2022 IEEE International Conference on
Systems, Man, and Cybernetics (SMC)

PM4SOS: low-effort resource allocation optimization in a dynamic environment

1st Jair José Ferronato
Informatics Graduate Program
PUCPR
Curitiba, Paraná, Brazil
jair.ferronato@gmail.com

2nd Edson Emílio Scalabrin
Informatics Graduate Program
PUCPR
Curitiba, Paraná, Brazil
edson.scalabrin@pucpr.br

3rd Deborah Ribeiro Carvalho
Health Technology Graduate Program
PUCPR
Curitiba, Paraná, Brazil
ribeiro.carvalho@pucpr.br

2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC) | 978-1-6654-5258-8/22/$31.00 ©2022 IEEE | DOI: 10.1109/SMC53654.2022.9945393

Abstract—**Surgical center scheduling is challenging to schedule physical spaces to reduce costs and increase productivity. This study presents a framework called PM4SOS (Process Mining for Simulation, Optimization, and Scheduling) that facilitates the generation of an operating room schedule in an automated way and integrates, reducing cognitive overload and waste of time. This framework allows the evaluation of restrictions to get the best performance of surgical schedules. PM4SOS combines process mining with data from event logs, generation of the automated simulation model, and case-based reasoning (CBR) to analyze management indicators, allowing significant gains in optimization of the scheduling of surgical centers. The results with PM4SOS help decision-making in hospital environments to better use physical resources and human resources, such as reducing waiting time, optimizing surgery execution time, and resource capacity management.**

Index Terms—**case-based reasoning, genetic algorithms, operation rooms, optimization, process mining, scheduling, simulation, simulated annealing.**

I. Introduction

Hospital environments, by their nature, are complex and operating rooms need planning. Surgical appointments affect other processes within a hospital, such as the availability of equipment, the scale of health professionals, and meeting current demands. In this scenario is challenging to discover and analyze the elements that influence surgical activity.

Surgical centers comprise a set of activities that use different human and physical resources that must be synchronized to obtain efficiency. Scaling surgeries in such environments must consider many conflicting situations that satisfy several constraints.

Task assignment and sequencing problems are considered NP-complete [1]. This type of drawback is difficult to unravel in typical ways; therefore, the quantity of computation required to seek out the best answer will increase exponentially with the scale of the matter [1]. For scheduling issues, the concept is to assign a collection of surgeries in operation rooms with schedules that satisfy a series of constraints, divided into two groups: hard and soft constraints. Rigid restrictions have higher priority than soft ones. The aim is to satisfy the rigid constraints and minimize the violation of soft constraints.

The main downside is apportioning the planned patient list in a schedule that meets and efficiently uses available resources. In this way, consider Patients' priorities and healthcare professionals' preferences. In addition, the maximization of the use of physical spaces and the number of surgeries must be taken into account to obtain profitability and reduce the waiting time in line for patients. Surgical scheduling task has always been solved manually in almost all healthcare institutions.

This study aims to integrate the elements of process mining, computer simulation, optimization, and CBR applied to the operating room scheduling problem. Our approach allows us to generate optimized solutions for the surgical center scheduling problem from readily available information in information systems.

Automating daily routines makes processes more efficient and employees more productive. The use of computer systems reduces the complexity of the activities involved.

Computer simulation is a technique that allows predicting, with some confidence, the behavior of a system based on data from specific inputs and respecting a set of premises [3]. Integrating simulation models with optimization modules is one of the current issues. One of the biggest impediments to the use of simulation and optimization is the execution time that this process demands [19].

Moreover, we need to feed the simulation model with accurate data to reduce cognitive effort and automate getting input parameters. Event logs store information such as identification of cases, activities, event timestamp, information about resources (people or devices) that perform the activities, and other data elements [2]. Process mining allows discovering the process model from an event log, identifying deviations, and optimizing processes in operation [2].

We want to optimize the processes to improve the hospital room programming system. Optimization is the art of finding the best alternative(s) among a given set of choices [4]. The method of finding the most or minimum potential cost that a given performance will attain in its domain of definition is thought of as improvement [4].

However, the surgical scheduling optimization process can be unfeasible due to its high complexity and an increasing number of variables and restrictions. For this environment, Case-Based Reasoning (CBR) uses the principle of experience to solve or provide suggestions for new problems. CBR engine contains a record of cases. The mechanism of retrieving get cases according to the similarity of the problem. Cases must be adjusted, evaluated, and re-established to suit the new problem.

