An ASP-based solution to the Operating Room Scheduling with care units

Giuseppe Galatà[1], Marco Maratea[2], Marco Mochi[1,2], Victoria Morozan[2] and Ivan Porro[1]

[1] *SurgiQ srl, Italy*

[2] *DIBRIS, University of Genova, Genova, Italy*

Abstract

The optimization of daily operating room surgery schedule can be problematic because of many constraints, like to determine the start time of different surgeries and allocating the required resources, including the availability of beds in different units. Recently, Answer Set Programming (ASP) has been successfully employed for addressing and solving real-life scheduling and planning problems in the health-care domain. In this paper we present an enhanced solution using ASP for scheduling and rescheduling operating rooms taking explicitly into considerations the availability of beds for intensive and post-anesthesia care units. We tested our solution on different benchmarks with realistic parameters. The results of our experiments show that ASP is a suitable methodology for solving also such enhanced problem.

Keywords

Healthcare, Operating Room Scheduling, Answer Set Programming

1. Introduction

Hospitals have in long waiting times, surgeries cancellation and even worst resource overload, problems that negatively impact the level of patients satisfaction and the quality of care provided. Within every hospital, Operating Rooms (ORs) are an important unit. As indicated by [1], the ORs account for approximately 33% of the total hospital budget because it includes high staff costs (e..g, surgeons, anaesthetists, nurses) and material cost. Nowadays, in most modern hospitals, long surgical waiting lists are present because of inefficient planning. Therefore, it is extremely important to improve the efficiency of ORs management to enhance the survival rate and satisfaction of patients, thereby improving the overall quality of healthcare system.
To manage the ORs, a solution has to provide the date and the starting time of the surgeries required, considering the availability of ORs and beds, and the other resources requested. In particular, after the surgeries, some patients could require to remain in the hospital, either in the post-anesthesia care unit for some hours or in the intensive care unit, for some days. The Operating Room Scheduling (ORS) [2, 3, 4, 1] problem is the task of assigning patients to ORs by considering specialties, surgery durations, shift durations, and beds availability, among

RCRA 2021: 28th RCRA International Workshop on Experimental Evaluation of Algorithms for Solving Problems with Combinatorial Explosion

giuseppe.galatà@surgiq.com (G. Galatà); marco.maratea@unige.it (M. Maratea); marco.mochi@unige.it (M. Mochi); victoria.morozan@edu.unige.it (V. Morozan); ivan.porro@surgiq.com (I. Porro)

others. Further, the solution must prioritise patients based on health urgency. In recent years a solution based on Answer Set Programming (ASP) [5, 6, 7] was proposed and is used for solving such problem [8, 9], that followed other similar scheduling problems in this context (e.g., Nurse Scheduling [10, 11]): this is because ASP combines an intuitive semantics [12] with the availability of efficient solvers [13, 14] put forward by the ASP Competition series (see, e.g., [15, 16, 17]). We have recently enhanced the previous solutions by incorporating bed management [18]. In this paper we improve the solution and present an enhanced encoding that takes into explicit account the availability of beds for post-anesthesia care unit (PACU), other than for the ward and the intensive care unit (ICU). The problem is expressed in ASP as modular additions to previous, more limited encoding, of ASP rules implementing PACU, and then efficient solvers like CLINGO [19] are used to solve the resulting ASP encoding. Results for planning horizons of 5 days, obtained on different scenarios with realistic parameters for a small-medium sized Hospital are positive, and inline with Hospital needs, and further confirm that ASP is a suitable methodology for solving scheduling problems in the healthcare domain.

The paper is structured as follows. Section 2 describes the target problem in an informal way, whose ASP encoding is presented in Section 3. Section 4 shows the results of our experiments. The paper ends in Section 5 by showing conclusions and possible topics for further research.

2. Problem Description

In this paper, the elements of the waiting list are called *registrations*. Each registration links a particular surgical procedure, with a duration, to a patient.

