Nuclear Medicine Rescheduling Problem: A Logic-based Approach

Carmine Dodaro^{*I*}, Giuseppe Galatà², Marco Maratea^{*I*,∗}, Cinzia Marte^{*I*,∗} and Marco Mochi³

¹DeMaCS, University of Calabria, Rende, Italy ²SurgiQ srl, Genova, Italy ³DIBRIS, University of Genoa, Genoa, Italy

Abstract

The Nuclear Medicine Scheduling problem consists of assigning patients to a day, in which the patient will undergo the medical check, the preparation, and the actual image detection process. The schedule of the patients should consider their different requirements and the available resources, e.g., varying time required for different diseases and radiopharmaceuticals used, number of injection chairs, and tomographs available. Recently, this problem has been solved using a logic-based approach utilizing the Answer Set Programming (ASP) methodology. However, it may be the case that a computed schedule can not be implemented due to a sudden emergency and/or unavailability of resources, thus rescheduling is needed.

In this paper we present an ASP-based approach to solve such situation, that we call Nuclear Medicine Rescheduling problem. Experiments employing real data from a medium size hospital in Italy show that our rescheduling solution provides satisfying results even when the concurrent number of emergencies and unavailability is significant.

Keywords

Answer Set Programming, Logic Programming, Digital Health

1. Introduction

Nuclear Medicine is a medical specialty that uses radiopharmaceuticals, a particular kind of drug containing radioactive elements, to treat or diagnose diseases. According to data by the Italian Ministry of Health, almost 2 millions nuclear medicine exams have been carried on during 2022 in Italy^{[1](#page-0-0)}. The process of treating patients with this technique is complex since it involves multiple resources of the hospital and requires multiple steps of varying time. Moreover, often these drugs contain radioactive elements characterized by short half-lives, meaning that they decay rapidly after their preparation. Thus, the timing should be as precise as possible in order to obtain images of good quality. Addressing this problem effectively is crucial due to the nature of the diagnosed illnesses and treated through nuclear medicine, alongside the significant costs associated with this kind of technique. An efficient, possibly optimal, solution can reduce the waiting time of the patients and can thus increase the effective utilization of the resources, avoiding waste of time and resources. Nevertheless, reducing the unnecessary time spent by the patients in the hospital is vital for increasing their satisfaction.

Thus, the Nuclear Medicine Scheduling (NMS) problem consists of assigning patients to a day, in which the patient will undergo the medical check, the preparation, and the actual image detection process. The schedule of the patients considers the different requirements of the patients and the available resources, e.g., varying time required for different procedures and radiopharmaceuticals used, number of injection

ceur-ws.org ISSN 1613-0073

HC@AIxIA 2024: Third AIxIA Workshop on Artificial Intelligence For Healthcare, November 25–28, 2024, Bolzano, Italy [∗]Corresponding author.

[†] These authors contributed equally.

 \bigcirc carmine.dodaro@unical.it (C. Dodaro); giuseppe.galata@surgiq.com (G. Galatà); marco.maratea@unical.it (M. Maratea); cinzia.marte@unical.it (C. Marte); marco.mochi@edu.unige.it (M. Mochi)

[0000-0002-5617-5286](https://orcid.org/0000-0002-5617-5286) (C. Dodaro); [0000-0002-1948-4469](https://orcid.org/0000-0002-1948-4469) (G. Galatà); [0000-0002-9034-25276](https://orcid.org/0000-0002-9034-25276) (M. Maratea); [0000-0003-3920-8186](https://orcid.org/0000-0003-3920-8186) (C. Marte); [0000-0002-5849-3667](https://orcid.org/0000-0002-5849-3667) (M. Mochi)

^{© 2024} Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

¹https://www.salute.gov.it/imgs/C_17_pubblicazioni_3425_allegato.pdf

chairs and tomographs available. We followed the definition of the problem given by Medipass^{[2](#page-1-0)}, leading provider of technological innovation across cancer care and diagnostic imaging in Italy, in collaboration with Surgi $Q³$ $Q³$ $Q³$, an Italian company active in planning and scheduling solutions.

