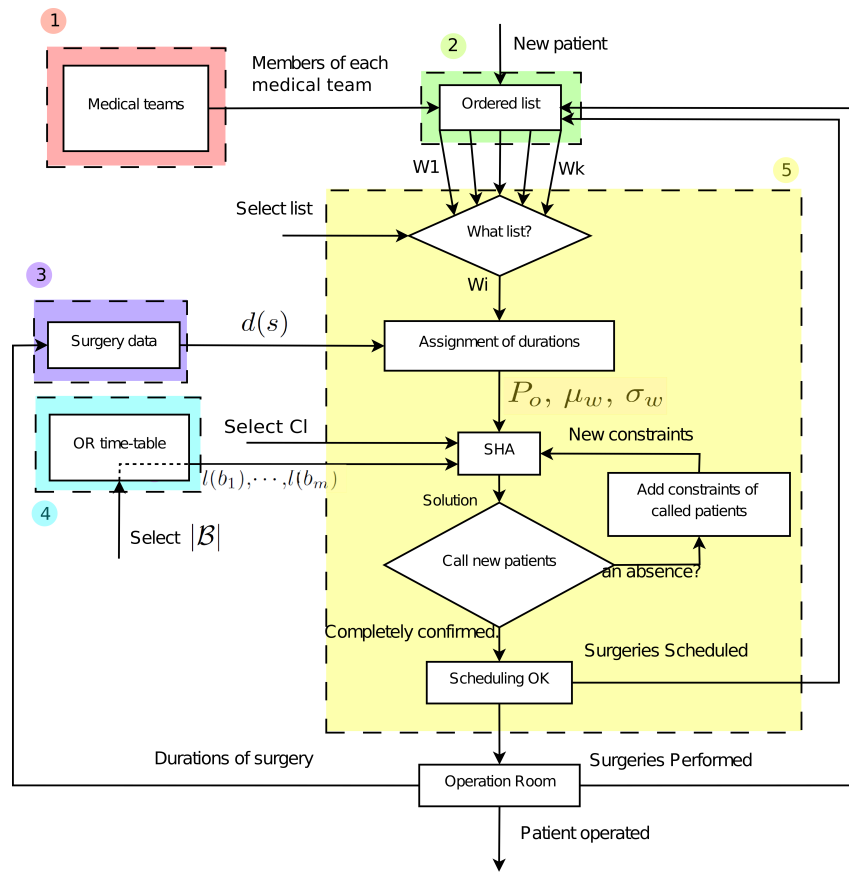


Figure 7.2: Flowchart of the DSS. Discontinuous colored boxes reference the 5 main panels included in the developed software tool: 1) Medical teams; 2) Patients; 3) Surgeries; 4) OR timetable; and 5) Scheduling

The DSS has a database, which is updated every time that a surgery is performed. The pathology and its surgery durations are saved in the database. So considering this information, the tool assigns average theoretical durations and standard deviation to each surgery in the waiting list. In this way, the vectors μ_w and σ_w are generated. On the other hand, the DSS save the OR time-table introduced previously by the head of the department. That is, the time blocks booked for each team as well as their duration defined as $l(b_i)$. The DSS performs an operation scheduling in an iterative way. The input data that the team's manager has to introduce in the DSS to schedule the next time blocks are: i) the minimum confidence level Cl and ii) the number

of time blocks to schedule $|\mathcal{B}|$.

When a patient has been scheduled, his/her state change from “pending” to “scheduled”. Once a specific surgery is performed, the surgeon indicates this situation in the DSS and the corresponding patient is removed from the waiting list. Additionally, the surgeon introduces the operating time in the tool and this new input data is used to update the average duration. If finally a scheduled surgery is not performed, the DSS changes the state of this surgery from “scheduled” to “pending”.

7.2 CIPLAN: a software tool for elective surgery scheduling

In this section, based on the previously described DSS, we propose a software-based solution called CIPLAN which is presented, along with the advantages and features that it brings. The software tool’s focus is to help doctors to spend less time in managing tasks and have a better occupancy rate within the available capacity. This application is made to be used by the medical staff of surgery Departments. Its interface is designed to help to manage the medical teams and patients in a more clear, efficient and easy way. It will also decrease the stress level and will allow doctors to offer more of their time to patients. CIPLAN has been developed in JAVA language and has a SQL database. Moreover, in order to provide a safe work environment and better security of the medical data, different access level are provided to the users.

Other software tools, such as GUIDE [70] or HEAT[54] have been proposed for the development and analysis of clinical pathways in order to control the effectiveness and efficiency of the medical interventions. However, these software tools do not provide decisions about the scheduling of patients. There are other commercial software solutions available for general scheduling which could be adapted to be used in a hospital. However, these solutions, in addition of having a high price, may not consider aspects related with the management of medical teams, waiting lists or ORs. Our proposed solution had been developed in collaboration with doctors of the “Lozano Blesa” hospital. So, their concerns and preferences had been considered not only at the level of surgical scheduling but also at the user interface level.

7.2.1 Features of CIPLAN

After the user has logged in, depending on the access level, the application can be used for:

1. Managing medical teams
 - Add/remove doctor from team or medical teams;
 - Move doctor to another medical team;
 - Change the medical leader of the team or the team name.
2. Managing patients
 - Add/remove/update patients or patient details;
 - Add/remove/change surgeries from patients medical history;
 - Manually schedule a patient to a certain medical team and doctor and to a certain day;
 - Unschedule a patient that cannot reach the hospital in the assigned day.
3. Managing operating rooms
 - Add/remove operating rooms from the operating theater;
 - Schedule a medical team in a certain day for each operating room;
 - Change data in the currently available time table.
4. Planning and scheduling patients from a certain waiting list for a given number of working days.
5. Adding the information of the performed surgery for each patient to whom the scheduled surgery is completed.

7.2.2 Back end implementation

The *back-end implementation* has 5 different internal packages. One package for each *user interface* language available (English and Spanish), one for the *main-files*, one for the *common classes* (e.g., Patient, Doctor, Surgery or

Database Connection/Query) and another one for defining and computing the *heuristic model*

The *main-files* package contains the entry-point class of this application which creates a “Welcome window” and makes it visible (Fig. 7.3).

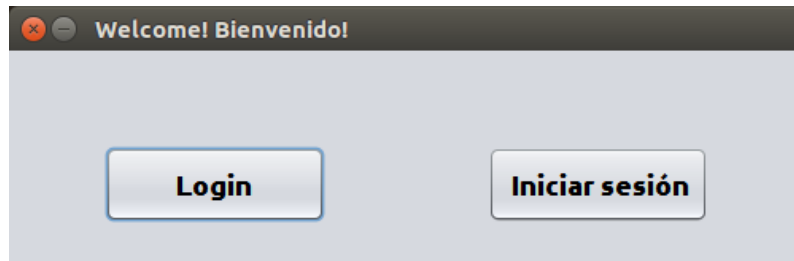


Figure 7.3: Welcome window of CIPLAN

At this point, the user will choose the language to use during the next session. For the following descriptions, we will use the English version of the interface. If the application cannot connect to the database, an error message will be prompted.

The session is initialized with a login form in the language chosen by the user (Fig 7.4). If the user name is not in the database or the password is wrong, an error message will be prompted to inform the user about the login failure. After a successful log in, the application will start with the user’s clearance level. Further details about the interface application can be found in Sec. 7.2.4.

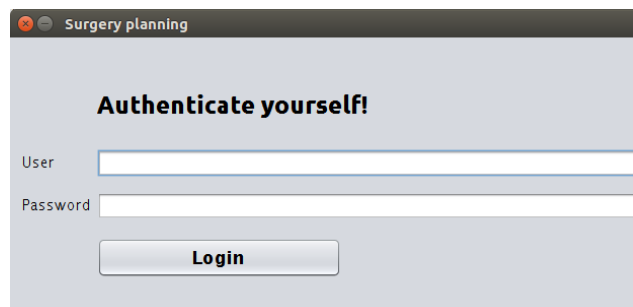


Figure 7.4: English session login window for the surgery planning application

The *common File* package contains all the classes for defying and managing patients, users, doctors, operating rooms, as well as the connection and queries

with the database. All these classes are explained below:

- **PatientIdentifiers** contains patientId, lastName, firstName, gender, dateOfBirth, remarks and medicalHistory. Remarks are strictly connected with the patient and not with any surgery from patient's medical history
- **PatientMedicalHistory** contains vectors of same length for surgeryName, pathologyName, doctorInCharge, doctorTeamLead, admissionDate, scheduledForSurgery, scheduledDate and surgeryCompleted. The fields "scheduled- ForSurger" and "surgeryComplete" are boolean fields that express if the patient was scheduled for surgery (or not) and if the surgery was completed (or not) - only if the patient was scheduled. The "admissionDat" field is used to arrange the patients in the waiting list regarding the number of days the patients are waiting to be scheduled.
- **PatientList** is the interface class with the database for managing new or existing patients.
- **MedicIdentifier** contains the last name and the first name of the doctor, the doctor's department, "IDnumber" and the coordinator's "IDnumber"
- **MedicalTeams** is the interface class with the database for managing new and existing doctors and medical teams
- **SurgeriesList** is the interface class with the database for managing new and available surgeries that can be performed in the current department
- **ORClass** contains the ID, name and type of the operating room, as well as the booking details like: date, starting and ending hour and teamName
- **OperationRooms** is the interface class with the database for managing the operating theatre and the assigned medical teams to each operating room
- **Users** is the interface class with the database for managing the users. This class contains a private passPhrase to encrypt the passwords.

The *heuristic model* package contains the implementation of the SHA. To use this class and schedule patients, an instance of it needs to be created. Before calling the public method of scheduling patients, the “patientList” needs to be set. If there are no patients in the waiting list the scheduling does not start. Also, if the number of time blocks to schedule is less than 1, the scheduling will not start. In both cases, an error message will be displayed.

The *user interface* package will be explained in more details in Sec. 7.2.4.

7.2.3 Database

The application has a local relational database (within the hospitals intra-net). Its structure is showed in Fig. 7.5.

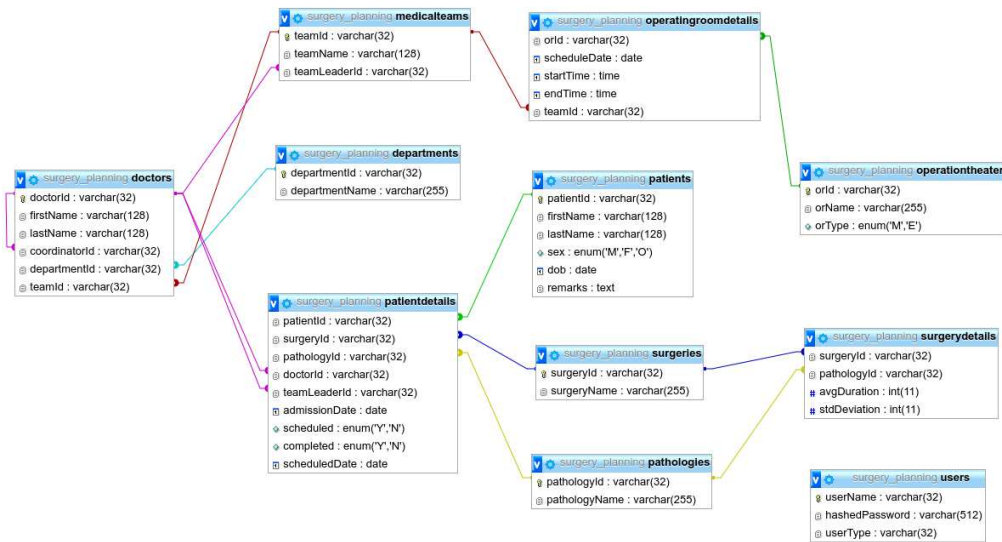


Figure 7.5: Database structure for CIPLAN

In the lower right corner, is the “User” table, it contains information about all users allowed to connect, their access level and their hashed passwords. Access levels are described as follows:

- *head of department*: can book time blocks of the ORs for the teams. Moreover he/she also manage the medical staff;
- *coordinator (team leader)*: can perform scheduling for his team’s waiting list;

- *medic*: can add/remove new/performed surgeries in the department;
- *assistant (nurse)*: add/update patients in the database;

Besides “Users” table, there are other primary tables in the database, as it follows:

- **surgeries** contains id and name for the available surgeries in the department
- **pathologies** contains id and name for the pathologies treated in the department
- **departments** contains id and name of all the departments joined the current medical teams
- **patients** contains id, first name, last name, gender, date of birth and remarks

All the other tables depend on one or more tables described above. These are:

- **doctors** contains doctor Id, first name, last name, coordinator Id, department Id and team Id
- **medicalTeams** contains team Id, team name and coordinator Id
- **patientDetails** contains patient Id, surgery Id, pathology Id, doctor Id, teamLeader Id, admissionDate, scheduled, completed, scheduledDate
- **operatingRoomDetails** contains operatingRoom Id, scheduleDate (=booking date), start time, end time, team Id
- **operationTheatre** contains operatingRoom Id, operatingRoom Name, operatingRoom Type (Morning or Afternoon)

According with the details given in previous section, database files (databaseQueries and databaseConnection) are in the common files package. DatabaseQueries class contains methods with all the queries for the database.

7.2.4 User interface

The User Interface (UI) is designed and implemented in six main panels. The first five appear in discontinuous colored boxes in the flowchart of the DSS (Fig. 7.2). They are split as follow:

1. *Medical teams*: all the actions about medical staff can be performed in this panel by the head of department. All other users can only see the information.
2. *Patients*: all users can see/add/update/remove a patient or patient's details.
3. *Surgeries*: the set of elective surgeries that can be performed in the department are added in this panel. Existing surgeries can be updated or removed by any user with access level "Medic" or above. The average duration and the standard deviation of any surgery can be consulted here.
4. *OR timetable*: all users can see the timetable for each OR, but only the head of department can edit data from this panel.
5. *Schedule*: all coordinators have access to this panel for creating schedules from team's waiting list.
6. *User*: every user can change its own password. The head of department can edit other users credentials. This panel is also used for ending the current working session by logging out the user. The application will not close if the user does not logout.

Medical teams panel

"Medical teams" panel is shown in Fig. 7.6. It has three different kinds of subpanels: i) the last one allows managing doctor, ii) the penultimate allows managing medical team, and iii) the other ones give information about the medical teams.

In Fig. 7.6, first 5 subpanels are describing the current 5 available teams. The name in each one of these subpanels correspond with the medical team's name (e.g. "Superman", "Rainbows" or "coder fault"). These subpanels have

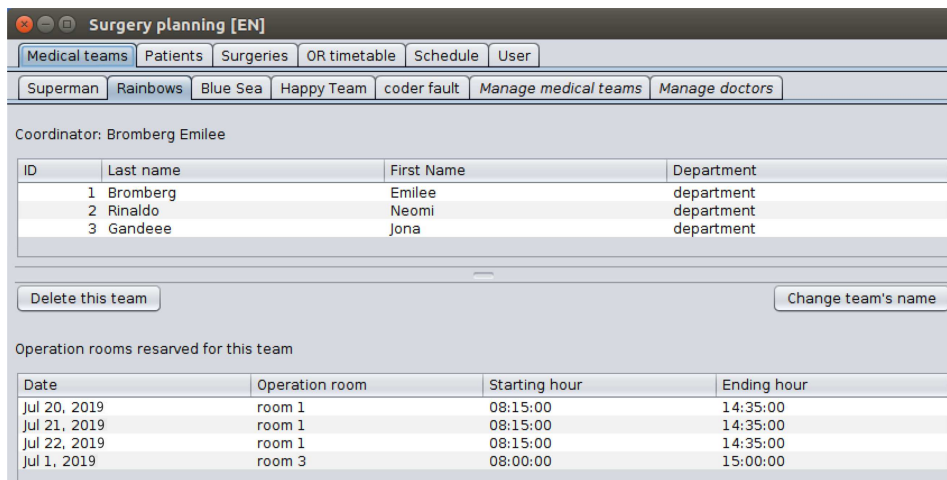


Figure 7.6: Medical teams panel for CIPLAN

all the details about the medical team: doctors belonging, team coordinator and time blocks booked for the team.