We organized this paper as follows. We have detailed the scheduling problem in Section 2. In Section 3, we present the framework PM4SOS. We describe the optimization scheduling mechanism with details of experiments in integrated technologies in Section 4. Displays results and instantiates using a simple example in Section 5. Discusses the results and present the conclusions, limitations, learning, and opportunities in Section 6.

II. Surgical Scheduling Problem (SSP)

The scheduling operation room exists mainly in two versions: The advance scheduling establishes a date of the patient's (elective) surgery, and its objective is to minimize patient waiting time. In contrast, the allocation Scheduling involves setting up the operating room and the start time of the process on the specific day of the surgery, with a weekly schedule. [5].

Patient scheduling and operating room planning are complex tasks with several factors to consider, such as uncertainty in patient arrival, surgical procedure time, and medical prioritization and diagnosis [6].

A timetable is a schedule that has to suit many constraints. Individuals employ constraints almost universally by [8] to deal with timetabling problems. The schedule of surgeries involves the satisfaction of constraints categorized into hard and soft constraints. Hard constraints make scheduling unfeasible if not addressed. Soft constraints can reduce or increase the quality of the solution but not make scheduling unfeasible. We established the hard constraints: a) No operation room overlap; b) No surgeon overlap; c) Specific equipment and soft constraints: a) Surgeon's preferred day.

Above, in Table I, describe the description of objects in the SSP.

TABLE I
Object description to SSP

Class	***Description***
Surgeon	ID, Name of the surgeon and Preferred day
Procedure	ID, Name of procedure and Time of duration
Room	ID, Name of room and Room special equipment
Surgery	Surgeon, Procedure, Room, Day and Time

According to [7], the planning model will confirm the allocation of resources, jointly with operating rooms, surgeons, and assistants to surgeries, the sequence of surgeries among operating rooms, and their beginning times. The two steps determine surgical scheduling: a) a weekly arrangement to assign a selected date to every patient looking ahead to surgery, and b) sequencing and planning surgical cases on a given date [7].

To allocate time slots for scheduling surgeries can be used a two-dimensional matrix.

A set of six variables was constructed for the SSP problem. The set of surgeries C={c1,c2...}, the set of surgeons S={s1,s2...}, the set of Procedures P={p1,p2...}, the set of Operating Rooms R={r1,r2...}, the day set D={d1,d2...} and the time set T={t1,t2...}.

There are close relations between surgeries set, surgeons set, and surgical procedures set. Suppose c1,s1,p1 is one among the relations. That indicates surgeon code s1 performs that surgery code c1 and procedure code p1. In addition, s1,d1, and t1 are another relation, which means that one surgery is assigned in-room s1 on day d1 on time t1. If c1,s1,p1 link with r1,d1,t1, it means that surgeon code s1 in procedure code p1 will be scheduled at day d1 on time t1 at room r1 to perform surgery c1. These connections are tabulated in a matrix detailed in Table II.

TABLE II
Example of Scheduling matrix

	c1,s1,p1	c2,s2,p2	c3,s3,p3	...
r1,d1,t1	1	-1	0	...
r1,d1,t2	1	0	1	...
r1,d1,t3	-1	0	1	...
r1,d2,t1	-1	1	0	...
r2,d3,t1	0	0	1	...
...	...	...	...	...
#scheduled	2	1	3	...

In the table above, where each cell m_{lc} where l represents a row, and c represents a column, in the schedule matrix, $M = m_{lc}$ which can contain the following values:

- m_{lc} = 0 slot is available.
- m_{lc} = 1 slot occupied by a schedule.
- m_{lc} = -1 slot not available.

The number of slots needed is adequate to the set of surgeries multiplied by the set of days of the week with attainable planning and multiplied by the periods composed of one attainable hour of schedule.

III. Framework PM4SOS

This research investigates operating room scheduling. We developed a framework called PM4SOS. The latter integrates process mining, computer simulation, process optimization, and CBR to reuse past solutions, generating schedules more suited to the hospital's dynamics. Figure 1 presents the framework structure, and its elements.