The overall goal of the ORS problem is to assign the maximum number of registrations to the operating rooms (ORs). As first requirement, the assignments must guarantee that the duration of surgeries assigned to a particular OR does not exceed the closing time of the OR itself: patients must be assigned to a starting time taking into account the duration of their surgeries, to be sure that the surgery is completed before the closing time of the OR. Moreover, registrations are not all equal: they can be related to different medical conditions and can be in the waiting list for different periods of time. These two factors can be unified in a unique concept: *priority*. Registrations are classified according to three different priority categories, namely P_1, P_2 and P_3. The first one gathers either very urgent registrations or the ones that have been in the waiting list for a long period of time; it is required that these registrations are all assigned to an OR. Then, the registrations of the other two categories are assigned to the top of the ORs capacity, prioritizing P_2 over P_3 (*minimization*). Moreover, in addition to the solution proposed in previous works, in this paper we handle the management of the post-anesthesia care unit (PACU) unit, other than the ward and the intensive care unit (ICU). The stay in PACU or ICU can be needed for patients involved in complex surgical operations; in particular, patients requiring ICU remain in this unit for 1 or more days, while patients requiring PACU can remain in the unit for few hours (usually 60 or 180 minutes) after the surgery. Thus, the solution have to assign a bed for the needed amount of time to patients in the PACU or in the ICU.

However, in hospital units it is frequent that one planned assignment of ORs cannot be fulfilled due to complications or conflicts that may occur either during the surgery or before. In particular, surgeries may last longer than expected or some patients may delete the registration.

```
0 {x(RID,PR,ORID,S,DAY,HOUR): HOUR + SURGDUR < M} 1:- reg(RID,PR,SURGDUR,_,SPECID,_,_,PACUDUR),
     mss(ORID,S,SPECID,DAY), blockDuration(M,ORID,S),HOUR=0..M.
regSurgery(RID,DAY, HOUR .. HOUR+SURGDUR-1,ORID,S) :- x(RID,_,ORID,S,DAY,HOUR),
     reg(RID,_,SURGDUR,_,_,_,_,_).
:- regSurgery(RID1,DAY,HOUR,ORID,S), regSurgery(RID2,DAY,HOUR,ORID,S), RID1 != RID2.
admissionDatesIC(RID,DAY..DAY + ADMDURIC - 1) :- reg(RID,_,_,_,_,ADMDURIC,_,_),
     x(RID,_,_,_,DAY,_), ADMDURIC > 0.
admissionDatesPACU(RID,HOUR+SURGDUR..HOUR+SURGDUR+PACUDUR-1,S,OPDAY):-
     reg(RID,_,SURGDUR,_,_,_,_,PACUDUR),x(RID,_,_,S,OPDAY,HOUR),PACUDUR>0,HOUR+SURGDUR>HOUR.
:- #count {RID: admissionDatesIC(RID,DAY)} > M, bedsAvailable(0,M,DAY).
:- #count {RID: admissionDatesPACU(RID,HOUR,S,DAY)} > M, bedsAvailable(6,M,HOUR,DAY).
```

Figure 1: ASP encoding of the scheduling problem

Therefore, in such cases it is required to compute a new schedule which reallocates the ORs and, at the same time, minimizes the differences with a previously computed schedule. This problem is usually referred to as *rescheduling*. In our solution, we propose a rescheduler dealing with changes in surgeries of patients requiring to stay in the PACU after the surgery. The main objective of the scheduling was to assign the largest possible number of registrations to the OR sessions, while in the rescheduling problem the objective is to reassign all the previously allocated registrations and the reallocated ones with the least possible disruption to the old schedule.

3. ASP Encoding

In this section we present the ASP encoding for the new rules and constraints required for the managments of the PACU and ICU. These rules can be used in addition to an encoder for the ORS problem.

We assume the reader is familiar with syntax and semantics of ASP. Starting from the specifications in the previous section, here we present the ASP encoding, based on the input language of CLINGO [19]. For details about syntax and semantics of ASP programs we refer the reader to [12].