The two buttons available on a “team” subpanel (Fig. 7.6) are enabled only for team leaders. A team leader can change the name of his own team. After the change, any user will be able to see it. If the team does not have members, the leader can delete the team. “Manage medical teams” subpanel (Fig. 7.7) has two parts but only one of them is enabled at a time.



Figure 7.7: Medical teams management subpanel for CIPLAN

The enabled part in Fig. 7.7, allows the user to change the medical team leader with another doctor from the same team, or to add a new doctor to an existing team. If the user wants to add a new medical team, then checking the box ‘ Add new medical team” is mandatory. Doing so, the second part will be enabled and the first part will be disabled. The user has the ability to manage the medical teams only if it is Head of Department. No other user type is allowed to modify the structures of the medical teams and their coordinators. Even though the medical staff can only be updated by the Head of Department, all users can see the details of any doctor in “Manage doctors” panel (Fig. 7.8).

Figure 7.8: Medical staff management for CIPLAN

From the drop-down box, in Fig. 7.8, user can select any existing doctor and see its details. The available details about the selected doctor include: coordinator’s details, team name, and doctor’s details. The Head of Department can remove the selected doctor if the current doctor is not a team leader. For deleting a team leader, another doctor from the same team must take the team leader position. Doing so, the doctor is not a team leader anymore and can be removed from the database. If there is no other doctor in the selected doctor’s team, removing the doctor means leaving a medical team empty (no member

at all) and this is not a realistic option. For this case, when the team should no longer be in the system, the Head of Department should move selected doctor (current team leader of a team that is no longer needed) into another team. In the bottom-right side, there is a button for this step. After this operation, the empty team will be deleted from the system. At this point, the selected doctor is not a team leader anymore and the doctor team is no longer in the system. So, there are no restrictions anymore in removing the selected doctor from the database. This way of removing the medical team or team leader from database has been chosen because, when a doctor is removed from the database, all the patients that the doctor is assigned to perform a surgery, will be automatically transferred in team leader's waiting list.

Patients panel

The second main panel is about patients and it is composed of four subpanels. The first subpanel is shown in Fig. 7.9. It contains the team (or doctor) patient

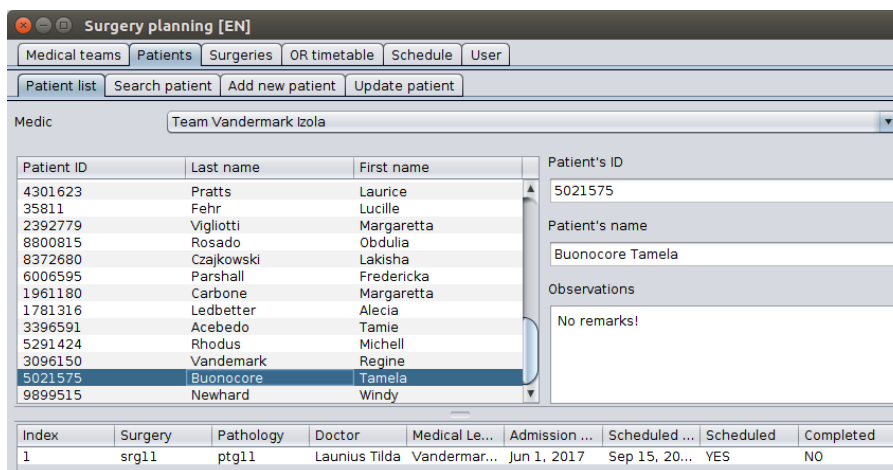


Figure 7.9: Patient list subpanel in CIPLAN

list, depending on the selection in the drop-down box. After a selection is made, the list of patients will be shown in the left side of the interface. In this panel, all the details about a patient, in a certain waiting list, can be seen. When a patient from the list is selected, fields on the right side and the bottom table will be updated with selected patient's details and its full medical history. To see details about a certain patient, user should use "Search patient" subpanel

Index	Surgery	Pathology	Doctor	Medical Le...	Admission ...	Scheduled ...	Scheduled	Completed
1	srg47	ptg47	Maguire Ka...	Vandermar...	Jun 1, 2017		NO	NO

Figure 7.10: Search patient in CIPLAN

(Fig. 7.10). The patient can be searched in database using 2 different criteria (mapped on two different buttons):

- *patient Id.* The id is the history medical number that a patient has assigned in the local database when the patient was added
- *patient name.* It is recommended full name, but using partial name (either last or first) is also acceptable.

Figure 7.11: Add new patient in CIPLAN

If a patient does not exist in the database, it can be added using “Add new patient” panel (Fig 7.11). To make the search option easier and accurate, all the fields are mandatory. If in the adding of a new patient there is an existing patient in database that matches the input details, an error message will be prompted. After patient details have been added in the database, a message will inform the user that the adding operation was completed successfully. Moreover, a waiting list of patients can be upload from a CVS file using the “Add a list of patients (CVS)” button. The CVS input file must have a predefined structure, in other cases, an error message is shown.

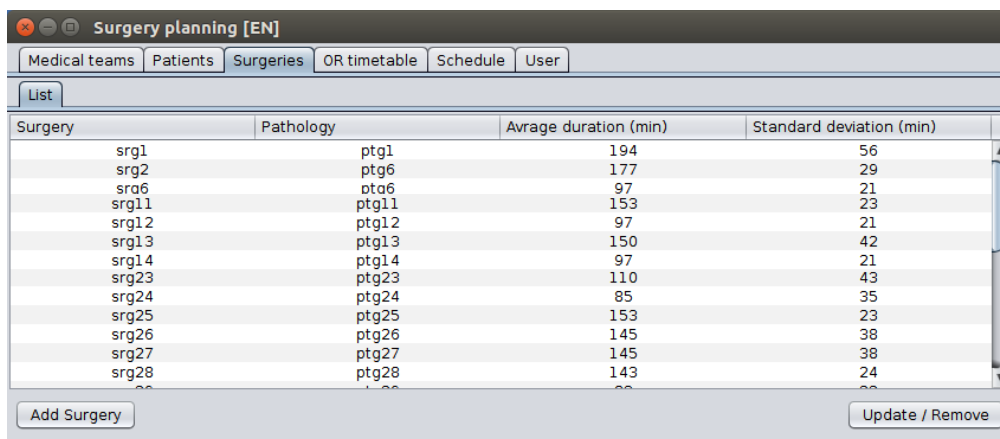
Figure 7.12: Update patients information in CIPLAN

In the “Update patient” panel (Fig. 7.12) any personal details about an existing patient can be modified. Like in “Search patient” panel, “Update patient” panel also allows the user to search patients by name (and birth date) or by ID. In the right side, the user can add or update surgeries for patient. “Add surgery” button will be enabled only if the “Add new surgery” option is checked. In this case, user can add one or more surgeries (only one at a time) for patient using the drop-down boxes. If “Update existing surgery” option is checked, the drop-down boxes will only contain approved surgeries that were not completed (either scheduled or not). At this point, when a surgery

is selected, user can change the medical team and/or the doctor in charge or schedule/unschedule the patient. If the doctor in charge cannot complete the surgery, another doctor (from the same team or another one) can be assigned. If the patient was scheduled for surgery but unavailable on the schedule date, user can unschedule the patient's surgery or reschedule it on another free day. Any changes should be saved using "Save surgery" button. If for some reason, the patient (either scheduled or not) will not have the surgery anymore, user can remove the surgery from patient's medical history. Everything in "Patients" main panel is available to all users, regardless the clearance level.

Surgeries

The third main panel, "Surgeries", is shown in Fig 7.13. This panel contains a list with all surgeries that can be performed in the Department. Every surgery has assigned the usual *pathology*, the average duration [min] and the *standard deviation* [min]. Buttons are enabled for all users with clearance level greater or equal with "medic". These buttons offer the possibility to add new surgeries, update the timings for the existing ones or remove surgeries from database. Removing surgeries can only be done if there is no doctor assign to them and no patient admitted for this surgery and not performed yet.



Surgery	Pathology	Average duration (min)	Standard deviation (min)
srg1	ptg1	194	56
srg2	ptg6	177	29
srg6	ptg6	97	21
srg11	ptg11	153	23
srg12	ptg12	97	21
srg13	ptg13	150	42
srg14	ptg14	97	21
srg23	ptg23	110	43
srg24	ptg24	85	35
srg25	ptg25	153	23
srg26	ptg26	145	38
srg27	ptg27	145	38
srg28	ptg28	143	24

Figure 7.13: Surgery panel in CIPLAN

OR timetable

Fig. 7.14 shows the fourth main panel called “OR timetable”. It contains as many subpanels as there are ORs available for performing elective surgeries in the department. So, each subpanel represents one OR. Selecting one subpanel, it is shown the time-table of the corresponding OR. These subpanels are enabled for editing only for the Head of Department. The other types of user can only see the time blocks booked for any OR.

Each operating room has a default name that can be changed using the corresponding button. A team can be booked in the selected operating room only if:

- there is no other team booked in the desired date
- team is not booked in the desired date on another operating room

Medical Team	Date	Starting hour	Ending hour
Rainbows	Jul 20, 2019	08:15:00	14:35:00
Rainbows	Jul 21, 2019	08:15:00	14:35:00
Rainbows	Jul 22, 2019	08:15:00	14:35:00
Blue Sea	Jul 23, 2019	08:15:00	14:35:00
coder fault	Jul 24, 2019	08:15:00	14:35:00
Happy Team	Jul 25, 2019	08:15:00	14:35:00
Blue Sea	Jul 26, 2019	08:00:00	15:00:00

Figure 7.14: OR timetable panel in CIPLAN

The processes from which the head of the department can book an OR for a given team is as follow. First, the team and the date must be selected. Then the starting and ending time of the block are chosen. Finally, if all the fields are filed correctly, pressing “Book” button a new booking will be added in the operating room’s timetable. There is no edit option for existing bookings. To do so, the wrong booking must be deleted, then create another booking. To

delete an existing booking, a pop-up window will request for the date of the booking that should be deleted.

There are two types of ORs available: morning and afternoon. The difference between them is only about the timing. A morning OR can be available between 8:00 and 17:55, but an afternoon OR can be available only between 14:00 and 20:55. Regardless of the type, an OR can be deleted only if there is no team booked from the current date forward and if it is not the only one left.

Schedule

The fifth panel, “Schedule”, is the main reason of CIPLAN. This panel allows to any user with a clearance level of at least “coordinator” to schedule patients automatically.

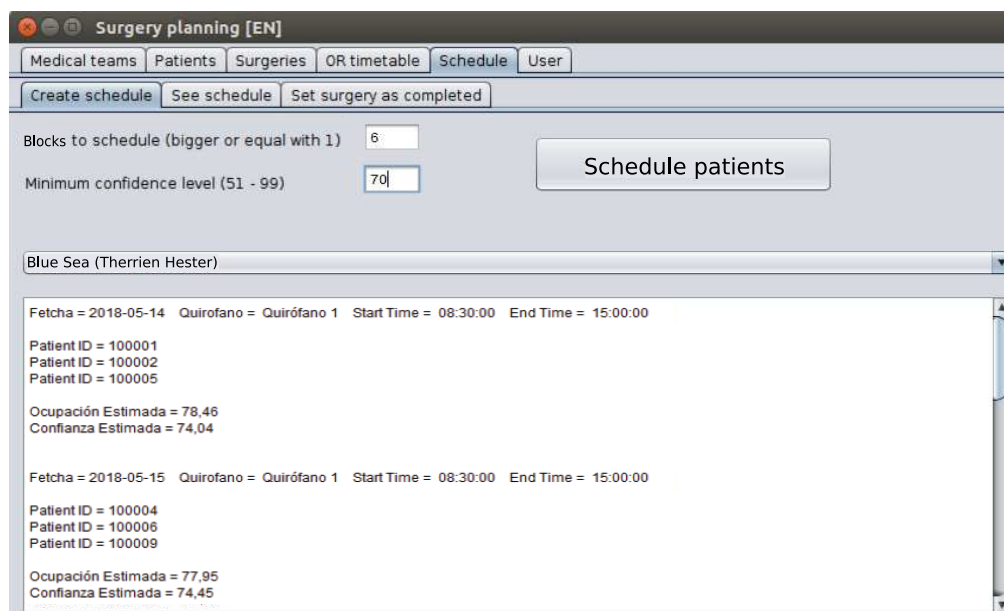


Figure 7.15: Create scheduling in CIPLAN

The sub-panel “Create schedule” is shown in Fig. 7.15. It allows the user to schedule patients for a certain number of time blocks. The maximum number of time blocks is automatically updated in the corresponding text box when a medical team is selected. For the scheduling program to start, the number of

time blocks to schedule has to be greater or equals to 1. Another parameter needed for scheduling patients is the minimum confidence level allowed not exceeding the available time. After the team is selected and all parameters are between boundaries, a schedule can be ordered with “Schedule patients” button. A scheduling summary will appear in the white area at the bottom of the panel. It will contain the following information for each time block:

- the date, the name of the OR and the starting/ending time booked in the block
- the ID of the patients scheduled
- the estimated occupation rate and the confidence level of not exceeding the available time.

To see scheduling details for a certain time block, subpanel “See schedule” (Fig. 7.16) can be used. First, an OR or a medical team must be selected and then a date is chosen. The calendar will only show the available dates for the selected OR or medical team. All details about the patients scheduled are shown. Moreover, at the bottom of ”See schedule” panel, can be found the total estimated time along with a percentage from the total day time, also shown below.

Team Name	Operation ro...	Patient Id	First Name	Last Name	Surgery	Pathology	Estimated Duration[min]	Doctor Name	Admission date
Blue Sea	quirofano	7424396	Clarissa	Raab	srg13	ptg13	150	Ruggien Syble	Jun 1. 2019

Figure 7.16: See schedule in CIPLAN

After a surgery is performed, it should be checked as completed. Sub-panel ”Set surgery as completed”, as shown in figure 7.17, allows users to do so. As a first step, the user should select the day in which the patient was scheduled.

To set a surgery as completed, the user has to first select the patient and then press the “Set completed” button. An informing message will be shown before surgery to be set as completed. At this point, the user can either cancel the operation or accept the action of setting it as completed.

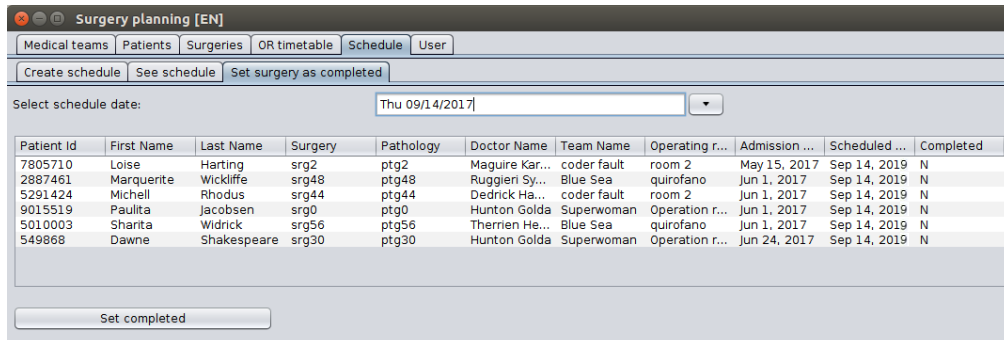


Figure 7.17: Set a surgery as completed in CIPLAN

User

The last main panel contains only “User” related operations like: Create user (Fig. 7.18), delete user (Fig. 7.19), manage user (Fig. 7.20) and logout (Fig. 7.21). Only the Head of Department can create a new user or delete an existing one. When creating a new user, the default value for the clearance level of the new user is “medic”.