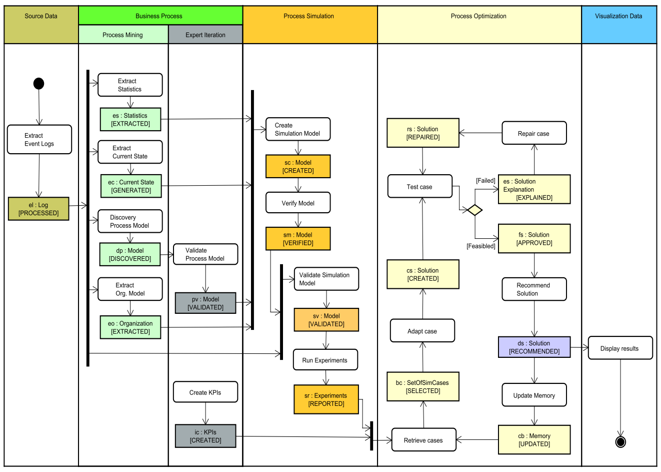


Fig. 1. PM4SOS Framework

Above, we describe the steps to extract parameters from method mining to construct the automated simulation model and optimization mechanism for planning.

1) **Source Data**: From the event log (L_0), we extracted an XES format file.
2) **Process Mining**: The event log provides information extracted as input: a) intervals between arrivals of each patient, the average time of execution of the activities, b) the organizational model that includes the performers of the activities and their respective links to the tasks, and c) the process model, which

comprises the application of discovery algorithms such as Inductive Minner. We used the framework PM4Py [9] to perform the main functions of process mining.

3) **Human Iteration**: The process requires human intervention in the following situations: a) when validating the process model and b) when creating minimum and leading indicators for each queue attribute. These established indicators allow us to compare the results obtained by running the simulation.

4) **Process Simulation**: To create the simulation model, we define some steps. Process mining allows generating a process tree with the sequence of activities known within the [10] event log. The Tree of Process shows an instance's path in the simulation model. Another subsidy for creating the simulation model was the Direct Flow Graph (DFG) to obtain the execution frequencies of the edges that connect each activity in the process tree and for decision objects. The event logs L_0 represent the actual system and provides the probability distributions to be input to the simulation model. We used the Poisson probability distribution for the interval between arrivals and activities' execution. We generate the Discrete Event Simulation (DES) model. We generate new instances with the inter-arrival metrics obtained from process mining, and we generated an event log at the end of the L_S simulation. Then, validate the simulation model automatically through a compliance analysis function that compares logs L_0 and L_S. An extra function estimates the cost of perfect alignment between the two logs. Thus, we observed that the average similarity between the simulated and original logs is more significant than 80%. We use the statistical results of running the simulation experiments during the preparation stage of the optimization.

5) **Process Optimization**: The process optimization is concentrated in CBR and consists of five phases, namely: a) **Retrieve**: Search based on cases of situations similar to the identified problem. From a query, identify a base of activity cases with high similarity [11]. b) **Adapt**: From a recovered case, its suggested solution is the subject of a reuse attempt to solve the target problem. It is considered an attempt since not always a solution retrieved and applied in the past fits the target problem perfectly. c) **Test**: A solution generated in the adaptation phase is not always correct and applicable. If the solution is correct, the cycle of case-based reasoning continues. On the other hand, if the adapted solution is not applicable, solution repair is performed [12]. d) **Repair**: Repairing failures, when detected, involves detecting which parts of the solution contain failures and the recovery or generation of explanations for these failures. We use fault explanations to modify the solution or the way the system the solution has arrived so that the present case can lead successfully and to prevent the failure occur again in the future [12]. e) **Store**: The new solution obtained — or the new adapted and revised case — is incorporated into the existing case database. It increases the knowledge available to the reasoner [12].

We add to the Process Simulation phases to recommend a schedule solution. The best solution that solves hard constraints and minimizes soft constraints are considered the recommendation.

6) **Display Results**: PM4SOS shows the results of the entire process performed. In this step, a dashboard presents a solution schedule.

IV. Optimize Scheduling

The simulation allows for analyzing future scenarios. Combining this "looking ahead" with process mining techniques, we have a powerful combination as an analysis tool based on event logs.

The optimization step takes as input the data obtained from the event logs in XES[1] format and with the application of process mining. Therefore, it is possible to obtain the process model, the average interval between the arrivals of process instances, and the organizational model that comprises the performers of each activity of the discovered process. Subsequently, the simulation model is created automatically, without human interaction. This mechanism consists of a combination of a process tree and a DFG [18].