Given that complex combinatorial problems, possibly involving optimizations, such as the NMS problem, are usually the target applications of logic-based approaches, in [\[1\]](#page-11-0) a solution based on the AI language for knowledge representation and reasoning Answer Set Programming (ASP) has been provided. In the ASP methodology, first the problem is formally represented in terms of logical rules (usually called encoding), then the encoding in given in input, together with the data, to a solver, which computes a solution to the original problem. The reason for employing ASP are multiple, other than it is the language we know better, and a number of scheduling applications, also in Healthcare, have been already successfully addressed with ASP (see, e.g., [\[2,](#page-11-1) [3,](#page-11-2) [4,](#page-11-3) [5,](#page-11-4) [6,](#page-11-5) [7\]](#page-11-6)): A simple but rich syntax [\[8\]](#page-11-7), which includes optimization statements as well as powerful database-inspired constructs like aggregates, an intuitive semantics [\[9\]](#page-11-8), the availability of efficient solvers (see, e.g., [\[10,](#page-11-9) [11,](#page-11-10) [12\]](#page-11-11)) able to solve optimization problems via powerful algorithms (see, e.g., $[13, 14]$ $[13, 14]$ $[13, 14]$, and the fruitful combination with machine learning approaches (see, e.g., [\[15,](#page-11-14) [16\]](#page-12-0)).

However, it may be well the case that a previously viable solution may no longer be feasible for a number of reasons which include sudden emergencies coming to the table and/or the unavailability of resources. In turn, these scenarios can produce two different types of rescheduling: If the reason is known prior, the rescheduling is called 'offline', since it does not affect the patients already in the hospital, otherwise it is 'online', since it is created while the patients are in the hospital, but without affecting the patients already under treatment.

In this paper we present an ASP-based approach to solve such problem, that we call Nuclear Medicine Rescheduling (NMR) problem. In particular, we address online rescheduling for the reallocation of resources due to emergencies that demand immediate access to equipment or require more time on the resources. Experiments employing real data from a medium size hospital in Italy, provided by Medipass, show that the solution produces satisfying results even when the concurrent number of emergencies and unavailability is up to 20% of the patients involved in the scheduling procedure.

The paper is structured as follows. Sections [2](#page-1-2) introduces needed preliminaries about ASP. Then, Section [3](#page-3-0) reviews the NMS problem, in terms of an informal description of the problem and the ASP encoding for representing it. Section [4](#page-5-0) and [5](#page-7-0) present the main contributions of our paper, i.e., the NMR problem, in terms of informal description and ASP encoding, and its experimental evaluation, respectively. The paper ends by discussing related works and conclusions in Section [6](#page-10-0) and [7,](#page-10-1) respectively.

2. Background on ASP

Answer Set Programming (ASP) [\[9\]](#page-11-8) is a programming paradigm developed in the field of non-monotonic reasoning and logic programming. In this section, we overview the language of ASP. More detailed descriptions and a more formal account of ASP, including the features of the language employed in this paper, can be found in [\[9,](#page-11-8) [8\]](#page-11-7). Hereafter, we assume the reader is familiar with logic programming conventions.

Syntax. The syntax of ASP is similar to the one of Prolog. Variables are strings starting with an uppercase letter, and constants are non-negative integers or strings starting with lowercase letters. A *term* is either a variable or a constant. A *standard atom* is an expression $p(t_1,...,t_n)$, where p is a *predicate* of arity *n*, denoted by p/n , and t_1, \ldots, t_n are terms. An atom $p(t_1, \ldots, t_n)$ is ground if t_1, \ldots, t_n are constants. A *ground set* is a set of pairs of the form ⟨*consts*: *con j*⟩, where *consts* is a list of constants and *con j* is a conjunction of ground standard atoms. A *symbolic set* is a set specified syntactically as ${Terms_1 : Conj_1; \cdots ; Terms_t : Conj_t}$, where $t > 0$, and for all $i \in [1, t]$, each *Terms_i* is a list of terms such that $|Terms_i| = k > 0$, and each *Con j_i* is a conjunction of standard atoms. A *set term* is either a

²https://ergeagroup.com/it/.

³[https://surgiq.com/.](https://surgiq.com/)

symbolic set or a ground set. Intuitively, a set term $\{X : a(X, c), p(X); Y : b(Y, m)\}$ stands for the union of two sets: the first one contains the *X*-values making the conjunction $a(X, c)$, $p(X)$ true, and the second one contains the *Y*-values making the conjunction $b(Y, m)$ true. An *aggregate function* is of the form *f*(*S*), where *S* is a set term, and *f* is an *aggregate function symbol*. Basically, aggregate functions map multisets of constants to a constant, e.g., the function *#count* computes the number of terms.