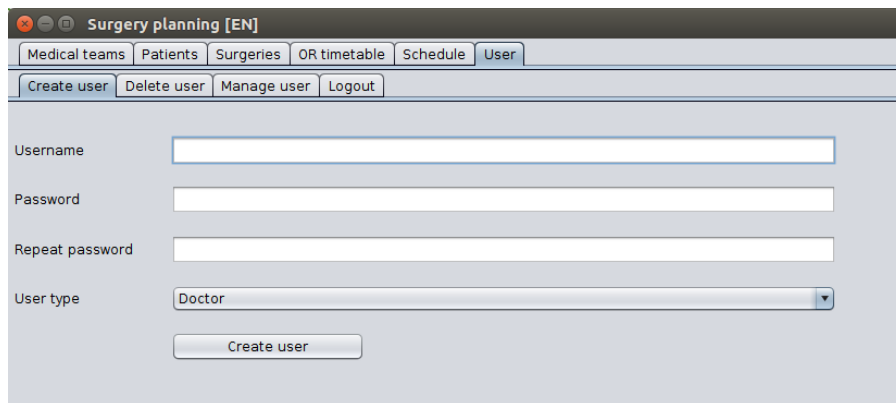


Figure 7.18: Create a new user in CIPLAN

If a user should no longer have access to the application, the Head of Department can delete the user and its details (password and clearance level) from database. Once the operation is successfully completed, the user that

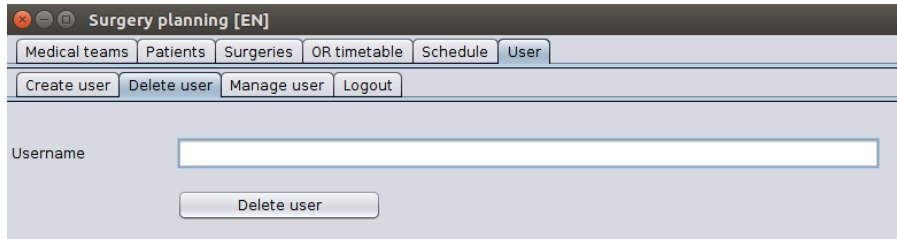


Figure 7.19: Delete a existing user in CIPLAN

had just been deleted, cannot connect into the application. If the user does not exist or cannot be deleted, an error message will be prompted.

If the user wants to change its personal details (password), user can do it in "Manage user" subpanel. For changing the password of its own account, the user needs to put the current password along with the new one. The admin user can change password of any other user without needing the current password of that user.

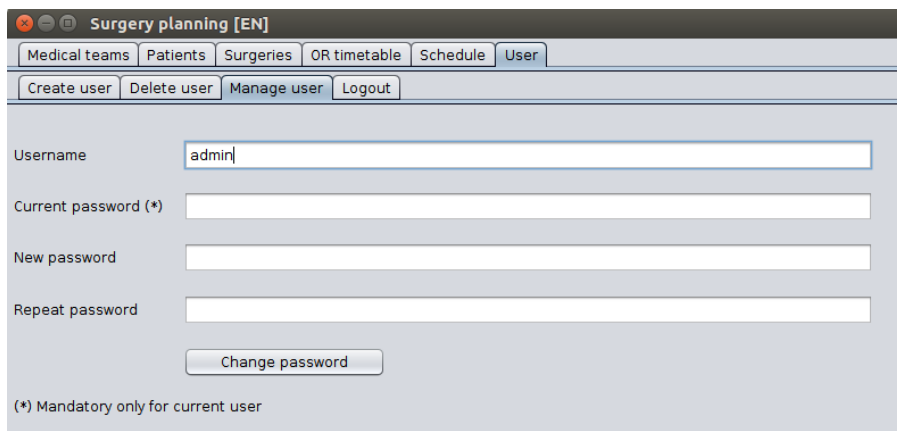


Figure 7.20: Modify access information for a existing user in CIPLAN

The last sub-panel contains only the logout button. The application cannot be closed through regular closing option regarding security reasons. Thus, to close the application, users must logout first.

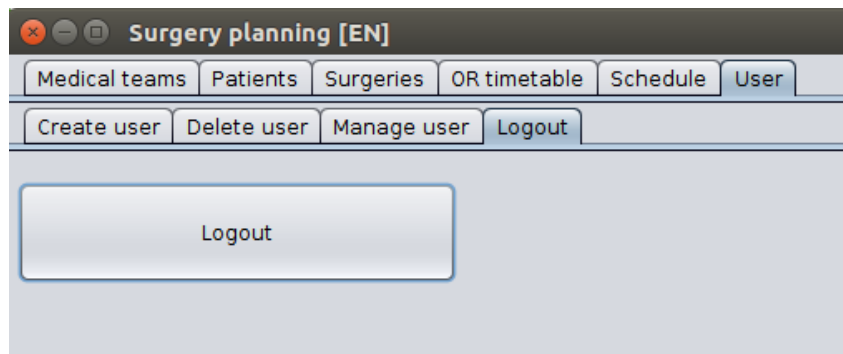


Figure 7.21: Logout

7.3 Real case application using CIPLAN

A case study using CIPLAN in the OSD of the LBH is analyzed. First, the hospital description is introduced, then the methodology used in the case study is given and finally the scheduling obtained using CIPLAN is compared with the manual scheduling.

7.3.1 Hospital Description

The LBH is a public hospital located in Zaragoza, Spain. The hospital provides health service to around 325.000 people and it is a reference center of specialized attention of a population over 1 million inhabitants. The hospital has 800 beds and 15 ORs for major surgery which is performed by 8 different surgical departments. We set our case study in the Orthopedic Surgical Department (OSD).

The OSD is composed by 5 medical teams and has assigned 3 ORs (OR_1 , OR_2 and OR_3) from 8:30 a.m. to 3 p.m. every day. The current policy of the department uses OR_1 and OR_2 for performing elective surgeries while the OR_3 is used for urgent patients. Tab. 7.1 shows the weekly OR time-table during the mornings in the OSD. According to Tab. 7.1 each medical team has two morning time blocks per week for performing surgeries on the elective patients.

Moreover, there are others time blocks available for the department during the afternoons (from 15:30 to 20:30). However these afternoon blocks have a

Table 7.1: OR time-table in the Orthopedic Surgical Department of the HCU

	OR 1	OR 2	OR 3
Mon	team 1	team 2	team 3
Tue	team 4	team 5	team 1
Wed	team 2	team 3	team 4
Thu	team 5	team 1	team 2
Fri	team 3	team 4	team 5

variable time-table depending on the urgent surgeries. Normally, an afternoon time block per week is assigned to each medical team.

7.3.2 Methodology of the case study

The main objective of the developed software tool is to improve the efficiency and quality of the surgical service. So, in the case study we compare the scheduling obtained manually with the scheduling obtained by using CIPLAN. The methodology is as follows:

1. Analysis of manual scheduling.

Considering historical data of one medical team of the OSD, we analyze the elective blocks scheduled during one year. In total, 156 consecutive surgical blocks of elective patients are studied. 90 of them have morning schedule while the others 56 have afternoon schedule. For each time block, the following information is obtained:

1. *Type of surgeries performed.* In total 93 different kinds of surgeries has been performed. Graph in Fig. 7.22 shows the occurrence of most common types of surgeries while Tab. 7.2 indicates the average and standard deviation values of their durations. Notice that 10 types of surgeries represent around 75% of the total surgeries performed.
2. *Starting and ending time of each surgery (real duration).*
3. *Expected and real occupation rate.* It is considered that the effective time in a surgery block is when a patient is being operated, therefore the

occupation rate in a time block is computed as the effective time divided by the total time.

4. *Confidence level of not exceeding the available time.* For computing the probability of overtime, the delayed of the starting time (Dt) in a block and the cleaning times (Ct) are assumed to follow normal distribution such that: $Dt = N(10, 12)$ and $Ct = N(20, 15)$.
5. *Real ending time* of the blocks.

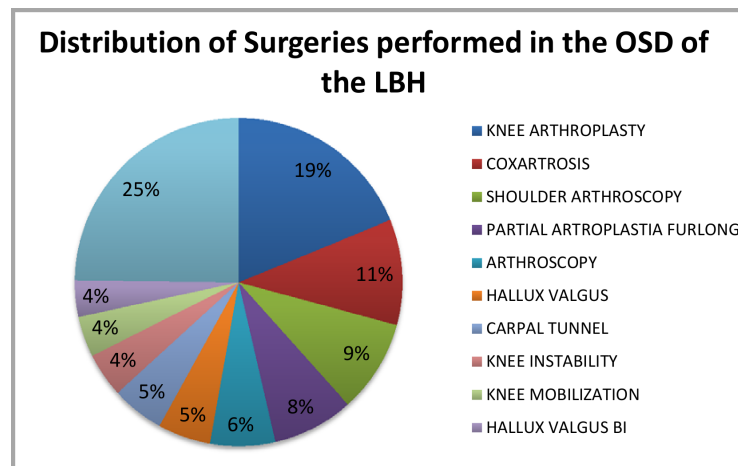


Figure 7.22: Occurrence of surgeries in the OSD of the LBH

2. Create the waiting list.

The manual scheduling and consequently the surgeries types in the waiting list are known. However, the preference order of each surgery in the waiting list is unknown. In order to be able to obtain an ordered waiting list from which the scheduling using CIPLAN is performed, it is assumed that the position of the patients (surgeries) in the list corresponds to the order in which they were operated by using the manual scheduling. That is, the first patient in the waiting list is the patient who was performed first in the first time block and so on. So, considering the surgeries performed in the 156 time blocks, an ordered waiting list of 397 patients is created.

Table 7.2: Duration of the surgeries more performed in the OSD of the LBH

Surgery type	Avg. duration [min]	Std. Deviation [min]
Knee arthroplasty	128	24,5
Coxarthrosis	127	28,2
Shoulder arthroscopy	113,3	29,6
Partial artroplastia furlong	115,7	27,5
Arthroscopy	77,9	18
Hallux Valgus	99	21,4
Carpal Tunnel	33,8	9,9
Knee instability	123,7	21,9
Knee mobilization	215	31,02
Hallux Valgus BI	97,2	17,15

3. Scheduling using CIPLAN.

First, the fields and data necessities to perform the scheduling using CIPLAN are included in the tool and then the scheduling is performed:

- All elective surgeries types that can be performed in the OSD are added to CIPLAN including their average duration and standard deviation (computed considering historical data of the team during last two years).
- A new medical team (team 1) is added to the tool. Moreover a new doctor (doctor 1) is added to team 1.
- All patients in the waiting list composed using the 156 time blocks are assigned to doctor 1.
- Considering the sequence of blocks in the manual scheduling, 3 time blocks are booked weekly for team 1 during 50 weeks. The available time of these blocks is exactly the same than in the manual scheduling: some of them from 8:30 to 15:00 (morning blocks) and other ones from 15:30 to 20:30 (afternoon blocks).
- Using CIPLAN, the 150 time blocks previously booked are scheduled considering a minimum confidence level of not exceeding the total time equal to 69 %. This value is fixed by doctors.

4. Analysis of the scheduling performed using CIPLAN.

For each time block scheduled, the following parameters are obtained:

- the expected occupation rate.
- the confidence level of not overtime.
- the “real” occupation rate.
- the “real” ending time.

The scheduling obtained by using CIPLAN has not been performed. So, in order to compute the “real” occupation rate, the time spent when the surgeries were performed (manual scheduling) has been considered.

7.3.3 Comparing the manual scheduling method with CIPLAN

Considering the methodology explained in the previous section, a comparison between the scheduling obtained manually and the scheduling obtained using CIPLAN is shown here. Particularly, the *expected occupation rate* (Fig. 7.23), the *real occupation rate* (Fig. 7.24), the *probability of not exceeding the available time* (Fig. 7.25) and the *ending time* (Fig. 7.26) of the blocks are represented. Due to the large number of blocks analyzed (150), for easier reading of the graphs, they are divided on morning and afternoon. Moreover all data of the scheduling can be consulted in Appendix A.

Fig. 7.23 shows the *expected occupation rate* in each time block. The average value obtained using CIPLAN (77.41%) is greater than using the manual method (75.5%). Moreover, it can be seen that in the scheduling obtained using CIPLAN the values of the expected occupation rate in each time block are more concentrated around the average value ($\sigma = 3.61$) than using the manual scheduling ($\sigma = 7.42$).

In Fig. 7.24 the *real occupation rate* obtained in each time block is shown. Again, the average value obtained using CIPLAN (76.92%) is greater than the average value using the manual method (74.39%). Using CIPLAN the improving in the occupation rate with respect to use the manual method is of 2.53 %.

Expected occupation rate

Avg. CIPLAN = 77.41%
Avg. Manual = 75.5%

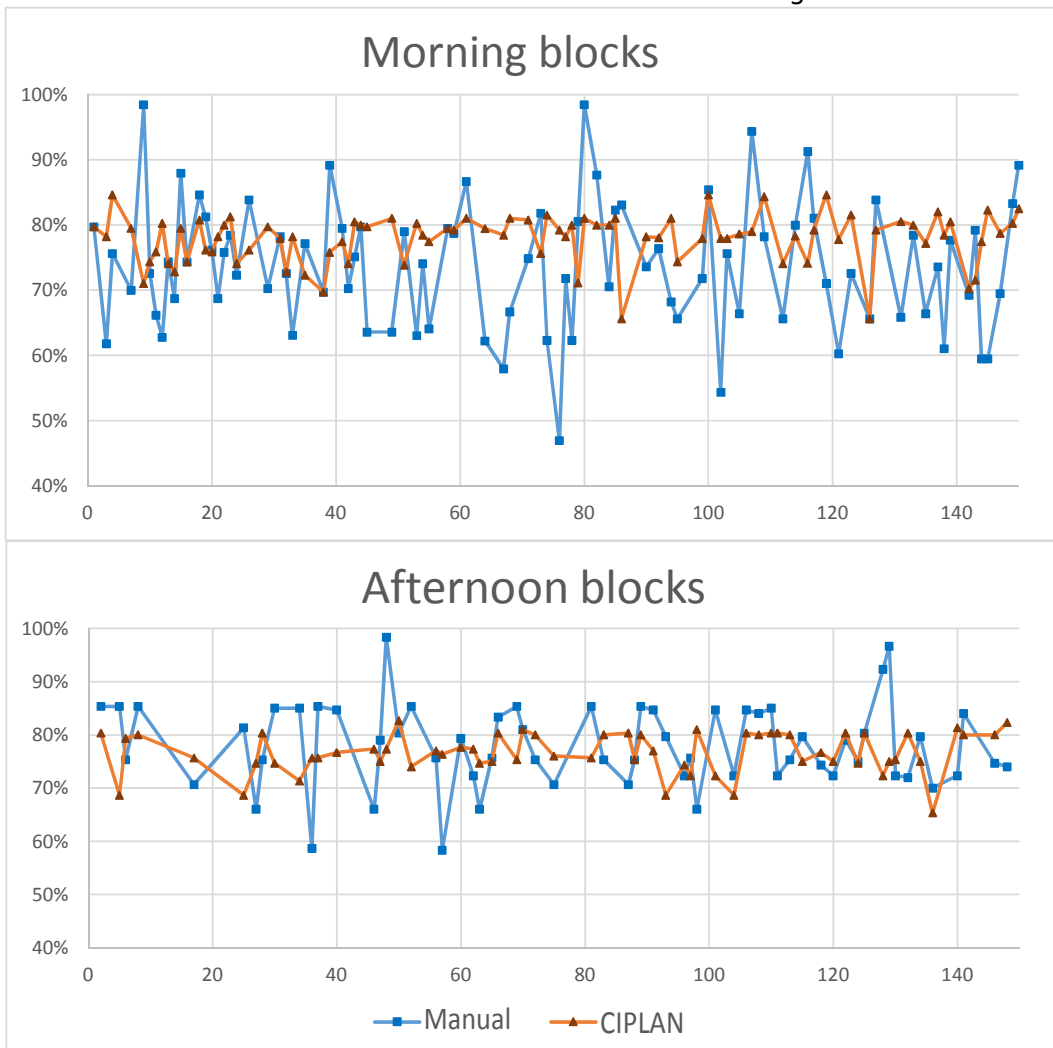


Figure 7.23: Comparison of the expected occupation rate obtained manually and using CIPLAN

In Fig. 7.24 also is possible to check that using CIPLAN there are some time blocks with occupation rates that are too high (over 95 %) and could lead to exceed the available time. This does not happen as frequently in the manual scheduling. Analyzing the results, we have observed that the scheduler (in the manual method) know the surgery skill of the surgeon and can estimate better