Data Model. The input data is specified by means of the following atoms:

- Instances of `reg(RID,PR,SURGDUR,ADMUR,SPECID,ADMURIC,ADM,PACUDUR)` represent the registrations, characterized by an id (RID), the priority level (PR), the duration of the surgery (SURGDUR), the number of days the patient will need to stay in the hospital (ADMUR), an id of the specialty of the surgical operation (SPECID), the numbers of days the patient will need to stay in the ICU (ADMURIC), the number of days before the surgical operation the patient will need a ward in the hospital, and the time slots the patients will need to stay in the PACU after the surgery (PACUDUR).
- Instances of `mss(ORID,S,SPECID,DAY)` represent the available surgery room, characterized by an id (ORID), which is available during the shift (S) for the specialty (SPECID) in the day (DAY).

```
0 {y(RID,PR,ORID,S,DAY,HOUR): HOUR + SURGDUR < M} 1 :-
    reg(RID,PR,SURGDUR,_,SPECID,_,_,PACUDUR), mss(ORID,S,SPECID,DAY),
    blockDuration(M,ORID,S),HOUR=0..M.
:- not y(RID,PR,_,_,_,_), x(RID,PR,_,_,_,_).
:~ y(RID,_,_,_,DAY,_),x(RID,_,_,_,OldD,_), DF = |DAY - OldD|. [DF@2, RID]
:~ y(RID,_,_,TURN,DAY,_),x(RID,_,_,OLDTURN,DAY,_),DF = |TURN - OLDTURN |. [DF@1, RID]
```

Figure 2: ASP encoding of the rescheduling problem

- Instances of `blockDuration(M,ORID,S)` represent the last time slot (`M`) in which the surgery room, characterized by an id (`ORID`), is open during the shift (`S`).
- Instances of `bedsAvailable(SPECID,M,DAY)` represent the number (`M`) of available beds in the specialty (`SPECID`) during the day (`DAY`).
- Instances of `bedsAvailable(SPECID,M,HOUR,DAY)` represent the number (`M`) of available beds in the specialty (`SPECID`) during the day (`DAY`) in the hour (`HOUR`).

The output is an assignment represented by atom of the form:

`x(RID,PR,ORID,S,DAY,HOUR)`

where the intuitive meaning is that the surgery of registration with id `RID` and priority level `PR` in the room `ORID` is assigned to the shift `S` in the day `DAY` and to the hour `HOUR`.

Encoding. The additional part of the encoding is shown in Figure 1, and is described in the following: the full encoding includes the one in [18] plus the current rules. To simplify the description, we denote as r_i the rule appearing at line i of Figure 1.

Rule r_1 assigns registrations to a day, an hour and a shift. The assignment is made assigning an hour that summed to the duration of the surgery is before the closing time of the room. Rule r_2 and $_3$ are used to check that every registration is assigned to a day, an hour, a shift, and an operatory room in which there are no patients already assigned. Then, rule r_4 and r_5 assigns the number of days and hours after the surgery to patients requiring to stay in ICU and PACU respectively. Rule r_5 and r_6 are used to be sure to have less patients than available beds in the ICU and PACU, characterized by a specialty ID equal to 0 and 6, respectively.

3.1. Rescheduling

We now formulate the rescheduling in ASP. As for the scheduling, in Figure 2 are presented just the new rules needed by the rescheduler, while the rules that checks the availability of beds and space in the ORs are equal to the rules presented in Figure 1, the only difference is that the assignments are represented by atoms y instead of x.

3.1.1. Data Model

The input data is specified by means of the following atoms:

- The old planning is encoded through facts represented by instances of the predicate `x(RID,PR,ORID,S,DAY,HOUR)`.

- `mss`, `reg` and `blockDuration` are described by the same predicates as in the previous section.

The output is a new assignment, represented by atoms of the form

$$\texttt{y(RID,PR,ORID,S,DAY,HOUR).} \tag{1}$$

3.1.2. Encoding

The new encoding is reported in Figure 2. It basically includes rules from r_2 to r_7 from the previous encoding, where atoms over the predicate x are replaced with y, respectively. Rule r_1 works as the first rule in the previous encoding. Constraint r_2 must be added to ensure that for every single registration in the old schedule (x predicate) there is an assignment in the new one (y predicate). Rule r_3 minimizes the difference in days between the new and old assignments, while rule r_4 minimizes the difference in shift between the new and old assignments that have been reassigned the same day as in the old schedule.