An *aggregate atom* is of the form $f(S) \prec T$, where $f(S)$ is an aggregate function, $\prec \in \{ \leq, \leq, >, \geq, \neq \}$, $=$ } is a operator, and *T* is a term called guard. An aggregate atom *f*(*S*) \prec *T* is ground if *T* is a constant and *S* is a ground set. An *atom* is either a standard atom or an aggregate atom. A *rule r* has the following form:

$$
a_1 | \ldots | a_n := b_1, \ldots, b_k, not \ b_{k+1}, \ldots, not \ b_m.
$$

where a_1, \ldots, a_n are standard atoms, b_1, \ldots, b_k are atoms, b_{k+1}, \ldots, b_m are standard atoms, and $n, k, m \ge 0$. A literal is either a standard atom *a* or its negation *not a*. The disjunction $a_1 | \ldots | a_n$ is the *head* of *r*, while the conjunction b_1, \ldots, b_k , *not* b_{k+1}, \ldots, not b_m is its *body*. Rules with empty body are called *facts*. Rules with empty head are called *constraints*. A variable that appears uniquely in set terms of a rule *r* is said to be *local* in *r*, otherwise it is a *global* variable of *r*. An ASP program is a set of *safe* rules, where a rule *r* is *safe* if the following conditions hold: *(i)* for each global variable *X* of *r* there is a positive standard atom ℓ in the body of r such that X appears in ℓ , and *(ii)* each local variable of r appearing in a symbolic set {*Terms*:*Conj*} also appears in a positive atom in *Conj*.

A *weak constraint* [\[17\]](#page-12-1) ω is of the form:

$$
:\sim b_1,\ldots,b_k, not\ b_{k+1},\ldots, not\ b_m.\ [w@l]
$$

where *w* and *l* are the weight and level of ω, respectively. (Intuitively, [*w*@*l*] is read as "weight *w* at level *l*", where the weight is the "cost" of violating the condition in the body of *w*, whereas levels can be specified for defining a priority among preference criteria). An ASP program with weak constraints is $\Pi = \langle P, W \rangle$, where *P* is a program and *W* is a set of weak constraints.

A standard atom, a literal, a rule, a program or a weak constraint is *ground* if no variables appear in it.

Semantics. Let *P* be an ASP program. The *Herbrand universe U^P* and the *Herbrand base B^P* of *P* are defined as usual. The ground instantiation G_P of P is the set of all the ground instances of rules of P that can be obtained by substituting variables with constants from *UP*.

An *interpretation I* for *P* is a subset *I* of B_P . A ground literal ℓ (resp., *not* ℓ) is true w.r.t. *I* if $\ell \in I$ (resp., $\ell \notin I$), and false (resp., true) otherwise. An aggregate atom is true w.r.t. *I* if the evaluation of its aggregate function (i.e., the result of the application of *f* on the multiset *S*) with respect to *I* satisfies the guard; otherwise, it is false.

A ground rule *r* is *satisfied* by *I* if at least one atom in the head is true w.r.t. *I* whenever all conjuncts of the body of *r* are true w.r.t. *I*.

A model is an interpretation that satisfies all rules of a program. Given a ground program *G^P* and an interpretation *I*, the *reduct* [\[18\]](#page-12-2) of G_P w.r.t. *I* is the subset G_P^I of G_P obtained by deleting from G_P the rules in which a body literal is false w.r.t. *I*. An interpretation *I* for *P* is an *answer set* (or stable model) for *P* if *I* is a minimal model (under subset inclusion) of G_P^I (i.e., *I* is a minimal model for G_P^I) [\[18\]](#page-12-2).

Given a program with weak constraints $\Pi = \langle P, W \rangle$, the semantics of Π extends from the basic case defined above. Thus, let $G_{\Pi} = \langle G_P, G_W \rangle$ be the instantiation of Π ; a constraint $\omega \in G_W$ is violated by an interpretation *I* if all the literals in ω are true w.r.t. *I*. An *optimum answer set* for Π is an answer set of *G^P* that minimizes the sum of the weights of the violated weak constraints in *G^W* in a prioritized way.

Syntactic shortcuts. In the following, we also use *choice rules* of the form $\{p\}$, where *p* is an atom. Choice rules can be viewed as a syntactic shortcut for the rule $p \mid p'$, where p' is a fresh new atom not appearing elsewhere in the program, meaning that the atom *p* can be chosen as true.

Table 1

Specifications for each protocol, including the number of time slots (TS) needed for each phase, the total time slots for the entire protocol, and the chair request.

3. Nuclear Medicine Scheduling

In this section, we review the Nuclear Medicine Scheduling (NMS) problem [\[1\]](#page-11-0). We start by outlining the key aspects of the problem, followed by a detailed explanation of the ASP encoding developed to model it.