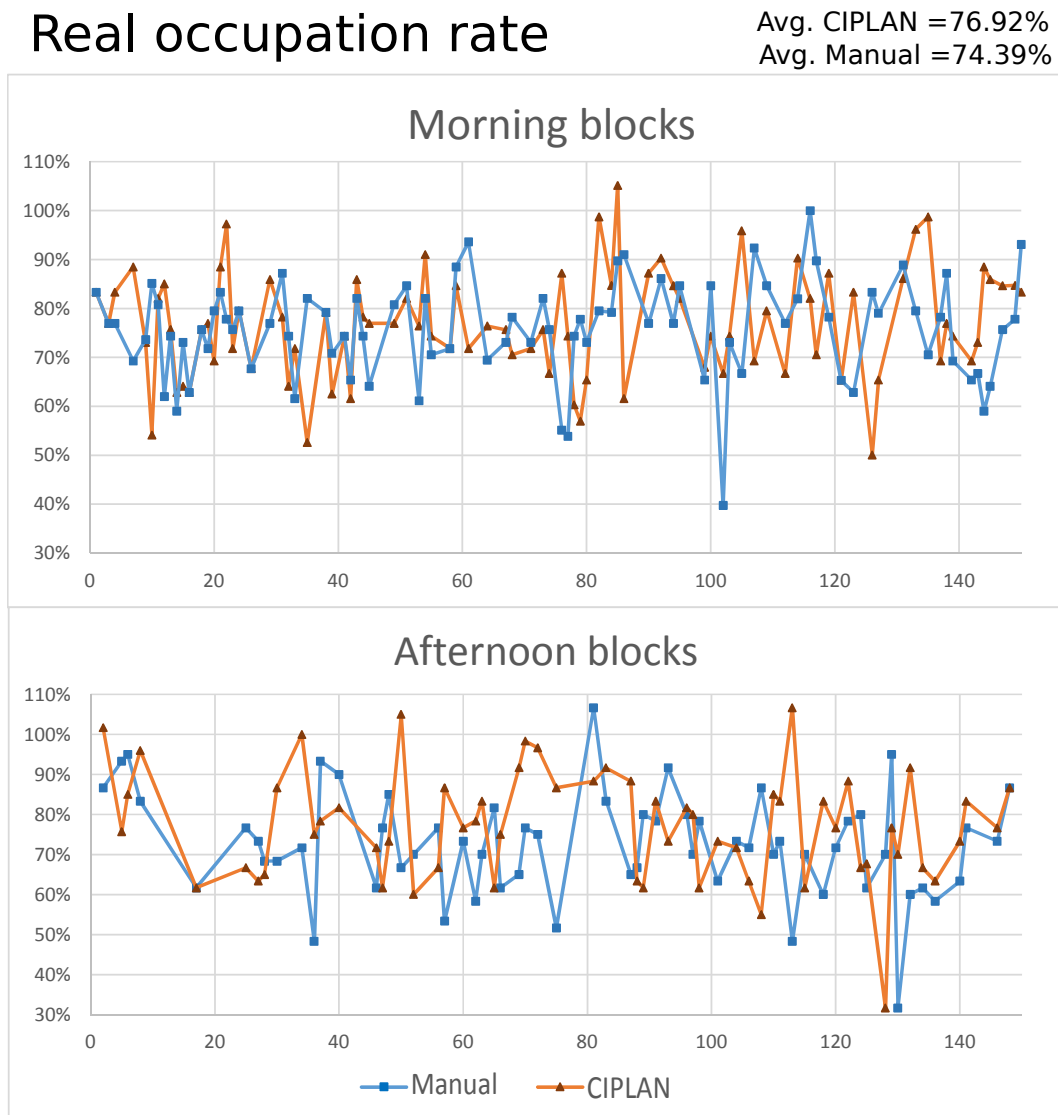


Figure 7.24: Comparison of the real occupation rate obtained manually and using CIPLAN

the expected duration of the surgery depending on the surgeon. However the CIPLAN scheduling algorithm initially use the average duration of the surgeries. These average durations are computed considering historical data, independently of the surgeon. As the tool is used, the time spent in each surgery as well as the surgeon who performs it will be registered in CIPLAN.

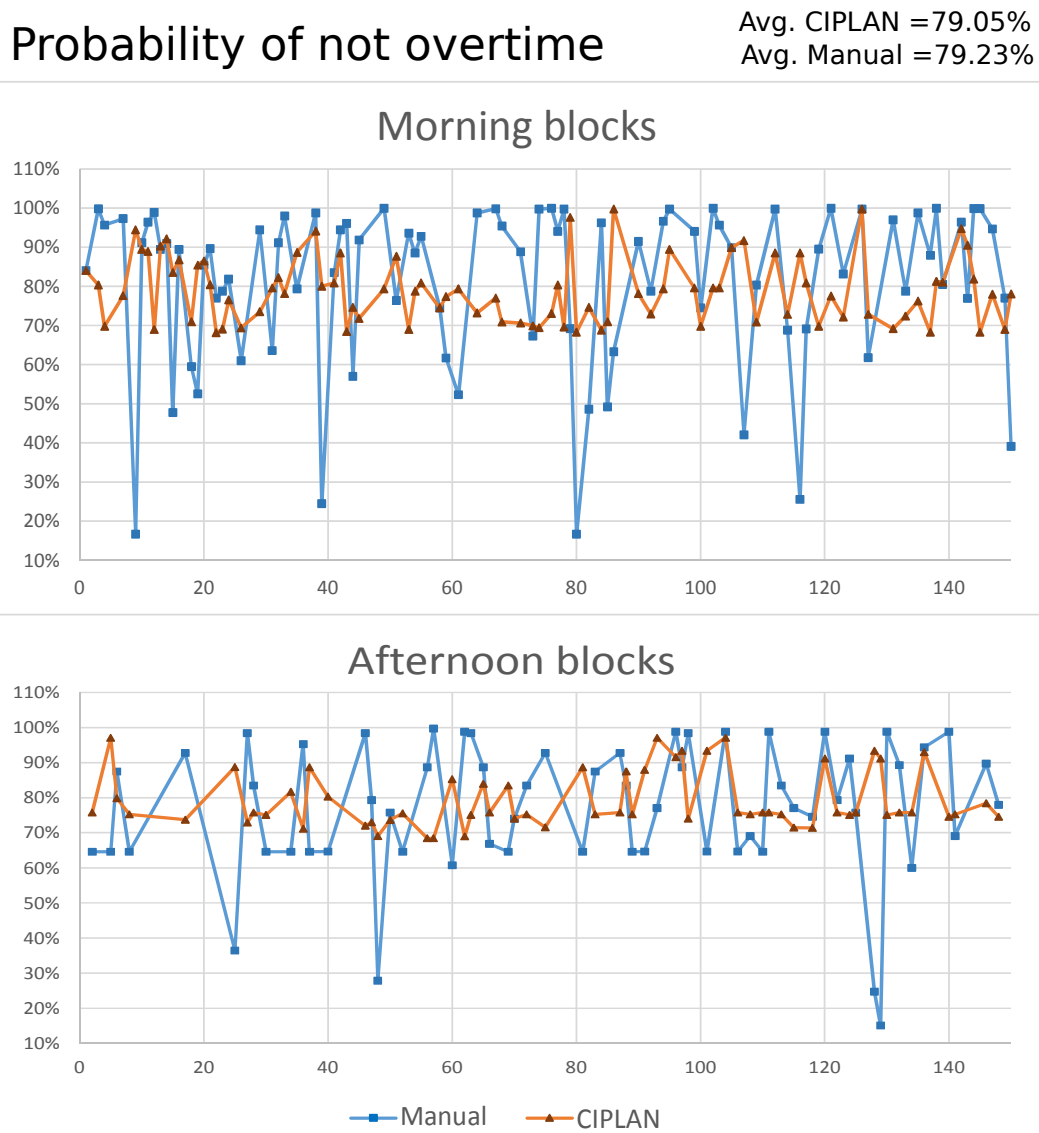


Figure 7.25: Probability of not exceeding the available time: Manual VS CIPLAN

So, the surgery duration will be more and more customized depending on the surgeon and better results will be obtained.

In Fig. 7.25 the confidence level of not exceeding the available time is shown. Using CIPLAN there are not time blocks with confidence level lower than 69 %. However, using the manual method there are several time blocks with a really

low confidence level. For example in the 5th morning time block (block 7 in the Appendix A), the confidence level of not exceeding the available time is 16.69 %. This low value is obtained due to the scheduler knows the surgery skill of the surgeon. Particularly, in this time block, 3 knee arthroplasties have been scheduled. Considering the average duration and the standard deviation based on historical data, the expected occupation rate is 98.4 % and the confidence level of not exceed the available time is 16.69 %. However, the scheduler knows that the surgeon who has to perform these knee arthroplasties needs 25 minutes less than the average duration of his team. Consequently, the real occupation rate obtained is 73.3 % and the time block ends at 13:52.

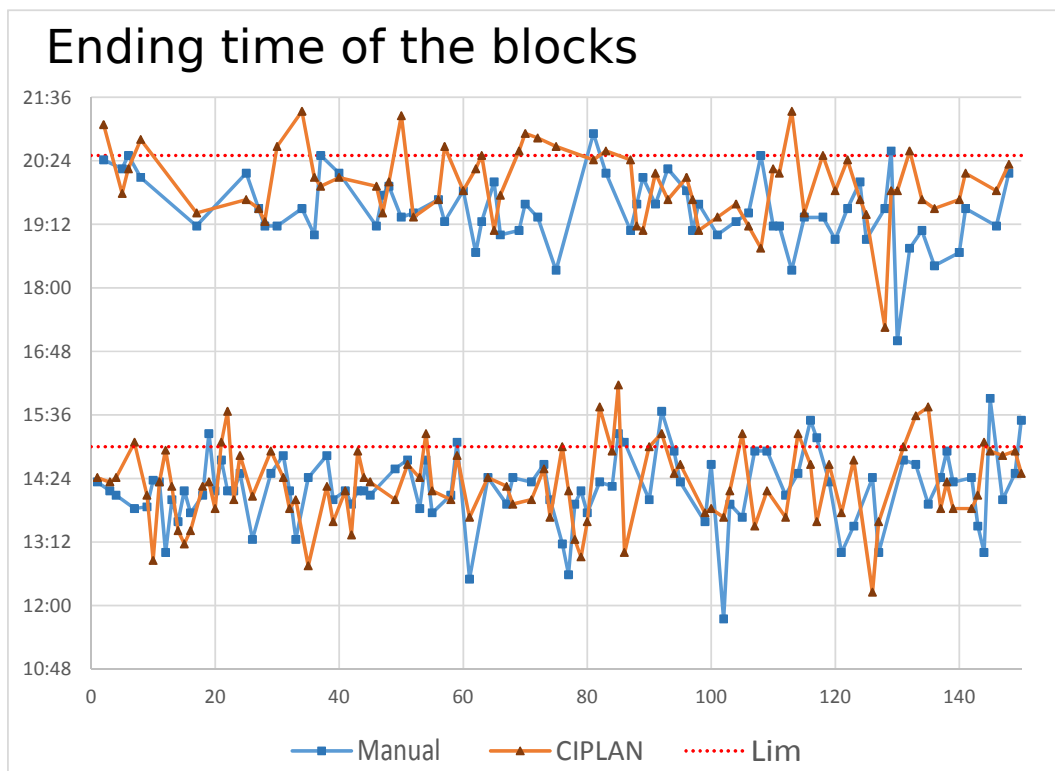


Figure 7.26: Ending time of the blocks: manual VS CIPLAN

In Fig. 7.26 we show the ending time of each block. It can be observed that using CIPLAN, there are 19 time block ending out of time while considering the manual scheduling there are 11 time blocks exceeding the ending time.

According to the result obtained in the case study, the quality of the service

decrease, ending some blocks out of time. However, this situation will be solved when the average duration and standard deviation of the surgeries will be customized depending on the surgeon. This customization will allow that the real occupation rate will be more similar to the expected ones and consequently, not only the number of block exceeding the available time will decrease, but also the exceeding time.

From an economical point of view, using CIPLAN the improving in the occupation rate with respect to use the manual method is of 2.53%. Only considering the OSD, this 2.53% implies an increment Δ_{Oc} of 109 effective hours per year in the use of the ORs (7.2):

$$\begin{aligned}\Delta_{Oc} &= 0.0253 \cdot \left(6.5 \left[\frac{hours}{block} \right] \cdot 90 \left[\frac{block}{team} \right] + 5 \left[\frac{hours}{block} \right] \cdot 56 \left[\frac{block}{team} \right] \right) \cdot 5[team] \\ &= 109,42 \left[\frac{effe. hours}{year} \right]\end{aligned}\tag{7.2}$$

Assuming an occupation rate of 76.92% and time blocks of 6.5 hours, these 109.42 effective hours are equivalent to an increment Δ_{Tb} of 21.88 morning blocks per year (7.3):

$$\Delta_{Tb} = \frac{109.42 \left[\frac{effe. h}{year} \right]}{0.7692 \left[\frac{effe. h}{h} \right] \cdot 6.5 \left[\frac{h}{block} \right]} = 21.88 \left[\frac{blocks}{year} \right]\tag{7.3}$$

Considering that the cost of open an OR during 6.5 hours is 5.850 euros, the use of CIPLAN in the surgical scheduling could suppose a saving of about 128.000 euros per year. The savings could be really greater extending the use of CIPLAN to other Departments.

The time execution of the first 30 blocks considering the manual (Fig. 7.27) and automatic (Fig. 7.28) schedulings are shown by two bar-graphs. Each row is composed by boxes that represent different actions in one surgical time block. Colored boxes represent surgeries while white boxes are cleaning times. On the other hand, in order to check how much the order of patients in the waiting list is respected, we have included the preference order of the patients inside the colored boxes. Due to the fact that the waiting list of patients has been constructed considering real surgeries, in Fig. 7.27 the patients are perfectly

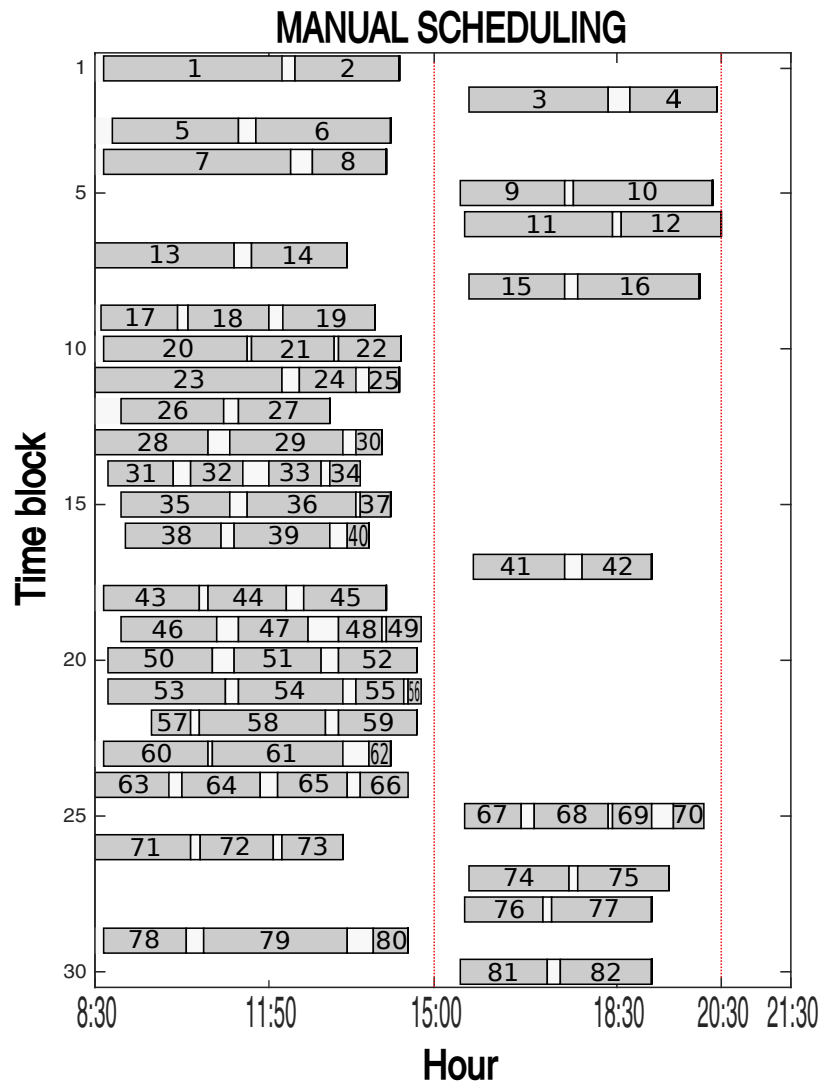


Figure 7.27: Manual scheduling of the blocks

ordered. However, considering the automatic method (Fig. 7.28), although the SHA try to make the scheduling respecting as much as possible the order of the patients in the waiting list, a little disorderliness results. In order to see it in a visual way, three colors have been used with a different meaning:

- The patients delayed or advanced at most two time blocks with respect to the manual scheduling (perfectly ordered) are in green boxes.