Process mining feeds the simulation model, and we compare the simulated data to the indicators defined by the experts. From the results, we can show deviations compared with queue indicators, and in this way, we recommend using process optimization.

The main focus of CBR is to unravel new issues by memory a similar previous drawback and thus reusing the knowledge stored [13]. Figure 2 shows the CBR life-cycle with the main components and artifacts.

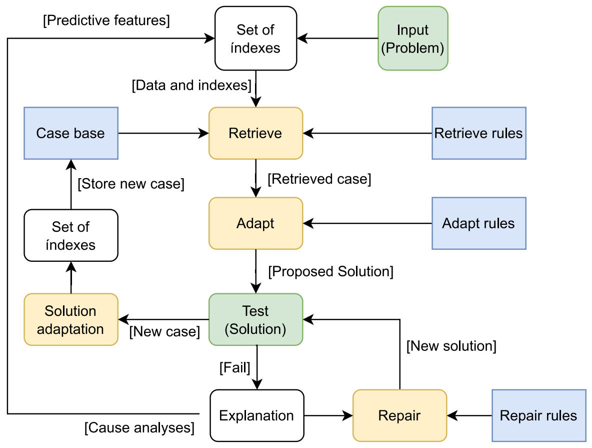


Fig. 2. The CBR life-cycle adapted from [13]

The formalism for representing the cases is of paramount importance to guarantee the success of the case-based reasoning system.

The queue indicators used are the number of patients going through the system (L), the average time of patients occupying the system (Lq), the number of patients on the waiting list (W), the average waiting time of patients in the waiting list (Wq) rate use of operating rooms (P). The queue discipline is (FIFO) First In, First Out.

Table III is our illustrative case base. For each case, we have: a) through the simulation model, we obtain the queue

[1]http://www.xes-standard.org

indicators; b) diagnosis identifies which indicator should be optimized; c) result of the optimization if it was successful or failed; d) description of the failure for the CBR to apply failover rules and e) solution that represents the final object optimized.

TABLE III
CASE REPRESENTATION

Case	L - Lq - W - Wq - P	Diag	Res	Fail	Solution
1	45 - 14 - 15 - 19 - 0.65	[1,4]	Fail	Desc	[ ]
2	12 - 10 - 20 - 21 - 0.91	[2,3]	Suc	[ ]	[Crom]
3	85 - 13 - 10 - 14 - 0.73	[3]	Suc	[ ]	[Crom]

Once the representation formalism is defined, it is necessary to transform the knowledge of a domain about the cases. Below, each step of the CBR mechanism will be detailed.

A. Retrieve cases

The first step is a case retrieve. It is a critical step in a CBR system. The search engine must be efficient since the search involves comparing how similar a target problem is to one or more cases. The retrieve function defines the set of similarity functions. Here we will use a linear similarity for each value. After retrieving the cases, we need to perform an adapt step.

Creating a surgery schedule is a problem that can be solved using a heuristic search. Algorithms in this category try to find the optimal solution but may be ineffective as complexity increases. Finding a significantly good solution could take the time or be not possible for additional complex requirements. Here is where genetic algorithms come into play. *Genetic Algorithms* is a particular evolutionary class that uses techniques inspired by evolutionary biologies such as heredity, transformation, natural selection, and recombination. [16].

The first factor we tend to contemplate once managing a genetic algorithm is a way to represent our solution so that it is viable for genetic operations like crossover and mutation [16]. Also, we tend to specify how good our resolution is, and in this manner, we should be ready to calculate the fitness value of our solution [16].

B. Adapt cases

The representation of a chromosome for the surgery scheduling problem needs to generate time slots to allocate the surgical procedures. We have segmented each slot into one-hour intervals for every day. We have established that the first surgery will start at seven in the morning, the last should begin at seven afternoons (twelve hours), and the working days are from Monday to Friday (a total of five days). We additionally established that the surgery would be of elective type only, not together with acute and emergency cases.

Now we want to assign a fitness cost to the chromosome. In this research, we used hard and soft requirements to calculate the appropriateness of a surgery schedule. We compare the result of the fitness function with a parameter assigned by an expert. We trigger an automatic repair mechanism if the result is lower than the established level.