4. Experimental Results

4.1. Benchmarks

The test cases we have assembled for the initial planning are based on the requirements of a typical middle sized hospital, with five surgical specialties to be managed. To test scalability, other than the 5-days planning period, which is the one that is widely used in Italian hospital units, two benchmarks of different beds availability were created. Each benchmark was tested 10 times with different randomly generated inputs. The characteristics of the tests are the following:

- 2 different benchmarks, comprising a planning period of 5 work days;
- 10 ORs (that can represent a hospital of small-medium size), unevenly distributed among the specialties;
- 5 hours long morning and afternoon sessions for each operating room, summing up to a total of respectively 3000 hours of ORs available time for the two benchmarks;
- for each benchmark are generated 350 patients from which the scheduler will draw the assignments.
- 90 percent of patients require to stay in the PACU, while 10 percent require to stay in the ICU. Registrations are characterized by a surgery duration, a specialty and a priority. In this way, we simulate the common situation where a hospital manager takes an ordered, w.r.t. priorities, waiting list and tries to assign as many elements as possible to each OR.

The surgery durations have been generated assuming a normal distribution, while the priorities have been generated from a quasi-uniform distribution of three possible values (with weights respectively of 0.20, 0.4 and 0.4 for registrations having priority 1, 2 and 3, respectively).

The two scenario generated differs for the beds availability, in particular, the first scenario (Scenario A) simulates high availability of beds for all the specialties, even for the PACU and

Table 1
Beds Available in Scenario A and B in all the specialties

Specialty	#beds Scenario A	#beds Scenario B
ICU	40	4 - 6
1	80	20 - 50
2	58	10 - 35
3	65	10 - 35
4	57	8 - 18
5	40	10 - 25
PACU	15	5

ICU, while the second scenario (Scenario B) is characterized by low beds availability for all the units. In scenario A, the number of beds are the same for each specialty in all the day, while in scenario B each day the number of beds increase for each specialty. The number of available beds for the different specialties in the two scenarios are summed up in Table 1, where for scenario B the range is reported.

4.2. Results

The experiments were run on a AMD Ryzen 5 2600 CPU @ 3.40GHz with 16 GB of physical RAM. Results of the experiments are reported for both scenario A and B in Table 2. Each benchmark was tested 10 times with different randomly generated inputs. A time limit of 60 seconds was set for each experiment. In each table averages for 10 instances for each benchmark are reported. The first three columns show the number of assigned registrations out of the generated ones for each priority P1, P2 and P3, while the last three columns show a measure of the total time occupied by the assigned registrations as a percentage of the total OR time available (indicated as OR time Eff in the tables) and the total percentage of ICU and PACU beds usage (indicated respectively as ICU Eff and PACU Eff in the tables). As we can see, in scenario A (Table 2) the solution was almost able to assign a day and hour of surgery to all the patients with priority 2. In this scenario the scheduler is more limited by the OR time available than by the availability of beds in ICU and PACU. To validate the goodness of our results we used as baseline a model that does not take into account PACU beds. Thus, we deleted rules r_5 and r_7 from the encoder presented in Figure 1. The results obtained by the model in the scenario A lead to an increase of the Total OR time Efficency of just 2%, since this slight increase can be explained by the fact that the model can fill some gap between patients since it doesn't take into account the stay in PACU, we can say that adding the rules for the management of the PACU beds does not deteriorate the performance of the model. In scenario B (Table 2) having decreased the number of beds available in all the units, the schedule is not able to assign as many patients as before, and in particular, the scheduler is limited by the available beds in ICU. Indeed, the solution reachs 100% of usage of ICU beds with all the instances, while the percentage usage of PACU beds, even if it is bigger than the usage in scenario A, is lower than 80%. Thus, from the results, an hospital manager could decide to increase the number of beds for the ICU or implementing shared beds between the two units.

Table 2
Averages of the results for Scenario A and B.