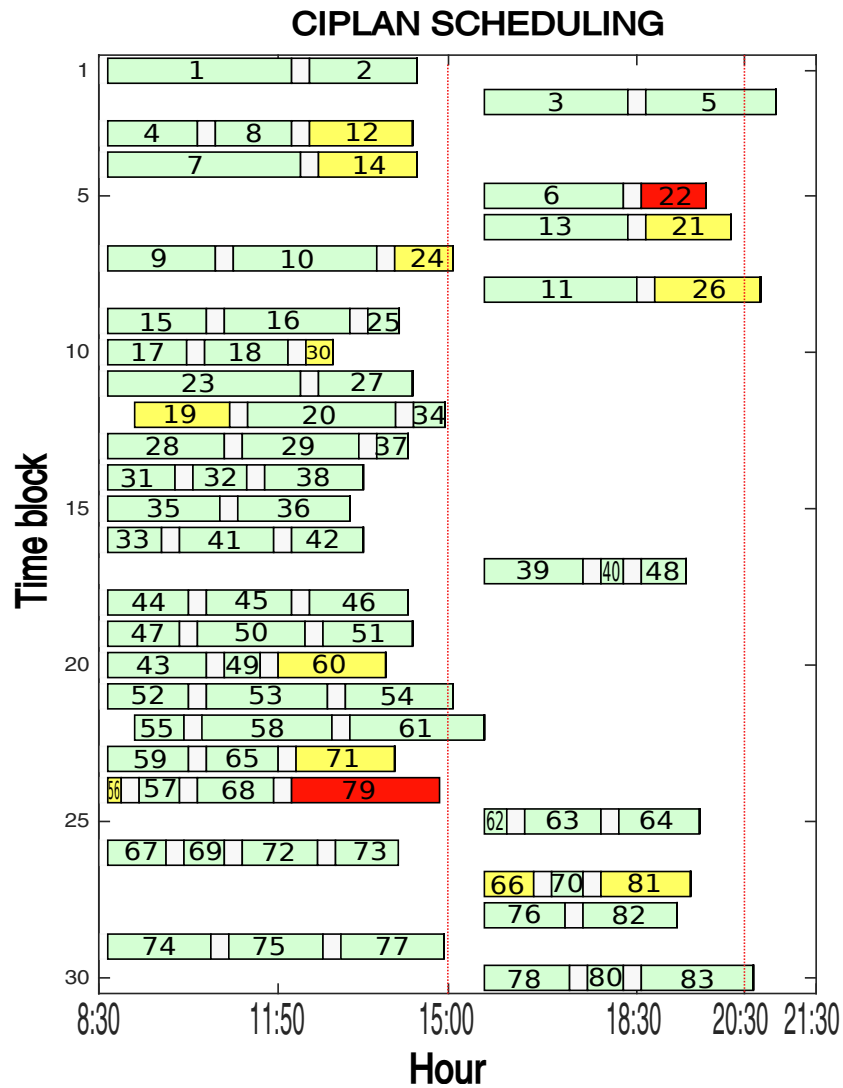


Figure 7.28: Automatic scheduling of the blocks

- The patients delayed or advanced 3 or 4 time blocks with respect to the manual scheduling are in yellow boxes.
- The patients delayed or advanced more than 4 time blocks with respect to the manual scheduling are in red boxes.

The graphs in Fig. 7.27 and Fig. 7.28 only show the first 30 blocks. In Appendix A can be consulted the scheduling of the 150 blocks. The number of

patients advanced or delayed 3 or 4 blocks (yellow) is 80. However, only 5 of them (19, 56, 66, 125 and 260) have been delayed. The number of patients advanced more than 4 time blocks (red) is 49 and there are not patients delayed more than 4 blocks with respect the manual scheduling. This means:

- 1.33% of the patients are delayed 3 or 4 blocks with respect to the preference order.
- The rest of patients have been scheduled no more than 2 blocks after than the block where they were scheduled considering the manual method.

The doctors consider that the order of the patients in the waiting list is much more respected in the scheduling obtained using CIPLAN than in the scheduling carried out manually by them.

7.4 Discussion

Normally, in hospital, doctors are responsible for scheduling patients from the waiting list to the time blocks. Currently, this task is performed manually and has three main problems: i) doctors need to spend their time in the administrative task (scheduling); ii) usually under or over utilization of the ORs is obtained; and iii) objectivity can be questioned due to the scheduling is made by humans. This chapter presents a *Decision Support System* (DSS), that using a *specific heuristic algorithm* SHA, helps the hospital managers in the scheduling of elective patients. In addition to schedule patients, the DSS also includes features enabling the management of the medical teams and the OR time-table, updating the waiting list of patients, iterative planning and automatic improvement of the input data. In collaboration with the Orthopedic department in the "Lozano Blesa" hospital (HCU) in Zaragoza a software tool (CIPLAN) based on the proposed DSS has been developed. The preferences and concerns of doctors have been considered not only at the surgery scheduling level but also at the user interface level. So, a friendly interface divided in six main panels is proposed in CIPLAN. It has been developed in Java language and use a SQL database. Moreover in order to provide a safe environment to work and better security of the medical data, different access level are provided to every user.

In order to check the impact of the software tool in the efficiency and the quality of the surgical service a case study considering real data in the Orthopedic Department of the HCU has been discussed. The results show that using CIPLAN an improving in the occupation rate of the ORs of 2.53% can be reached. This means a saving of 128,000 euros yearly only considering the Orthopedic department.

However, it has been observed that there are significant differences in the surgery duration depending on the experience and on the surgery skill of the surgeon. So, if the SHA works with general average duration, over and under real occupation rate are obtained more frequently than working with customized data. It would be convenient to customize the data of the surgical durations depending on each surgeon. In this way, the expected occupation rate and the expected ending time will be more similar to the real ones improving the quality of the service.

Conclusions

The management of healthcare systems is a complex task due to its size, the huge number of agents involved and their different expectations. However, it could be alleviated by the use of technology. In this thesis, new methods and formal models for healthcare system management are proposed. These new methods allow to prevent system failures and on the other hand, to improve the quality and efficiency of the hospitals.

The first part of the thesis deals with the modeling and analysis of hospitals by the use of clinical pathways. In [6] a methodology to model healthcare systems using *Stochastic Well-formed Nets* (SWN) is proposed. SWNs is a class of colored Petri Nets that allows catching the multifaceted nature of the system. However, due to the size and structure of the models, state-based techniques and event simulations are hard to be used in the analysis of the system.

In Chapter 2 we propose a methodology based on a set of relaxations, abstractions and modifications for obtaining from the SWN model two different facets of the system where formal subclasses of Petri Nets are used. The main advantage of these subclasses is that structural techniques can be used to analyze different properties of the system. However, the decolorizing of the SWN model implies the loss of synchronization between patients and their customized information. The first facet (*resource management facet*) it is modeled by S⁴PR subnets and considers the sharing resource of the clinical pathways. The second one (*handshake between clinical pathways*) uses *Deterministically Synchronized Sequential Process* (DSSP) and considers the asynchronous communication between clinical pathways.

A desirable property is the liveness of the system. In healthcare systems, liveness implies that a patient cannot be trapped in the middle of a clinical pathway. This situation could happen for example due to a badly designed communication protocol. In Chapter 3 it is proposed a very easy and intuitive deadlock prevention method in DSSP models. It is based on buffers pre-assignment. Considering healthcare systems, this means that all information required for following a clinical pathway must be available in the buffers before that a patient start it. This is an hyper-constraint that is far along of a “maximally permissive” approach. So, future questions that could improve the method are: is it necessary to pre-assign all buffers? Which is the minimum set of buffers that must be pre-assigned to ensure the liveness? Pre-assignment method is easy to be applied, however, it only works in some particular structure of DSSP.

Chapter 4 provides a general method for liveness enforcement of DSSP. It is based on computing a control PN from the original DSSP system. This control PN has a predefined type of structure and evolves synchronously with the original DSSP. Through guards expressions, the state of the control PN prevents the firing of some transition in the original DSSP ensuring that deadlock states will not be reached.

So if a DSSP modeling a healthcare system is not live due to a badly designed communication protocol, depending on the structure of the DSSP, it could be easily solved using the buffer pre-assignment method. However, if the structure of the DSSP is not appropriate for applying buffer pre-assignment, the general approach can be used to prevent the deadlock of the net. This general approach ensures the liveness in any DSSP structure. Moreover, this second method is more permissible than the buffers pre-assignment approach. It happens because a clinical pathway can start despite its input buffers are empty if for sure there will be enough tokens in a future marking.

The second part of the thesis has to do with the scheduling of patients in surgery services. In Chapter 5 the scheduling of elective patients under *Operation Room* (OR) block booking has been approached. Three scheduling criteria have been established:

- C1 *optimize the use of the ORs*
- C2 *the total available time in a surgical block should not be exceeded*

- C3 respect as much as possible the order of the patients in the waiting list

Considering the average duration of the surgeries, 3 scheduling problems that try to obtain a given occupation rate of the blocks are proposed. These are combinatorial problems with high computational complexity, so different heuristics have been considered. The fastest and best solutions are obtained by solving the *Mixed Integer Linear Programming* (MILP) problem iteratively, that is, using *Receding Horizon Strategy* (RHS). It is possible to schedule 22 ORs in less than 16 [s]. The average occupation rate is the closest to the target valued (around 0,2% lower) and the lowest standard deviation of the occupation rate is obtained. Moreover, the scheduling that more respect the order of the patients is obtained.

However, the proposed MILP could be not robust enough because it uses only the average duration of the surgeries. This uncertainty could result in over/under utilization of some blocks. Moreover, the target occupation rate is an input parameter of the problem that could be difficult to select for inexperienced surgeons.

In order to overcome these problems, in Chapter 5, a *New-Mixed Integer Quadratic Constraint Programming* (N-MIQCP) problem is proposed. The N-MIQCP problem considers that the different durations in a block (surgery time, cleaning time, etc) follow a normal distribution. These assumptions allow us to introduce some chance constraints that impose a *Minimum Confidence Level* (Cl) not exceeding the available time. Moreover, in the N-MIQCP problem, the objective is to maximize the occupation rate keeping the chance constraints. In this way, it is avoided the task of selecting an appropriated target occupation rate.

The N-MIQCP problem is computationally very complex being only possible to solve small instances of 3 blocks in a reasonable time. For this reason, two heuristics methods have been proposed and compared. The better schedulings are obtained by using the *Specific Heuristic Algorithm* (SHA). Considering the *Orthopedic Surgery Department* (OSD) in the “Lozano Blesa” Hospital, a one-year scheduling simulation has been used to compare the heuristic approach with other approaches in bibliography. The results show that similar occupation rates and similar confidence levels are obtained, however, our approach respect much more the order of the patients in the waiting list. As future work,

it is considered the inclusion of some hospital resources to take into account in the scheduling of surgeries (e.g. time postoperative bed).

A 3 steps approach for the combined scheduling of elective and urgent patients is proposed in Chapter 6. In the first step, the elective patients are scheduled assuming a target elective surgery time. In the second step, the urgent patients are scheduled in the remaining time. Finally, in the last step, elective and urgent patients are sequenced. Different policies of time reserved for elective and urgent patients are evaluated. The results show that all ORs must be used to perform elective and urgent surgeries instead of reserving some ORs exclusively for one type of patients.

Finally, in Chapter 7, a *Decision Support System* (DSS) for managing of surgical departments is proposed. The main goal of the DSS is the scheduling of elective patients by using the SHA. Moreover, other useful features for the management of the department are included. The DSS has been implemented in a software tool called CIPLAN. This software tool has a friendly user interface which has been developed in collaboration with medical doctors in the OSD of the LBH. The benefits of the tool have been tested in a real case study considering the OSD in the LBH. A free increment of 22 mornings (from 8:30 to 15:00) time blocks per year is obtained by using the tool. This means to perform around 70 patients more per year or saving of 128.000 Euros. The result could be better extending the use of the tool to other departments in the hospital. An interesting improvement in the tool is the inclusion of urgent surgery scheduling. This will allow complete management of surgeries services without the necessity of other complementary scheduling.

Bibliography

- [1] *OECD, health data 2005 - statistics and indicators for 30 countries.* 2005. [cited at p. 96]
- [2] Z.Y. Abdelrasol, N. Harraz, and A. Eltawil. A proposed solution framework for the operating room scheduling problems. In *Proceedings of the world congress on engineering and computer science*, volume 2, pages 23–25, 2013. [cited at p. 97]
- [3] G. Amodio, M.-P. Fanti, L. Martino, A. Mangini, and W. Ukovich. A Petri Net Model for Performance Evaluation and Management of an Emergency Cardiology Departments. In *Proc. of the XXXV ORHAS conference*, Leuven, Belgium, 2009. [cited at p. 97]
- [4] H. Arabnejad and J. G. Barbosa. List scheduling algorithm for heterogeneous systems by an optimistic cost table. *IEEE Transactions on Parallel and Distributed Systems*, 25(3):682–694, March 2014. [cited at p. 125, 126]
- [5] S. Bernardi, J. Albareda, J.-M. Colom, and C. Mahulea. A model-based Approach for the Specification and Verification of Clinical Guidelines. In *First Workshop on Models and Methods for Hospital Management and Planning held in conjunction with ETFA'2014: 19th IEEE International Conference on Emerging Technologies and Factory Automation*, Barcelona, Spain, 2014. [cited at p. 97]
- [6] S. Bernardi, C. Mahulea, and J. Albareda. Toward a decision support system for the clinical pathways assessment. *Discrete Event Dynamic Systems: Theory and Applications*, 2019. in press. DOI:[10.1007/s10626-019-00279-9]. [cited at p. 2, 11, 18, 20, 21, 25, 28, 199]

- [7] J. Bowers and Gillian M. Managing uncertainty in orthopaedic trauma theatres. *European Journal of Operational Research*, 154(3):599 – 608, 2004. [cited at p. 142]
- [8] E.F. Camacho and C. Bordons. *Model Predictive Control*. Advanced Textbooks in Control and Signal Processing. Springer London, 2004. [cited at p. 108, 125]
- [9] E. Cano, C.A. Rovetto, and J.M. Colom. An algorithm to compute the minimal siphons in S⁴PR nets. *Discrete Event Dynamic Systems*, 22(4):403–428, 2012. [cited at p. 42]
- [10] B. Cardoen, E. Demeulemeester, and J. Belien. Operating room planning and scheduling: A literature review. *European Journal of Operational Research*, 201(3):921 – 932, 2010. [cited at p. 4, 97, 140]
- [11] E. Cela. *The quadratic assignment problem*. Combinatorial Optimization. Springer, 1998. [cited at p. 103, 107, 109]
- [12] G. Chiola, C. Dutheillet, G. Franceschinis, and S. Haddad. Stochastic well-formed colored nets and symmetric modeling applications. *IEEE Transactions on Computers*, 42(11):1343–1360, 1993. [cited at p. 2, 20, 21]
- [13] Y.M. Chou, A.M. Polansky, and R.L. Mason. Transforming non-normal data to normality in statistical process control. *Journal of Quality Technology*, 30(2):133–141, 4 1998. [cited at p. 154]
- [14] C.I.A. The world factbook., 2017. [cited at p. 96]
- [15] D. Clavel, D. Botez, C. Mahulea, and J. Albareda. Software Tool for Operating Room Scheduling in a Spanish Hospital Department. In *22th International Conference on System Theory, Control and Computing (ICSTCC)*, Sinaia, Romania, 2018. [cited at p. 6, 95, 161]
- [16] D. Clavel, C. Mahulea, J. Albareda, and M. Silva. A decision support system for elective surgery scheduling under uncertain durations. *Submitted for publication on IBM Medical Informatics and Decision Making*. [cited at p. 6, 95, 161]
- [17] D. Clavel, C. Mahulea, J. Albareda, and M. Silva. Towards Efficient Algorithms for Planning Surgeries in Operation Rooms. In *23st IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, September 2018. [cited at p. 6, 95]