The fitness function sets from 0 to 5 points. The score is an increment in specific situations: a) if a surgery uses a vacant operating room, b) if a surgery requires specific equipment and the scheduled room has the equipment, and c) if a surgeon does not overlap with other surgeries. We will not increment the score for that rule if a surgery schedule breaks a rule in any time slot it takes. The total score for a surgery schedule is the addition of the points from all surgeries. We represent fitness cost by single-precision floating-point numbers within the vary zero to one. Then calculate the fitness cost by the scheduled score divided by the most score. We tend to establish that the most score is the variety of surgeries multiplied by the times of the week.

The genetic algorithm randomly selects N pairs of parents from the present population and produces N new chromosomes by recreating a crossover operation on the pair of parents [16]. It then randomly selects N chromosomes from the present population and replaces them with new ones. The algorithm chooses chromosomes for replacement if it is among the most effective chromosomes within the population [16]. We repeat the two operations until the most effective chromosomes reach a fitness cost of one (meaning that each operation within the theme meets the requirements) [16]. If the answer is infeasible, executing the objective function is unnecessary. That way, we tend to evaluate constraints before objectives. If existence violates the constraints, we tend not to appraise the goals need not be the constraints violated.

The genetic algorithm performs two fundamental operations on chromosomes:

: This operation combines knowledge within the hash maps of two parents and creates a vector of slots in line with the content of the new hash map and splits hash maps from both parents into every which way sized chunks [16]. The quantity of crossover points (plus one) within the chromosome parameters defines the number of components of operations. After that, it generates a new chromosome with the half inherited from the parents and forms a brand new slot list [16].

: This operation permits to pick a surgery at random and moves it to a different randomly chosen slot. The mutation scale within the chromosome parameters defines the number of surgeries that move during a single operation.

Our research uses the Archive-based Micro Genetic Algorithm (AMGA2). This algorithm works with minimal population size and maintains a sizable external archive of affordable solutions obtained [14]. Using an external archive that stores an outsized range of solutions provides valuable info regarding the search area still as it tends to come up with an outsized range of Pareto points at the end of the simulation [14].

The Algorithm 1 [14] presents and includes the operations performed in the AMGA2 algorithm. Moreover, after this execution, a test is performed to verify the chromosome's fitness of the solution found.

C. Test solution

Only if we meet all hard constraints perform the test step. Then, the percentage of soft constraints met is analyzed. Suppose the percentage is less than a specific value, for example. In that case, 50%, a recovery mechanism comes into play and provides a description to identify the cause so that the CBR engine can assess future similar issues.

Algorithm 1 AMGA2 pseudo-code [14]

```
Generate initial population
Evaluate initial population
Update the archive (using the initial population)
repeat
    Create parent population from the archive
    Create mating pool from the parent population and the archive
    Create offspring population from the mating pool by crossover followed by mutation
    Evaluate the offspring population
    Update the archive (using the offspring population)
until (termination)
Report desired number of solutions from the archive
```

D. Repair solution

In addition, an automatic mechanism is triggered and associated with a Simulated Annealing (SA) algorithm. This algorithm is essential because it allows adding quality to the solution found by the AMGA2 algorithm. Local search algorithms use the neighborhood and fitness functions to find a location optimum. SA uses a temperature control parameter and a cooling schedule to escape the local optimum and find a solution closer to a global optimum [15].

We adopted a SA algorithm, creating a cost function that analyzes the effort to resolve soft constraints. We calculate the total cost by combining the sum of the soft constraints to be resolved. The following parameters set a cooling schedule: a) Start temperature, b) Temperature drop function, and c) End temperature or stop condition.

The probability of accepting solutions worse than the current one in SA is higher initially. As the temperature drops, it finds a global optimum with a probability approaching 1. The SA adapted from [17] is bellow in Algorithm 2.

Algorithm 2 Simulated annealing algorithm

```
function SA(problem, schedule)
current ← problem.INITIAL; {Solution from AMGA2}
for $t = 1$ to $\infty$ do
    $T \leftarrow schedule(t)$
    if $T = 0$ then
        return current
    end if
    next ← a randomly selected successor of current
    $\Delta E \leftarrow VALUE(current) - VALUE(next)$
    if $\Delta E > 0$ then
        current ← next
    else
        current ← next only with probability $e^{-\Delta E/T}$
    end if
end for
end SA
```

Next, the solution's repair mechanism returns to the test step and runs the solution test with the minimum required parameters.

E. Store solution

After Repair and Test again, if the solution passes the Test with the necessary indicators, it can be considered a viable solution and stored in the case base to resolve future similar issues more quickly.