P1	P2	P3	Total OR time Eff	ICU Eff	PACU Eff	Scenario
100%	94%	15%	78%	15%	58%	A
100%	31%	3%	39%	100%	79%	B

4.3. Rescheduling

To test our solution for the rescheduling problem, we used 4 different results obtained by the scheduler in the Scenario A. In particular, we considered up to 4 patients of the specialty 4 requiring to postpone their surgery.

In all the 4 instances tested, with 1, 2, 3, and 4 patients requiring to postpone their surgery, the rescheduler is able to reassign a day of treatment to all the patients assigned in the old schedule, without dropping any registration.

Moreover, the rescheduler is able to assign the same day to each patient not involved in the changes in the scenario with 1, 2 and, 3 patients requiring to postpone their surgery.

5. Conclusions and Related Work

In this paper we have employed ASP for solving the ORS problem with ICU and PACU management, given ASP has already proved to be a viable tool for solving scheduling problems. Specifications of the problem are modularly expressed as rules in the ASP encoding, and the ASP solver clingo has been used. We finally presented the results of a preliminary experimental analysis on ORS benchmarks with realistic sizes and parameters on two scenario, and tested a solution to the rescheduling problem. Our work is different from other works in the literature dealing with PACU, such as [20], since in our model we schedule patients not only taking into account PACU, but also other units such as ICU and multiple specialties, moreover, we proposed a solution to the rescheduling problem and. Future work includes the design and analysis of more scenarios, involving larger hospitals and the shared usage of beds in ICU and PACU. We are currently working on extending our preliminary experiments. Moreover, we would like also to implement and test other solving procedures, e.g., [21, 22, 23, 24], considering the relation between ASP and SAT procedures [25, 26], whose goal would be to improve the current results. Finally, we plan to add this solution into a platform of solutions for scheduling problems in healthcare, similarly to, e.g., [27] in the context of SMT solving.

References

[1] N. Meskens, D. Duvivier, A. Hanset, Multi-objective operating room scheduling considering desiderata of the surgical team, Decis. Support Syst. 55 (2013) 650–659. URL: https://doi.org/10.1016/j.dss.2012.10.019. doi:10.1016/j.dss.2012.10.019.

[2] A. Abedini, H. Ye, W. Li, Operating room planning under surgery type and priority constraints, Procedia Manufacturing 5 (2016) 15–25.

[3] R. Aringhieri, P. Landa, P. Soriano, E. Tànfani, A. Testi, A two level metaheuristic for the operating room scheduling and assignment problem, Computers & Operations Research 54 (2015) 21–34.

[4] M. Hamid, M. M. Nasiri, F. Werner, F. Sheikhahmadi, M. Zhalechian, Operating room scheduling by considering the decision-making styles of surgical team members: A comprehensive approach, Computers & Operation Research 108 (2019) 166–181.

[5] G. Brewka, T. Eiter, M. Truszczynski, Answer set programming at a glance, Communications of the ACM 54 (2011) 92–103.

[6] M. Gelfond, V. Lifschitz, The stable model semantics for logic programming, in: Proceedings of the Fifth International Conference and Symposium , Seattle, Washington, August 15-19, 1988 (2 Volumes), MIT Press, 1988, pp. 1070–1080.

[7] M. Gelfond, V. Lifschitz, Classical Negation in Logic Programs and Disjunctive Databases, New Generation Comput. 9 (1991) 365–386.

[8] C. Dodaro, G. Galatà, M. Maratea, I. Porro, Operating room scheduling via answer set programming, in: AI*IA, volume 11298 of *LNCS*, Springer, 2018, pp. 445–459.

[9] C. Dodaro, G. Galatà, M. Maratea, I. Porro, An ASP-based framework for operating room scheduling, Intelligenza Artificiale 13 (2019) 63–77.

[10] C. Dodaro, M. Maratea, Nurse scheduling via answer set programming, in: LPNMR, volume 10377 of *LNCS*, Springer, 2017, pp. 301–307.

[11] M. Alviano, C. Dodaro, M. Maratea, An advanced answer set programming encoding for nurse scheduling, in: AI*IA, volume 10640 of *LNCS*, Springer, 2017, pp. 468–482.