- [18] D. Clavel, C. Mahulea, and M. Silva. Enforcing liveness in petri nets based distributed systems. *Submitted for publication on IEEE Transactions on Automatic Control*. [cited at p. 6, 61]
- [19] D. Clavel, C. Mahulea, and M. Silva. On Liveness Enforcement of DSSP net systems. In *52nd IEEE Annual Conference on Decision and Control (CDC)*, Las Vegas, Nevada, Estados Unidos, December 2016. [cited at p. 6, 40, 41]
- [20] D. Clavel, C. Mahulea, and M. Silva. From Healthcare System Specifications to Formal Models. In *IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Bari, Italy, October 2019. [cited at p. 6, 27]
- [21] D. Clavel, C. Mahulea, M. Silva, and J. Albareda. Operation Planning of Elective Patients in an Orthopedic Surgery Department. In *21st IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, September 2016. [cited at p. 6, 95, 104]
- [22] D. Clavel, X. Xie, C. Mahulea, and M. Silva. A three steps approach for surgery planning of elective and urgent patients. *IFAC-PapersOnLine*, 51(7):243–250, 2018. [cited at p. 6, 139]
- [23] J. M. Colom. The resource allocation problem in flexible manufacturing systems. In W.v.d. Aalst and E. Best, editors, *Applications and Theory of Petri Nets*, volume 2679 of *Lecture Notes in Computer Science*, pages 23–35. Springer-Verlag, Berlin, Heidelberg, 2003. [cited at p. 42]
- [24] R. David and H. Alla. Discrete, continuous, and hybrid petri nets. *Control Systems Magazine, IEEE*, 28:81–84, 07 2008. [cited at p. 12]
- [25] F. Dicesare, G. Harhalakis, J.M Proth, M. Silva, and F. Vernadat. *Practice of Petri Nets in Manufacturing*, volume 45. Chapman and Hall, 01 1993. [cited at p. 11, 12, 209]
- [26] M. Dios, J.-M. Molina-Pariente, V. Fernandez-Viagas, J.-L. Andrade-Pineda, and J.-M. Framinan. A decision support system for operating room scheduling. *Computers & Industrial Engineering*, 88:430 – 443, 2015. [cited at p. 162]
- [27] M. Dotoli, M.-P. Fanti, A. Mangini, and W. Ukovich. A continuous Petri net model for the management and design of emergency cardiology departments. In *ADHS'2019: 3rd IFAC Conference on Analysis and Design of Hybrid Systems*, Zaragoza, Spain, 2009. [cited at p. 97]

- [28] D.A. Etzioni, J.H. Liu, M.A. Maggard, and C.Y. Ko. The aging population and its impact on the surgery workforce. *Annals of surgery*, 238(2):170, 2003. [cited at p. 3]
- [29] J. Ezpeleta, J. M. Colom, and J. Martinez. A Petri net based deadlock prevention policy for flexible manufacturing systems. *IEEE Transactions on Robotics and Automation*, 11(2):173–184, April 1995. [cited at p. 42]
- [30] J. Gearhart, K. Adair, R. Detry, J. Durfee, K.Jones, and N. Martin. Comparison of open-source linear programming solvers. Technical Report SAND2013-8847, Sandia National Laboratories, Albuquerque, California, October 2013. [cited at p. 110]
- [31] G. Giachetti, B. Marín, and O. Pastor. Using UML as a Domain-Specific Modeling Language: A Proposal for Automatic Generation of UML Profiles. In Pascal van Eck, Jaap Gordijn, and Roel Wieringa, editors, *Advanced Information Systems Engineering*, pages 110–124. Springer Berlin Heidelberg, 2009. [cited at p. 19]
- [32] A. Giua, F. DiCesare, and M. Silva. Generalized mutual exclusion constraints on nets with uncontrollable transitions. In *IEEE International Conference on Systems, Man and Cybernetics*, pages 974–979. IEEE, 1992. [cited at p. 43]
- [33] S. Glouberman and H. Mintzberg. Managing the care of health and the cure of diseasepart i: Differentiation. *Health care management review*, 26(1):56–69, 2001. [cited at p. 3]
- [34] E. Hans, G. Wullink, M. Van Houdenhoven, and G. Kazemier. Robust surgery loading. *European Journal of Operational Research*, 185(3):1038–1050, 2008. [cited at p. 5, 121, 134, 135]
- [35] J. H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, 1975. [cited at p. 107]
- [36] IBM. *IBM ILOG CPLEX Optimization Studio. Software*. 2016. [cited at p. 110]
- [37] Institute of Medicine. *Guidelines for Clinical Practice: From Development to Use*. The National Academies Press, Washington, DC, 1992. [cited at p. 28]
- [38] R.L. Jackson. The business of surgery. Managing the OR as a profit center requires more than just IT. It requires a profit-making mindset, too. *Health Management Technology*, 23(7):20–22, 2002. [cited at p. 96]

- [39] D.-S. Johnson. *Near-optimal bin packing algorithms*. PhD thesis, Massachusetts Institute of Technology, 1973. [cited at p. 125, 126]
- [40] N. Karabakal. *A C Code for Solving the Generalized Assignment Problem*. Number MI 48109-2117. University of Michigan, Ann Arbor, May 1992. [cited at p. 113]
- [41] N. Karabakal and J. Bean. *A steepest descent multiplier adjustment method for the generalized assignment problem*. Univ. of Michigan, Ann Arbor, MI (United States), Dec 1994. [cited at p. 109]
- [42] M. Lamiri, X. Xie, A. Dolgui, and F. Grimaud. A stochastic model for operating room planning with elective and emergency demand for surgery. *European Journal of Operational Research*, 185(3):1026–1037, 2008. [cited at p. 142]
- [43] M. Lamiri, X. Xie, and S. Zhanga. Column generation approach to operating theater planning with elective and emergency patients. *IIE Transactions*, 40(9):838–852, 2008. [cited at p. 142]
- [44] E. Lanzarone, A. Matta, and E. Sahin. Operations management applied to home care services: The problem of assigning human resources to patients. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 42(6):1346–1363, 2012. [cited at p. 97]
- [45] E. Lanzarone, A. Matta, and G. Scaccabarozzi. A patient stochastic model to support human resource planning in home care. *Production Planning & Control*, 21(1):3–25, 2010. [cited at p. 97]
- [46] Z. Li and M. Zhou. Elementary siphons of Petri nets and their application to deadlock prevention in flexible manufacturing systems. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 34(1):38–51, Jan 2004. [cited at p. 42]
- [47] A. Macario, T.S. Vitez, B. Dunn, and T. McDonald. Where are the costs in perioperative care?: Analysis of hospital costs and charges for inpatient surgical care. *Anesthesiology*, 83(6):1138–1144, 1995. [cited at p. 96]
- [48] C. Mahulea, L. Mahulea, J.-M. Garcia-Soriano, and J.-M. Colom. Petri Nets with Resources for Modeling Primary Healthcare Systems. In *ICSTCC'2014: 18th International Conference on System Theory, Control and Computing*, Sinaia, Romania, 2014. [cited at p. 97]

- [49] M.A. Marsan, G. Balbo, G. Conte, S. Donatelli, and G. Franceschinis. *Modelling with Generalized Stochastic Petri Nets*. John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 1994. [cited at p. 12]
- [50] T. Mens and P. Van Gorp. A taxonomy of model transformation. *Electronic Notes in Theoretical Computer Science*, 152:125 – 142, 2006. Proceedings of the International Workshop on Graph and Model Transformation (GraMoT 2005). [cited at p. 21]
- [51] T. Murata. Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, 77(4):541–580, 1989. [cited at p. 71]
- [52] M. Gourgand N. Klement, N. Grangeon. Medical Imaging : Exams Planning and Resource Assignment: Hybridization of a Metaheuristic and a List Algorithm. In *10th International Joint Conference on Biomedical Engineering Systems and Technologies (BIOSTEC 2017)*, number 5, pages 260–267, Porto, Germany, February 2017. [cited at p. 125, 126]
- [53] J. Park and S.A. Reveliotis. Deadlock avoidance in sequential resource allocation systems with multiple resource acquisitions and flexible routings. *IEEE Transactions on Automatic Control*, 46(10):1572–1583, Oct 2001. [cited at p. 42]
- [54] L. Parrilla, J. Garca, J. Albareda, and C. Mahulea. Heat: A tool to develop, analyze and monitor clinical pathways. In *2017 IEEE 14th International Conference on Networking, Sensing and Control (ICNSC)*, pages 186–191, May 2017. [cited at p. 167]
- [55] L. Recalde, E. Teruel, and M. Silva. On well-formedness analysis: The case of deterministic systems of sequential processes. In *Structures in Concurrency Theory*, pages 279–293. Springer, 1995. [cited at p. 17, 42]
- [56] L. Recalde, E. Teruel, and M. Silva. Modeling and analysis of sequential processes that cooperate through buffers. *IEEE Trans. Robotics and Automation*, 14(2):267 – 277, 1998. [cited at p. 2, 11, 16, 42]
- [57] s. Baair, M. Beccuti, D. Cerotti, M. De Pierro, S. Donatelli, and G. Franceschinis. The greatspn tool: Recent enhancements. *SIGMETRICS Perform. Eval. Rev.*, 36(4):4–9, March 2009. [cited at p. 28]
- [58] S. Sahni and T. Gonzalez. P-complete approximation problems. *Journal of the ACM (JACM)*, 23:255 – 265, 1976. [cited at p. 109]

- [59] B. Selic. A Systematic Approach to Domain-Specific Language Design Using UML. In *10th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC'07)*, pages 2–9, May 2007. [cited at p. 19]
- [60] B. Selic, C. Bock, S. Cook, P. Rivett, T. Rutt, E. Seidewitz, and D. Tolbert. Omg unified modeling language (version 2.5). Technical report, Malina Software Corp., Ottawa, Canada, 03 2015. [cited at p. 19]
- [61] D.B. Shmoys and E. Tardos. An approximation algorithm for the generalized assignment problem. *Mathematical Programming*, 62(1):461–474, 1993. [cited at p. 109]
- [62] O.V. Shylo, O.A. Prokopyev, and A.J. Schaefer. Stochastic operating room scheduling for high-volume specialties under block booking. *INFORMS Journal on Computing*, 25(4):682–692, 2012. [cited at p. 5, 121, 134, 135]
- [63] M. Silva. *Las Redes de Petri : en la Automtica y la Informtica 1 ed.* AC, 1985. [cited at p. 12]
- [64] M. Silva. Introducing petri nets. In *Practice of Petri Nets in manufacturing* [25], pages 1–62. [cited at p. 71, 75]
- [65] M. Silva, E. Teruel, and J.-M. Colom. Linear algebraic and linear programming techniques for the analysis of P/T net systems. *Lecture on Petri Nets I: Basic Models*, 1491:309–373, 1998. [cited at p. 17, 75]
- [66] E. Teruel and M. Silva. Structure theory of equal conflict systems. *Theoret. Comput. Sci.*, 153(1 and 2):271–300, 1996. [cited at p. 11, 17, 42]
- [67] F. Tricas. *Deadlock Analysis, Prevention and Avoidance in Sequential Resource Allocation Systems*. PhD thesis, University of Zaragoza, 05 2003. [cited at p. 33]
- [68] F. Tricas, F. Garcia-Valles, J. M. Colom, and J. Ezpeleta. A Petri Net Structure-Based Deadlock Prevention Solution for Sequential Resource Allocation Systems. In *IEEE International Conference on Robotics and Automation*, pages 271–277, April 2005. [cited at p. 2, 11, 14]
- [69] J.T. van Essen, E.W. Hans, J.L. Hurink, and A. Oversberg. Minimizing the waiting time for emergency surgery. *Operations Research for Health Care*, 1(2):34 – 44, 2012. [cited at p. 150, 152]

- [70] S. Wanderer, D. Stavrou, G. Lypowy, C. Latter, and C. Caudle. The guide architecture for implementation of clinical practice guidelines. [cited at p. 167]
- [71] S. Wang, D. You, and M. Zhou. A Necessary and Sufficient Condition for a Resource Subset to Generate a Strict Minimal Siphon in S 4PR. *IEEE Transactions on Automatic Control*, 62(8):4173–4179, Aug 2017. [cited at p. 42]
- [72] L. Wells. Performance Analysis Using CPN Tools. In *Proceedings of the 1st International Conference on Performance Evaluation Methodologies and Tools*, valuetools '06, New York, NY, USA, 2006. ACM. [cited at p. 28]
- [73] A. Zimmermann. Modeling and evaluation of stochastic Petri Nets with TimeNET 4.1. In *6th International ICST Conference on Performance Evaluation Methodologies and Tools*, pages 54–63, Oct 2012. [cited at p. 28]

Appendices

Appendix A

Numerical results from the scheduling case application

In this appendix, all data related with the scheduling case application performed in Sec. 7.3 are given. Particularly, the waiting list of patients is given in Fig.A.1, Fig. A.2 and Fig.A.3. The first column (#) of the tables in these figures represents the preference order of the patients, the second (Avg.) and third (Std.) columns indicates the average duration and the standard deviation of their surgery. Finally the last column (real) shows the real duration of the surgeries.

The manual scheduling of each block is shown in Fig. A.4, Fig. A.6 and Fig. A.8 while the automatic scheduling is given in Fig. A.5, Fig. A.7 and Fig.A.9. Each row in these tables represent a block. The type of block (morning or afternoon), the preference order of the patients scheduled, the expected and real occupation rate, the ending time and the confidence level of not exceeding available time are given.