V. Experiment Results

We conducted experiments to develop an application in a Python programming language. We obtain the data for the experiments in an emergency hospital. We filtered the data over two weeks, and appointments in six operating rooms were analyzed. Subsequently, we use the data as input to the simulation model. Instances were created and analyzed with this framework's simulation execution and optimization.

We established some parameters in this experiment: a) The sum of hours on schedule must not exceed the capacity of the available slots in the week; b) For the execution of the AMGA2, a stopping criterion was established: the maximum fitness obtained with the resolution of the rigid constraints or the stagnation of obtaining improvements; c) The repair mechanism use SA and have as a metric: the Repair only is triggered if AMGA2 was not able to obtain at least 50% of the soft constraints solution.

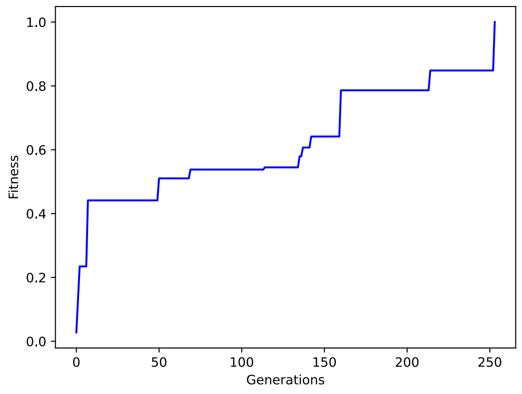


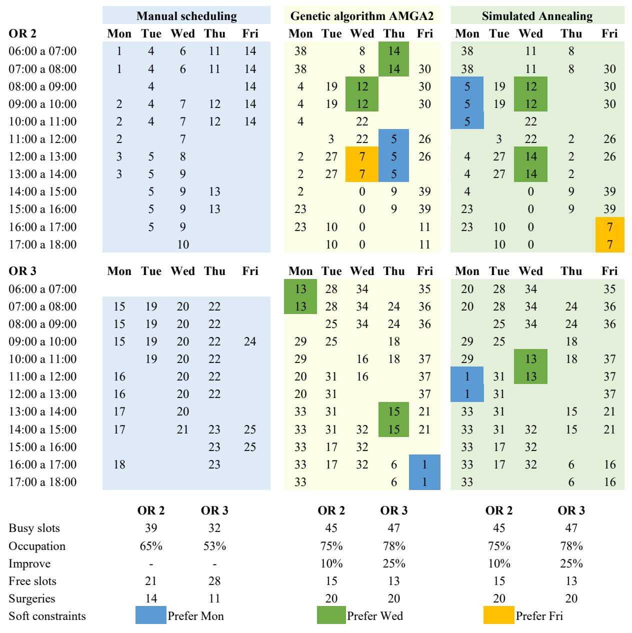
Fig. 3. Fitness by AMGA2 generations

We describe the results obtained with the CBR adaptation phase with the application of AMGA2 and the CBR repair phase with the application of the SA in Figure 4. The gains obtained with the use of multi-objective optimization techniques are significant. Manual scheduling had an occupancy rate of 65% in OR 2 and 53% in OR 3. The adaptation stage with AMGA2 increased 10% in occupancy for OR 2 and 25% for OR3. The number of surgeries was increased from 12 to 16 in OR2 and from 11 to 18 in OR3 with an optimization framework.

The experiment used a sample of 30 surgeries distributed in two rooms scheduled manually and automatically. We apply a paired t-test of related samples with a significance level of 5%. Manual scheduling t(30) = -10.08, P= 4.93182E-11 and automated scheduling t(30) = -18.73, P = 1.55409E-18. We inferred a significant difference between the scheduling performed manually and the scheduling performed by the PM4SOS.

We do not increase the use of room spaces when applying the SA. However, it was possible to make the allocation to prioritize the surgeons' preferred days. The experiment started with evaluating seven soft restrictions and solved six with an 85% resolution rate. This way, quality was added to the initial solution obtained with AMGA2.