[12] F. Calimeri, W. Faber, M. Gebser, G. Ianni, R. Kaminski, T. Krennwallner, N. Leone, M. Maratea, F. Ricca, T. Schaub, Asp-core-2 input language format, Theory and Practice of Logic Programming 20 (2020) 294–309.

[13] M. Gebser, B. Kaufmann, T. Schaub, Conflict-driven answer set solving: From theory to practice, Artificial Intelligence 187 (2012) 52–89.

[14] M. Alviano, G. Amendola, C. Dodaro, N. Leone, M. Maratea, F. Ricca, Evaluation of disjunctive programs in WASP, in: M. Balduccini, Y. Lierler, S. Woltran (Eds.), LPNMR, volume 11481 of *LNCS*, Springer, 2019, pp. 241–255.

[15] F. Calimeri, M. Gebser, M. Maratea, F. Ricca, The design of the fifth answer set programming competition, CoRR abs/1405.3710 (2014). URL: http://arxiv.org/abs/1405.3710. arXiv:1405.3710.

[16] M. Gebser, M. Maratea, F. Ricca, The design of the seventh answer set programming competition, in: M. Balduccini, T. Janhunen (Eds.), LPNMR, volume 10377 of *Lecture Notes in Computer Science*, Springer, 2017, pp. 3–9.

[17] M. Gebser, M. Maratea, F. Ricca, The seventh answer set programming competition: Design and results, Theory Pract. Log. Program. 20 (2020) 176–204.

[18] C. Dodaro, G. Galatà, M. K. Khan, M. Maratea, I. Porro, An ASP-based solution for operating room scheduling with beds management, in: RuleML+RR, volume 11784 of *Lecture Notes in Computer Science*, Springer, 2019, pp. 67–81.

[19] M. Gebser, R. Kaminski, B. Kaufmann, M. Ostrowski, T. Schaub, P. Wanko, Theory solving made easy with clingo 5, in: ICLP (Technical Communications), volume 52 of *OASICS*, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016, pp. 2:1–2:15.

[20] B. King, A. Leach, M. Platt, D. White, Scheduling surgical operations and the post-

anesthesia care unit using work tours and binary programming, AIMS International Journal of Management 11 (2017) 49.

[21] E. Giunchiglia, M. Maratea, A. Tacchella, D. Zambonin, Evaluating search heuristics and optimization techniques in propositional satisfiability, in: R. Goré, A. Leitsch, T. Nipkow (Eds.), IJCAR, volume 2083, Springer, 2001, pp. 347–363.

[22] E. Giunchiglia, M. Maratea, A. Tacchella, Dependent and independent variables in propositional satisfiability, in: S. Flesca, S. Greco, N. Leone, G. Ianni (Eds.), JELIA, volume 2424 of *Lecture Notes in Computer Science*, Springer, 2002, pp. 296–307.

[23] E. Giunchiglia, M. Maratea, A. Tacchella, (In)Effectiveness of look-ahead techniques in a modern SAT solver, in: F. Rossi (Ed.), CP, volume 2833 of *Lecture Notes in Computer Science*, Springer, 2003, pp. 842–846.

[24] E. D. Rosa, E. Giunchiglia, M. Maratea, A new approach for solving satisfiability problems with qualitative preferences, in: M. Ghallab, C. D. Spyropoulos, N. Fakotakis, N. M. Avouris (Eds.), ECAI, volume 178 of *Frontiers in Artificial Intelligence and Applications*, IOS Press, 2008, pp. 510–514.

[25] E. Giunchiglia, M. Maratea, On the Relation Between Answer Set and SAT Procedures (or, Between cmodels and smodels), in: ICLP, volume 3668 of *LNCS*, Springer, 2005, pp. 37–51.

[26] E. Giunchiglia, N. Leone, M. Maratea, On the relation among answer set solvers, Ann. Math. Artif. Intell. 53 (2008) 169–204.

[27] A. Armando, C. Castellini, E. Giunchiglia, M. Idini, M. Maratea, TSAT++: an open platform for satisfiability modulo theories, Electronic Notes in Theoretical Computer Science 125 (2005) 25–36.