Waiting list			
#	Avg.	Std.	Real
1	182,5	40,3	205
2	128,1	24,5	120
3	128,1	24,5	160
4	128,1	24,5	100
5	113,3	29,6	145
6	128,1	24,5	155
7	217,0	47,0	215
8	77,9	18,0	85
9	128,1	24,5	120
10	128,1	24,5	160
11	127,0	28,2	170
12	99,1	21,4	115
13	160,0	30,0	160
14	113,3	29,6	110
15	128,1	24,5	110
16	128,1	24,5	140
17	128,1	24,5	88
18	128,1	24,5	93
19	128,1	24,5	106
20	127,0	28,2	165
21	77,9	18,0	95
22	77,9	18,0	72
23	182,5	40,3	215
24	53,8	7,5	65
25	20,8	8,0	35
26	113,3	29,6	118
27	113,3	29,6	105
28	128,1	24,5	130
29	128,1	24,5	130
30	33,8	9,9	30
31	77,9	18,0	75
32	77,9	18,0	60
33	77,9	18,0	60
34	33,8	9,9	35
35	182,5	40,3	125
36	127,0	28,2	125
37	32,5	7,0	35
38	128,1	24,5	110
39	128,1	24,5	110
40	33,8	9,9	25
41	113,3	29,6	105
42	99,1	21,4	80
43	128,1	24,5	110
44	123,8	21,9	90
45	77,9	18,0	95
46	113,3	29,6	110
47	99,1	21,4	80
48	65,0	17,0	50
49	40,0	10,0	40
50	99,1	21,4	120

Waiting list			
#	Avg.	Std.	Real
51	99,1	21,4	100
52	99,1	21,4	90
53	103,0	21,0	135
54	103,0	21,0	120
55	46,6	19,2	55
56	15,0	3,0	15
57	46,6	19,2	45
58	113,3	29,6	145
59	113,3	29,6	90
60	128,1	24,5	120
61	128,1	24,5	150
62	50,5	17,5	25
63	77,9	18,0	85
64	77,9	18,0	90
65	75,5	18,2	80
66	50,5	17,5	55
67	64,0	17,0	65
68	99,1	21,4	85
69	33,8	9,9	45
70	46,6	19,2	35
71	128,1	24,5	110
72	99,1	21,4	84
73	100,0	28,0	70
74	99,1	21,4	115
75	99,1	21,4	105
76	113,3	29,6	90
77	113,3	29,6	115
78	99,1	21,4	95
79	128,1	24,5	165
80	46,6	19,2	40
81	127,0	28,2	100
82	128,1	24,5	105
83	77,9	18,0	125
84	77,9	18,0	60
85	99,1	21,4	90
86	50,5	17,5	65
87	127,0	28,2	155
88	77,9	18,0	75
89	77,9	18,0	60
90	77,9	18,0	65
91	77,9	18,0	85
92	50,0	10,0	50
93	40,0	10,0	40
94	127,0	28,2	95
95	128,1	24,5	120
96	127,0	28,2	155
97	127,0	28,2	60
98	46,6	19,2	105
99	77,0	29,0	60
100	33,8	9,9	25

Waiting list			
#	Avg.	Std.	Real
101	65,0	26,0	60
102	128,1	24,5	140
103	128,1	24,5	140
104	217,0	47,0	250
105	33,8	9,9	35
106	99,1	21,4	95
107	99,1	21,4	75
108	46,6	19,2	25
109	75,5	18,2	60
110	127,0	28,2	125
111	127,0	28,2	145
112	182,5	40,3	185
113	127,0	28,2	105
114	46,6	19,2	60
115	128,1	24,5	110
116	99,1	21,4	85
117	217,0	47,0	240
118	75,5	18,2	80
119	128,1	24,5	130
120	128,1	24,5	120
121	33,8	9,9	25
122	20,8	8,0	15
123	56,0	17,0	55
124	53,8	7,5	50
125	50,5	17,5	50
126	53,8	7,5	50
127	33,8	9,9	45
128	99,1	21,4	85
129	99,1	21,4	100
130	113,3	29,6	115
131	123,8	21,9	115
132	217,0	47,0	195
133	77,9	18,0	60
134	123,8	21,9	140
135	123,8	21,9	175
136	128,1	24,5	120
137	113,3	29,6	80
138	127,0	28,2	165
139	127,0	28,2	115
140	53,8	7,5	50
141	128,1	24,5	105
142	128,1	24,5	105
143	99,1	21,4	105
144	33,8	9,9	40
145	46,6	19,2	40
146	46,6	19,2	35
147	127,0	28,2	135
148	128,1	24,5	150
149	33,8	9,9	35
150	75,5	18,2	105

Figure A.1: Waiting list: patients from 1-150

Waiting list			
#	Avg.	Std.	Real
151	50,5	17,5	75
152	77,0	29,0	60
153	46,6	19,2	35
154	128,1	24,5	130
155	99,1	21,4	100
156	127,0	28,2	85
157	128,1	24,5	125
158	127,0	28,2	115
159	107,0	25,0	105
160	75,5	18,2	60
161	99,1	21,4	160
162	56,0	17,0	40
163	53,0	17,0	40
164	99,1	21,4	105
165	113,3	29,6	70
166	77,9	18,0	90
167	46,6	19,2	60
168	184,0	26,0	180
169	120,0	10,0	160
170	33,8	9,9	25
171	217,0	47,0	175
172	99,1	21,4	105
173	99,1	21,4	105
174	77,9	18,0	100
175	99,1	21,4	115
176	46,6	19,2	35
177	128,1	24,5	155
178	99,1	21,4	90
179	126,0	37,0	95
180	123,8	21,9	90
181	113,3	29,6	125
182	113,3	29,6	160
183	113,3	29,6	145
184	113,3	29,6	130
185	33,8	9,9	30
186	128,1	24,5	100
187	128,1	24,5	95
188	127,0	28,2	105
189	115,8	27,5	125
190	123,8	21,9	120
191	103,0	21,0	125
192	65,0	17,0	40
193	113,3	29,6	120
194	113,3	29,6	105
195	127,0	28,2	170
196	127,0	28,2	100
197	65,0	17,0	50
198	127,0	28,2	185
199	115,8	27,5	110
200	113,3	29,6	80

Waiting list			
#	Avg.	Std.	Real
201	99,1	21,4	75
202	128,1	24,5	140
203	33,8	9,9	55
204	20,8	8,0	20
205	123,8	21,9	95
206	77,9	18,0	55
207	77,9	18,0	60
208	127,0	28,2	120
209	115,8	27,5	170
210	128,1	24,5	130
211	128,1	24,5	120
212	33,8	9,9	30
213	128,1	24,5	85
214	128,1	24,5	100
215	128,1	24,5	100
216	128,1	24,5	190
217	128,1	24,5	130
218	113,3	29,6	145
219	151,0	42,0	105
220	77,9	18,0	60
221	127,0	28,2	150
222	99,1	21,4	100
223	127,0	28,2	130
224	127,0	28,2	155
225	68,0	27,0	55
226	99,1	21,4	165
227	120,0	30,0	120
228	33,8	9,9	10
229	128,1	24,5	180
230	128,1	24,5	140
231	68,0	27,0	35
232	113,3	29,6	125
233	99,1	21,4	70
234	113,3	29,6	130
235	113,3	29,6	70
236	128,1	24,5	125
237	128,1	24,5	115
238	128,1	24,5	135
239	103,0	17,0	100
240	56,0	17,0	65
241	127,0	28,2	115
242	127,0	28,2	120
243	33,8	9,9	20
244	128,1	24,5	130
245	113,3	29,6	160
246	127,0	28,2	160
247	112,0	27,0	115
248	128,1	24,5	150
249	60,0	20,0	90
250	77,9	18,0	60

Waiting list			
#	Avg.	Std.	Real
251	128,1	24,5	160
252	128,1	24,5	170
253	217,0	47,0	240
254	128,1	24,5	120
255	99,1	21,4	90
256	99,1	21,4	130
257	99,1	21,4	105
258	123,8	21,9	115
259	77,9	18,0	65
260	77,9	18,0	75
261	217,0	47,0	240
262	115,8	27,5	90
263	127,0	28,2	95
264	127,0	28,2	95
265	113,3	29,6	75
266	99,1	21,4	80
267	217,0	47,0	215
268	77,9	18,0	70
269	217,0	47,0	220
270	99,1	21,4	110
271	50,5	17,5	50
272	56,0	17,0	50
273	33,8	9,9	30
274	127,0	28,2	105
275	127,0	28,2	110
276	184,0	26,0	230
277	184,0	26,0	130
278	128,1	24,5	120
279	123,8	21,9	140
280	77,9	18,0	95
281	128,1	24,5	120
282	99,1	21,4	115
283	127,0	28,2	90
284	128,1	24,5	120
285	217,0	47,0	220
286	128,1	24,5	175
287	128,1	24,5	125
288	113,3	29,6	70
289	113,3	29,6	75
290	127,0	28,2	105
291	127,0	28,2	160
292	33,8	9,9	30
293	112,0	27,0	90
294	127,0	28,2	120
295	115,8	27,5	140
296	115,8	27,5	140
297	74,0	28,0	65
298	50,5	17,5	45
299	128,1	24,5	170
300	113,3	29,6	135

Figure A.2: Waiting list: patients from 151-300

Waiting list				Waiting list			
#	Avg.	Std.	Real	#	Avg.	Std.	Real
301	75,0	22,0	45	351	217,0	47,0	315
302	113,3	29,6	75	352	20,8	8,0	25
303	75,5	18,2	75	353	128,1	24,5	115
304	33,8	9,9	30	354	128,1	24,5	105
305	113,3	29,6	160	355	46,6	19,2	50
306	72,0	18,0	85	356	217,0	47,0	190
307	92,0	31,0	60	357	128,1	24,5	120
308	217,0	47,0	215	358	123,8	21,9	110
309	217,0	47,0	235	359	123,8	21,9	135
310	113,3	29,6	105	360	77,9	18,0	65
311	123,8	21,9	130	361	68,0	19,0	55
312	75,5	18,2	85	362	53,8	7,5	45
313	75,5	18,2	50	363	127,0	28,2	125
314	97,2	17,2	75	364	128,1	24,5	90
315	33,8	9,9	35	365	115,8	27,5	130
316	128,1	24,5	155	366	115,8	27,5	100
317	97,2	17,2	85	367	115,8	27,5	150
318	128,1	24,5	100	368	115,8	27,5	100
319	113,3	29,6	85	369	127,0	28,2	120
320	128,1	24,5	180	370	97,2	17,2	100
321	128,1	24,5	145	371	113,3	29,6	125
322	128,1	24,5	128	372	123,8	21,9	130
323	123,8	21,9	140	373	33,8	9,9	40
324	75,0	22,0	40	374	97,2	17,2	135
325	128,1	24,5	95	375	77,9	18,0	90
326	128,1	24,5	100	376	46,6	19,2	35
327	20,8	8,0	15	377	115,8	27,5	95
328	128,1	24,5	125	378	184,0	26,0	185
329	97,2	17,2	100	379	128,1	24,5	150
330	65,0	17,0	60	380	128,1	24,5	120
331	217,0	47,0	95	381	65,0	10,0	65
332	128,1	24,5	185	382	123,8	21,9	135
333	75,0	22,0	90	383	113,3	29,6	140
334	33,8	9,9	45	384	128,1	24,5	145
335	32,5	7,0	30	385	63,0	17,0	75
336	182,5	40,3	150	386	97,2	17,2	100
337	128,1	24,5	145	387	127,0	28,2	120
338	128,1	24,5	120	388	97,2	17,2	90
339	50,5	17,5	45	389	127,0	28,2	150
340	113,3	29,6	75	390	127,0	28,2	110
341	75,5	18,2	60	391	184,0	26,0	150
342	50,5	17,5	50	392	113,3	29,6	150
343	128,1	24,5	120	393	144,0	44,0	65
344	97,2	17,2	105	394	77,9	18,0	75
345	33,8	9,9	50	395	77,9	18,0	125
346	113,3	29,6	90	396	145,0	38,0	53
347	97,2	17,2	85	397	33,8	9,9	40
348	113,3	29,6	165				
349	123,8	21,9	120				
350	50,5	17,5	20				

Figure A.3: Waiting list: patients from 301-397

Manual Scheduling

Block	Type	Patients scheduled					Real Ocu.	End Time	Expected Ocu.	Conf. Level
		1	2	3	4	5				
1	M	1	2				83,3%	14:20	79,7%	84,0%
2	A	3	4				86,7%	20:25	85,3%	64,6%
3	M	5	6				76,9%	14:10	61,8%	99,8%
4	M	7	8				76,9%	14:05	75,6%	95,7%
5	A	9	10				93,3%	20:15	85,3%	64,6%
6	A	11	12				95,0%	20:30	75,3%	87,5%
7	M	13	14				69,2%	13:50	70,0%	97,3%
8	A	15	16				83,3%	20:05	85,3%	64,6%
9	M	17	18	19			73,6%	13:52	98,5%	16,7%
10	M	20	21	22			85,1%	14:22	72,6%	91,2%
11	M	23	24	25			80,8%	14:20	66,2%	96,4%
12	M	26	27				61,9%	13:00	62,8%	98,9%
13	M	28	29	30			74,4%	14:00	74,4%	89,5%
14	M	31	32	33	34		59,0%	13:35	68,7%	90,9%
15	M	35	36	37			73,1%	14:10	87,9%	47,7%
16	M	38	39	40			62,8%	13:45	74,4%	89,5%
17	A	41	42				61,7%	19:10	70,7%	92,7%
18	M	43	44	45			75,6%	14:05	84,6%	59,5%
19	M	46	47	48	49		71,8%	15:15	81,3%	52,6%
20	M	50	51	52			79,5%	14:10	76,2%	85,4%
21	M	53	54	55	56		83,3%	14:45	68,7%	89,7%
22	M	57	58	59			77,8%	14:10	75,8%	77,0%
23	M	60	61	62			75,6%	14:10	78,5%	78,7%
24	M	63	64	65	66		79,5%	14:30	72,3%	81,9%
25	A	67	68	69	70		76,7%	20:10	81,3%	36,5%
26	M	71	72	73			67,7%	13:15	83,8%	61,1%
27	A	74	75				73,3%	19:30	66,0%	98,4%
28	A	76	77				68,3%	19:10	75,3%	83,5%
29	M	78	79	80			76,9%	14:30	70,3%	94,4%
30	A	81	82				68,3%	19:10	85,0%	64,6%
31	M	83	84	85	86		87,2%	14:50	78,2%	63,6%
32	M	87	88	89			74,4%	14:10	72,6%	91,2%
33	M	90	91	92	93		61,5%	13:15	63,1%	98,0%
34	A	94	95				71,7%	19:30	85,0%	64,6%
35	M	96	97	98			82,1%	14:25	77,2%	79,3%
36	A	99	100	101			48,3%	19:00	58,7%	95,3%
37	A	102	103				93,3%	20:30	85,3%	64,6%
38	M	104	105				79,2%	14:50	69,7%	98,8%
39	M	106	107	108	109		70,8%	14:00	89,2%	24,5%
40	A	110	111				90,0%	20:10	84,7%	64,6%
41	M	112	113				74,4%	14:10	79,5%	83,5%
42	M	114	115	116			65,4%	13:55	70,3%	94,4%
43	M	117	118				82,1%	14:10	75,1%	96,1%
44	M	119	120	121	122		74,4%	14:10	79,7%	3,9%
45	M	123	124	125	126	127	64,1%	14:05	63,6%	91,9%
46	A	128	129				61,7%	19:10	66,0%	98,4%
47	A	130	131				76,7%	19:45	79,0%	79,3%
48	A	132	133				85,0%	19:55	98,3%	27,8%
49	M	134	135				80,8%	14:35	63,6%	99,9%
50	A	136	137				66,7%	19:20	80,3%	75,8%

Figure A.4: Manual Scheduling: blocks 1-50

Automatic Scheduling

Block	Type	Patients scheduled				Real Ocu.	End Time	Expected Ocu.	Conf. Level
1	M	1	2			83,3%	14:25	79,7%	84,0%
2	A	3	5			101,7%	21:05	80,3%	75,8%
3	M	4	8	12		76,9%	14:20	78,2%	80,3%
4	M	7	14			83,3%	14:25	84,6%	69,8%
5	A	6	22			75,7%	19:47	68,7%	97,1%
6	A	13	21			85,0%	20:15	79,3%	79,8%
7	M	9	10	24		88,5%	15:05	79,5%	77,6%
8	A	11	26			96,0%	20:48	80,0%	75,3%
9	M	15	16	25		73,1%	14:05	71,0%	94,5%
10	M	17	18	30		54,1%	12:51	74,4%	89,5%
11	M	23	27			82,1%	14:20	75,9%	88,9%
12	M	19	20	34		85,0%	14:56	80,3%	68,9%
13	M	28	29	37		75,6%	14:15	74,1%	90,3%
14	M	31	32	38		62,8%	13:25	72,8%	92,1%
15	M	35	36			64,1%	13:10	79,5%	83,5%
16	M	33	41	42		62,8%	13:25	74,4%	86,8%
17	A	39	40	48		61,7%	19:25	75,7%	73,8%
18	M	44	45	46		75,6%	14:15	80,8%	70,9%
19	M	47	50	51		76,9%	14:20	76,2%	85,4%
20	M	43	49	60		69,2%	13:50	75,9%	86,5%
21	M	52	53	54		88,5%	15:05	78,2%	80,4%
22	M	55	58	61		97,2%	15:40	80,0%	68,1%
23	M	59	65	71		71,8%	14:00	81,3%	69,0%
24	M	56	57	68	79	79,5%	14:50	74,1%	76,6%
25	A	62	63	64		66,7%	19:40	68,7%	88,7%
26	M	67	69	72	73	67,7%	14:04	76,2%	69,4%
27	A	66	70	81		63,3%	19:30	74,7%	72,9%
28	A	76	82			65,0%	19:15	80,3%	75,8%
29	M	74	75	77		85,9%	14:55	79,7%	73,5%
30	A	78	80	83		86,7%	20:40	74,7%	75,1%
31	M	84	85	87		78,2%	14:25	77,9%	79,6%
32	M	88	89	90	92	64,1%	13:50	72,8%	82,2%
33	M	86	94	95		71,8%	14:00	78,2%	78,1%
34	A	93	96	98		100,0%	21:20	71,3%	81,7%
35	M	91	97	99		52,6%	12:45	72,3%	88,7%
36	A	100	101	102		75,0%	20:05	75,7%	71,2%
37	A	103	106			78,3%	19:55	75,7%	88,7%
38	M	104	105			79,2%	14:15	69,7%	94,1%
39	M	107	108	110		62,5%	13:35	75,8%	80,0%
40	A	112	114			81,7%	20:05	76,7%	80,3%
41	M	109	111	116		74,4%	14:10	77,4%	80,9%
42	M	113	115	121		61,5%	13:20	74,1%	88,5%
43	M	117	118	122		85,9%	14:55	80,5%	68,4%
44	M	119	120	123		78,2%	14:25	80,0%	74,6%
45	M	128	129	130		76,9%	14:20	79,7%	71,8%
46	A	124	126	131		71,7%	19:55	77,3%	72,0%
47	A	127	133	137		61,7%	19:25	75,0%	73,0%
48	A	125	136	140		73,3%	20:00	77,3%	69,1%
49	M	132	143			76,9%	14:00	81,0%	79,3%
50	A	134	135			105,0%	21:15	82,7%	73,6%