Figure 3 shows that entering an additional substantial variety of surgeries is attainable, the number of times which will increase. This approach uses a two-tier fitness



	Manual scheduling					Genetic algorithm AMGA2					Simulated Annealing				
OR 2	**Mon**	**Tue**	**Wed**	**Thu**	**Fri**	**Mon**	**Tue**	**Wed**	**Thu**	**Fri**	**Mon**	**Tue**	**Wed**	**Thu**	**Fri**
06:00 a 07:00	1	4	6	11	14	38		8	🟩 14		38		11	8	
07:00 a 08:00	1	4	6	11	14	38		8	🟩 14	30	38		11	8	30
08:00 a 09:00		4			14	4	19	🟩 12		30	🟦 5	19	🟩 12		30
09:00 a 10:00	2	4	7	12	14	4	19	🟩 12		30	🟦 5	19	🟩 12		30
10:00 a 11:00	2	4	7	12	14	4		22			🟦 5		22		
11:00 a 12:00	2		7				3	22	🟦 5	26		3	22	2	26
12:00 a 13:00	3	5	8			2	27	🟨 7	🟦 5	26	4	27	🟩 14	2	26
13:00 a 14:00	3	5	9			2	27	🟨 7	🟦 5		4	27	🟩 14	2	
14:00 a 15:00		5	9	13		2		0	9	39	4		0	9	39
15:00 a 16:00		5	9	13		23		0	9	39	23		0	9	39
16:00 a 17:00		5	9			23	10	0		11	23	10	0		🟨 7
17:00 a 18:00			10				10	0		11		10	0		🟨 7
OR 3	**Mon**	**Tue**	**Wed**	**Thu**	**Fri**	**Mon**	**Tue**	**Wed**	**Thu**	**Fri**	**Mon**	**Tue**	**Wed**	**Thu**	**Fri**
06:00 a 07:00						🟩 13	28	34		35	20	28	34		35
07:00 a 08:00	15	19	20	22		🟩 13	28	34	24	36	20	28	34	24	36
08:00 a 09:00	15	19	20	22			25	34	24	36		25	34	24	36
09:00 a 10:00	15	19	20	22	24	29	25		18		29	25		18	
10:00 a 11:00		19	20	22		29		16	18	37	29		🟩 13	18	37
11:00 a 12:00	16		20	22		20	31	16		37	🟦 1	31	🟩 13		37
12:00 a 13:00	16		20	22		20	31			37	🟦 1	31			37
13:00 a 14:00	17		20			33	31		🟩 15	21	33	31		15	21
14:00 a 15:00	17		21	23	25	33	31	32	🟩 15	21	33	31	32	15	21
15:00 a 16:00				23	25	33	17	32			33	17	32		
16:00 a 17:00	18			23		33	17	32	6	🟦 1	33	17	32	6	16
17:00 a 18:00						33			6	🟦 1	33			6	16

	OR 2	**OR 3**	**OR 2**	**OR 3**	**OR 2**	**OR 3**
Busy slots	39	32	45	47	45	47
Occupation	65%	53%	75%	78%	75%	78%
Improve	-	-	10%	25%	10%	25%
Free slots	21	28	15	13	15	13
Surgeries	14	11	20	20	20	20
Soft constraints	🟦 Prefer Mon		🟩 Prefer Wed		🟨 Prefer Fri	

Fig. 4. Results from experiments

assignment mechanism, ranking. The results infer that the PM4SOS allows adding value to the scheduling system and reduces operating costs. Using available data such as event logs to support creating and using a simulation model to optimize surgical schedules makes for a powerful and dynamic solution. We overcome restrictions by bringing agility to operational routines, reducing patient waiting times, efficiently using available resources, and increasing employee satisfaction. Moreover, the framework allows adapting the personal preferences of professionals to the demands of the hospital environment.

VI. Conclusions

This research integrates information available in information systems and aggregates simulation and computational optimization with the support of the case-based learning mechanism. We presented PM4SOS, an integrated framework to optimize a scheduling operations room from process mining and simulation. The PM4SOS allows the extraction of attributes from event logs to automate the creation of a simulation model and analyze queue indicators.

We used the CBR as a learning mechanism to retrieve from case base similar cases imputed as a problem. The CBR adapts multi-objective optimization that maintains an external archive of best and diverse solutions and a tiny working population. The test procedure evaluates the solution found by AMGA2. After recovery, the initial solution is improved with a local-based search to seek the satisfaction of soft constraints. CBR helped reduce scheduling planning time and stores the new solution in its case base for future reference to similar problems. The user views the operating room scheduling recommendation.