Figure A.5: Automatic scheduling: blocks 1-50

Manual Scheduling

Block	Type	Patients scheduled					Real Ocu.	End Time	Expected Ocu.	Conf. Level
		138	139	140						
51	M	138	139	140			84,6%	14:45	79,0%	76,4%
52	A	141	142				70,0%	19:25	85,3%	64,6%
53	M	143	144	145	146		61,1%	13:50	63,1%	93,6%
54	M	147	148	149			82,1%	14:45	74,1%	88,5%
55	M	150	151	152	153		70,5%	13:45	64,1%	92,8%
56	A	154	155				76,7%	19:40	75,7%	88,7%
57	A	156	157				53,3%	19:15	58,3%	99,7%
58	M	158	159	160			71,8%	14:05	79,5%	74,5%
59	M	161	162	163	164		88,5%	15:05	78,7%	61,7%
60	A	165	166	167			73,3%	19:50	79,3%	60,8%
61	M	168	169	170			93,6%	12:30	86,7%	52,3%
62	A	171					58,3%	18:40	72,3%	98,8%
63	A	172	173				70,0%	19:15	66,0%	98,4%
64	M	174	175	176			69,4%	14:25	62,2%	98,8%
65	A	177	178				81,7%	20:00	75,7%	88,7%
66	A	179	180				61,7%	19:00	83,3%	66,9%
67	M	181	182				73,1%	13:55	57,9%	99,8%
68	M	183	184	185			78,2%	14:25	66,7%	95,4%
69	A	186	187				65,0%	19:05	85,3%	64,6%
70	A	188	189				76,7%	19:35	81,0%	74,0%
71	M	190	191	192			73,1%	14:20	74,9%	88,8%
72	A	193	194				75,0%	19:20	75,3%	83,5%
73	M	195	196	197			82,1%	14:40	81,8%	67,3%
74	M	198	199				75,6%	14:00	62,3%	99,7%
75	A	200	201				51,7%	18:20	70,7%	92,7%
76	M	202	203	204			55,1%	13:10	46,9%	100,0%
77	M	205	206	207			53,8%	12:35	71,8%	94,1%
78	M	208	209				74,4%	13:55	62,3%	99,7%
79	M	210	211	212			77,8%	14:10	80,6%	69,2%
80	M	213	214	215			73,1%	13:45	98,5%	16,7%
81	A	216	217				106,7%	20:55	85,3%	64,6%
82	M	218	219	220			79,5%	14:20	87,7%	48,6%
83	A	221	222				83,3%	20:10	75,3%	87,5%
84	M	223	224				79,2%	14:15	70,6%	96,3%
85	M	225	226	227	228		89,7%	15:15	82,3%	49,2%
86	M	229	230	231			91,0%	15:05	83,1%	63,3%
87	A	232	233				65,0%	19:05	70,7%	92,7%
88	A	234	235				66,7%	19:35	75,3%	83,5%
89	A	236	237				80,0%	20:05	85,3%	64,6%
90	M	238	239	240			76,9%	14:00	73,6%	91,5%
91	A	241	242				78,3%	19:35	84,7%	64,6%
92	M	243	244	245			86,1%	15:40	76,4%	78,8%
93	A	246	247				91,7%	20:15	79,7%	77,0%
94	M	248	249	250			76,9%	14:55	68,2%	96,6%
95	M	251	252				84,6%	14:20	65,6%	99,7%
96	A	253					80,0%	19:50	72,3%	98,8%
97	A	254	255				70,0%	19:05	75,7%	88,7%
98	A	256	257				78,3%	19:35	66,0%	98,4%
99	M	258	259	260			65,4%	13:35	71,8%	94,1%
100	M	261	262				84,6%	14:40	85,4%	74,5%

Figure A.6: Manual Scheduling: blocks 51-100

Automatic Scheduling

Block	Type	Patients scheduled				Real Ocu.	End Time	Expected Ocu.	Conf. Level
51	M	138	139	144		82,1%	14:40	73,8%	87,7%
52	A	141	145	146		60,0%	19:20	74,0%	75,5%
53	M	142	147	149		76,4%	14:25	80,3%	68,9%
54	M	148	151	154		91,0%	15:15	78,5%	78,7%
55	M	150	155	156		74,4%	14:10	77,4%	80,9%
56	A	153	157	162		66,7%	19:40	77,0%	68,4%
57	A	152	161	163		86,7%	20:40	76,3%	68,4%
58	M	158	159	160		71,8%	14:00	79,5%	74,5%
59	M	166	167	168		84,6%	14:50	79,2%	77,4%
60	A	165	169			76,7%	19:50	77,7%	85,3%
61	M	164	171			71,8%	13:40	81,0%	79,3%
62	A	170	172	173		78,3%	20:15	77,3%	68,9%
63	A	174	175	176		83,3%	20:30	74,7%	75,1%
64	M	177	180	185		76,4%	14:25	79,4%	73,2%
65	A	178	179			61,7%	19:05	75,0%	84,0%
66	A	181	186			75,0%	19:45	80,3%	75,8%
67	M	182	187	192		75,6%	14:15	78,5%	77,0%
68	M	188	190	197		70,5%	13:55	81,0%	70,9%
69	A	183	184			91,7%	20:35	75,3%	83,5%
70	A	189	195			98,3%	20:55	81,0%	74,0%
71	M	191	200	201		71,8%	14:00	80,8%	70,6%
72	A	194	198			96,7%	20:50	80,0%	75,3%
73	M	193	196	203	204	75,6%	14:35	75,6%	69,9%
74	M	199	205	206		66,7%	13:40	81,5%	69,4%
75	A	207	209	212		86,7%	20:40	76,0%	71,5%
76	M	202	218	225		87,2%	15:00	79,2%	73,0%
77	M	210	220	222		74,4%	14:10	78,2%	80,3%
78	M	208	219	228		60,3%	13:15	80,0%	69,6%
79	M	211	213			56,9%	12:55	71,1%	97,6%
80	M	214	227	231		65,4%	13:35	81,0%	68,3%
81	A	215	226			88,3%	20:25	75,7%	88,7%
82	M	216	217	240		98,7%	15:45	80,0%	74,6%
83	A	221	232			91,7%	20:35	80,0%	75,3%
84	M	223	224	243		84,7%	14:55	80,0%	68,8%
85	M	229	230	249		105,1%	16:10	81,0%	70,9%
86	M	236	237			61,5%	13:00	65,6%	99,7%
87	A	234	238			88,3%	20:25	80,3%	75,8%
88	A	233	242			63,3%	19:10	75,3%	87,5%
89	A	235	241			61,7%	19:05	80,0%	75,3%
90	M	244	246	271		87,2%	15:00	78,2%	78,1%
91	A	239	248			83,3%	20:10	77,0%	87,9%
92	M	245	247	272		90,3%	15:15	78,1%	72,9%
93	A	250	251			73,3%	19:40	68,7%	97,1%
94	M	253	255			84,6%	14:30	81,0%	79,3%
95	M	252	254	273		82,1%	14:40	74,4%	89,5%
96	A	256	258			81,7%	20:05	74,3%	91,5%
97	A	261				80,0%	19:40	72,3%	93,4%
98	A	262	263			61,7%	19:05	81,0%	74,0%
99	M	257	259	264		67,9%	13:45	77,9%	79,6%
100	M	265	267			74,4%	13:50	84,6%	69,8%

Figure A.7: Automatic scheduling: blocks 51-100

Manual Scheduling

Block	Type	Patients scheduled					Real Ocu.	End Time	Expected Ocu.	Conf. Level
101	A	263	264				63,3%	19:00	84,7%	64,6%
102	M	265	266				39,7%	11:45	54,4%	100,0%
103	M	267	268				73,1%	13:55	75,6%	95,7%
104	A	269					73,3%	19:15	72,3%	98,8%
105	M	270	271	272	273		66,7%	13:40	66,4%	89,9%
106	A	274	275				71,7%	19:25	84,7%	64,6%
107	M	276	277				92,3%	14:55	94,4%	42,1%
108	A	278	279				86,7%	20:30	84,0%	69,1%
109	M	280	281	282			84,6%	14:55	78,2%	80,3%
110	A	283	284				70,0%	19:10	85,0%	64,6%
111	A	285					73,3%	19:10	72,3%	98,8%
112	M	286	287				76,9%	14:05	65,6%	99,7%
113	A	288	289				48,3%	18:20	75,3%	83,5%
114	M	290	291	292			81,9%	14:30	80,0%	68,8%
115	A	293	294				70,0%	19:20	79,7%	77,0%
116	M	295	296	297	298		100,0%	15:30	91,3%	25,6%
117	M	299	300	301			89,7%	15:10	81,0%	69,1%
118	A	302	303	304			60,0%	19:20	74,3%	74,6%
119	M	305	306	307			78,2%	14:20	71,0%	89,5%
120	A	308					71,7%	18:55	72,3%	98,8%
121	M	309					65,3%	13:00	60,3%	100,0%
122	A	310	311				78,3%	19:30	79,0%	79,3%
123	M	312	313	314	315		62,8%	13:30	72,6%	83,2%
124	A	316	317				80,0%	20:00	75,0%	91,2%
125	A	318	319				61,7%	18:55	80,3%	75,8%
126	M	320	321				83,3%	14:25	65,6%	99,7%
127	M	322	323	324			79,0%	13:00	83,8%	61,8%
128	A	325	326	327			70,0%	19:30	92,3%	24,7%
129	A	328	329	330			95,0%	20:35	96,7%	15,1%
130	A	331					31,7%	17:00	72,3%	98,8%
131	M	332	333	334			88,9%	14:45	65,8%	97,0%
132	A	335	336				60,0%	18:45	72,0%	89,3%
133	M	337	338	339			79,5%	14:40	78,5%	78,7%
134	A	340	341	342			61,7%	19:05	79,7%	60,0%
135	M	343	344	345			70,5%	13:55	66,4%	98,7%
136	A	346	347				58,3%	18:25	70,0%	94,4%
137	M	348	349	350			78,2%	14:25	73,6%	87,9%
138	M	351	352				87,2%	14:55	61,0%	100,0%
139	M	353	354	355			69,2%	14:20	77,7%	80,5%
140	A	356					63,3%	18:40	72,3%	98,8%
141	A	357	358				76,7%	19:30	84,0%	69,1%
142	M	359	360	361			65,4%	14:25	69,2%	96,4%
143	M	362	363	364			66,7%	13:30	79,2%	77,0%
144	M	365	366				59,0%	13:00	59,5%	99,9%
145	M	367	368				64,1%	15:55	59,5%	99,9%
146	A	369	370				73,3%	19:10	74,7%	89,8%
147	M	371	372	373			75,6%	14:00	69,5%	94,7%
148	A	374	375	376			86,7%	20:10	74,0%	78,0%
149	M	377	378				77,8%	14:30	83,3%	77,0%
150	M	379	380	381			93,1%	15:30	89,2%	39,2%

Figure A.8: Manual Scheduling: blocks 101-150

Automatic Scheduling

Block	Type	Patients scheduled			Real Ocu.	End Time	Expected Ocu.	Conf. Level
101	A	269			73,3%	19:20	72,3%	93,4%
102	M	260	266	274	66,7%	13:40	77,9%	79,6%
103	M	268	270	275	74,4%	14:10	77,9%	79,6%
104	A	278	280		71,7%	19:35	68,7%	97,1%
105	M	276	282		95,8%	15:15	78,6%	89,9%
106	A	281	288		63,3%	19:10	80,3%	75,8%
107	M	277	279		69,2%	13:30	79,0%	91,7%
108	A	283	289		55,0%	18:45	80,0%	75,3%
109	M	285	293		79,5%	14:10	84,4%	70,9%
110	A	284	300		85,0%	20:15	80,3%	75,8%
111	A	286	302		83,3%	20:10	80,3%	75,8%
112	M	287	290	292	66,7%	13:40	74,1%	88,5%
113	A	291	305		106,7%	21:20	80,0%	75,3%
114	M	295	296	298	90,3%	15:15	78,3%	72,8%
115	A	297	301	303	61,7%	19:25	75,0%	71,5%
116	M	294	299	304	82,1%	14:40	74,1%	88,5%
117	M	307	308		70,5%	13:35	79,2%	80,9%
118	A	306	311	315	83,3%	20:30	76,7%	71,4%
119	M	309	310		87,2%	14:40	84,6%	69,8%
120	A	314	316		76,7%	19:50	75,0%	91,2%
121	M	312	313	318	65,3%	13:45	77,8%	77,6%
122	A	319	320		88,3%	20:25	80,3%	75,8%
123	M	317	323	329	83,3%	14:45	81,5%	72,1%
124	A	321	324	327	66,7%	19:40	74,7%	75,1%
125	A	322	340		67,7%	19:23	80,3%	75,8%
126	M	325	326		50,0%	12:15	65,6%	99,7%
127	M	336	339	341	65,4%	13:35	79,2%	72,9%
128	A	331			31,7%	17:15	72,3%	93,4%
129	A	328	344		76,7%	19:50	75,0%	91,2%
130	A	330	335	343	70,0%	19:50	75,3%	75,1%
131	M	334	337	338	86,1%	15:00	80,6%	69,2%
132	A	332	346		91,7%	20:35	80,3%	75,8%
133	M	333	348	349	96,2%	15:35	80,0%	72,4%
134	A	342	347	360	66,7%	19:40	75,0%	75,8%
135	M	345	350	351	98,7%	15:45	77,2%	76,3%
136	A	352	353	355	63,3%	19:30	65,3%	93,0%
137	M	354	358	361	69,2%	13:50	82,1%	68,3%
138	M	357	359	362	76,9%	14:20	78,5%	81,3%
139	M	356	370		74,4%	13:50	80,5%	81,0%
140	A	364	365		73,3%	19:40	81,3%	74,6%
141	A	363	371		83,3%	20:10	80,0%	75,3%
142	M	366	372	373	69,2%	13:50	70,3%	94,8%
143	M	367	368	376	73,1%	14:05	71,5%	90,5%
144	M	369	374	375	88,5%	15:05	77,4%	81,9%
145	M	379	380	381	85,9%	14:55	82,3%	68,3%
146	A	377	382		76,7%	19:50	80,0%	78,4%
147	M	383	386	388	84,6%	14:50	78,7%	78,0%
148	A	378	385		86,7%	20:20	82,3%	74,6%
149	M	384	387	397	84,7%	14:55	80,3%	68,9%
150	M	391	392		83,3%	14:30	82,5%	78,0%

Figure A.9: Automatic scheduling: blocks 101-150