We highlighted the following findings with the creation of PM4SOS: a) From an event log, we extracted the process model and the necessary parameters to create a simulation model in an automated way and without human interaction; b) Use of XES standardized data source. This standardization allows the simulation to be performed with low effort and provides scheduling optimization; c) uses a learning mechanism to reuse the knowledge base for similar problems, and d) significant gains in the automation of scheduling and computational techniques concerning manual scheduling.

One limitation of this study is the quality of the discovered model; we offer a filter log. We observed the system's behavior and identified that the greater the number of restrictions, the longer the processing time to get the schedule.

Future research should expand PM4SOS to incorporate all operating rooms into a hospital setting and expand the number of hard and soft restrictions. We tend to explore the combination of other heuristics to amplify the satisfaction of surgical scheduling constraints.

Acknowledgment

The authors would like to thank the support and funding of CAPES (Coordination for the Improvement of Higher Education Personnel) - Brasil - Finance Code 001 under Grant Nos.: 88887.341518/2019-00

References

[1] Garey, M. R. Johnson, D. S, "Computers and intractability: a guide to the theory of NP-completeness", Freeman, 1979.
[2] W. M. P. van der Aalst, "Process Mining: Data Science in Action", 2nd ed. Heidelberg: Springer, 2016.
[3] L. Chwif and A. C. Medina, "Modelagem e simulação de eventos discretos. Teoria e aplicações", 4th ed., A. C. M. Leonardo Chwif, Ed. Rio de Janeiro: Elsevier, 2015.
[4] A. K. Bhunia, L. Sahoo, and A. A. Shaikh, "Advanced Optimization and Operations Research", Springer, 2019.
[5] Abdalkareem, Z.A., Amir, A., Al-Betar, M.A. et al. "Healthcare scheduling in optimization context: a review". Health Technol. 11, 445–469, 2021.
[6] Persson, Marie. "Modelling and Analysing Hospital Surgery Operations Management," Licentiate dissertation, Blekinge Institute of Technology, Karlskrona, 2007.
[7] Somayeh, Ghazalbash; Sepehri, Mohammad Mehdi; Shadpour, Pejman and Atighehchian, Arezoo, "Operating Room Scheduling in Teaching Hospitals". Advances in Operations Research. 2012.
[8] E. K. Burke, D. G. Elliman, R. Weare. A University Timetabling System Based on Graph Colouring and Constraint Manipulation, Journal of Research on Computing in Education, 27:1, 1-18, 1994 DOI: 10.1080/08886504.1994.10782112
[9] A. Berti, S. J. van Zelst, and W. van der Aalst, "Process Mining for Python (PM4Py): Bridging the Gap Between Process and Data Science", CEUR Workshop Proceedings, vol. 2374, pp. 13–16, may 2019. [Online]. Available: http://arxiv.org/abs/1905.06169.
[10] A. Vahedian K. and S. Alizadeh, "A new model for discovering process trees from event logs", Applied Intelligence, vol. 41, no. 3, 2014.
[11] Bergmann, Ralph. et. al. "Case-Based Reasoning - Introduction and Recent Developments". Künstliche Intelligenz, 2009.
[12] Kolodner, J. L. "Case-Based Reasoning", Morgan Kaufmann, San Mateo, CA, 1993.
[13] Riesbeck, C. K, Schank, R. C. "Inside Case-Based Reasoning". 1989. DOI: doi.org/10.4324/9780203781821
[14] Tiwari, Santosh, Fadel, Georges, Deb, Kalyan. "AMGA2: Improving the performance of the archive-based micro-genetic algorithm for multi-objective optimization". Engineering Optimization, 43. 2011.
[15] Aarts, E., Kort, J.,Michiels, W. Simulated Annealing. In Burke, E., Kendall, G.(eds.) Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques. Springer, 2005.
[16] Goldberg, David E.. Genetic Algorithms in Search, Optimization, and Machine Learning. EUA: Addison-Wesley, 2009.
[17] Russell, Stuart and Norvig, Peter. "Artificial Intelligence: A Modern Approach", 4th ed. 2020.
[18] Ferronato. J. J. and Scalabrin, E. E. "PM2Sim: The Automated Creation of a Simulation Model from Process Mining," 2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC), 2021, pp. 2232-2237, doi: 10.1109/SMC52423.2021.9659211.
[19] Banks, J. "Panel session: the future of simulation," in Proceedings of the 2001 Winter Simulation Conference, Arlington, VA, USA, 2001 pp. 1453-1460.