3.1. Problem Description

The NMS problem consists of assigning patients to a day and to a tomograph and/or an injection chair if required by the patient or the specific procedure. In our problem, for each day we consider a set of 120 time slots (TS), each representing 5 minutes. Each patient needs an exam and each exam is linked to a protocol defining the phases and the time required for each phase. We considered 11 different protocols. Each protocol can encompass up to four phases, represented as (p_1) anamnesis, (p_2) medical check, (*p*3) radiopharmaceuticals injection and bio-distribution time, and (*p*4) image detection. Moreover, each phase can require a different amount of time depending on the exam. Table [1](#page-3-1) shows the total time needed by each protocol, the partial time required by each phase, expressed in the number of time slots used, and which protocol requires an infusion chair for the phase (p_3) .

Due to the high number of phases required by each patient and the variety of the considered protocols, in many clinics the schedule of the patients is sub-optimal. A sub-optimal schedule is problematic not only because of the high cost of the drugs and machines involved in the exams, but is particularly detrimental for the patients since the order and the time required by each phase, in particular the injection and the bio-distribution time, is fundamental for a proper image detection.

Different clinics have different resource availability and may have different requirements in defining a proper solution. Here we present the criteria followed in the clinic that provided us with the real data of the patients and that we use to define the problem. We considered a clinic with two rooms, each with one tomograph and three injection chairs. We started from a list of patients, each requiring a specific protocol, to be assigned in a day. A proper solution must satisfy the following conditions:

- a starting and an ending time should be assigned to every scheduled patient for each required phase;
- there must be at most two patients concurrently in the medical check phase;
- the injection phase must be done in an injection chair or on a tomograph according to the required protocol;
- each injection chair and tomograph can be used by just one patient at the same time;
- patients requiring an injection chair must be assigned to the tomograph of the same room;
- protocol identified by the id 815 cannot be assigned on the same day and tomograph for more than one patient.

The solution should maximize the number of scheduled patients in the considered days and, to increase the satisfaction of the patients and the effectiveness of the exams, the solution should also try to minimize the unnecessary time spent in the clinic by the patients.

3.2. ASP Encoding

Now we present our ASP encoding developed to model the NMS problem. The underlying encoding is based on the input language of CLINGO [\[10\]](#page-11-9).

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

- instances of $reg(ID, D, PrID)$ represent a registration with identification number ID, on day D, for a specific exam with protocol number PrID;
- instances of avail (TS, D) denote that the time slots TS is available on day D;
- instances of exam(PrID, P, NumTS) denote the features of a exam, where PrID denotes the exam protocol number, P indicates the phase, and NumTS specifies the time required for that phase in terms of the number of time slots;
- instances of tomograph(T,R) and chair(C,R) denote the allocation of the tomograph T and the chair C to the room R, respectively;
- instances of required_chair(PrID) denote the necessity of a chair for the phases 1 and 2 for the protocol PrID;
- instances of cost(PrID, NumTS) represent the total duration in terms of time slots, denoted as NumTS, of the phases within the protocol identified by PrID;
- instances of $\text{limit}(\text{PrID},N)$ denote the maximum number N of exams with protocol number PrID that can be executed on a fixed tomograph in a day.

The output consists of assignments represented by the atom $x(ID, D, TS, PrID, P)$ where the intuitive meaning is that the registration with identification number ID for the exam with protocol number PrID, regarding the phase P, has been scheduled for the day D during the time slot TS. Additionally, it includes atoms chair(C,ID,D) and tomograph(T,ID,D), denoting the resource (either the chair C or the tomograph T, respectively) allocated to the patient ID on the day D.

Encoding. The related encoding is shown in Figure [1](#page-5-1) and is described next. To simplify the description, we denote as r_i the rule appearing at line i of Figure [1.](#page-5-1)

Rule r_1 may assign or not the registration with identifier ID to a specific time slot TS for the day D in phase 0, i.e., the phase of the anamnesis. Rule r_2 assigns an already scheduled session for a given phase P to the subsequent planned phases, under the condition that the start of the phase does not extend beyond the latest available time slot for a session on that day. Furthermore, it ensures that the subsequent phase starts at most 5 time slots after the ending of the previous phase. Rule r_3 ensures that the duration of the final phase is also consistent with the time slots, ensuring that all phases are completed within the specified limit. Rule r_4 keeps track of the time slots allocated to a patient during phase 0 via the auxiliary atom timeAnamnesis(ID, TS). Rule $r₅$ restricts the number of patients during the anamnesis phase to a maximum of two. Rule r_6 produces the auxiliary atom timeOccupation(ID, D, TS, END, PrID), representing the duration needed for each patient ID from the initial time slot TS of phase 1 to the final one END of phase 2, concerning the protocol PrID on the day D. Rule r_7 produces the auxiliary atom res (ID, D, TS, 0) for each time slot derived from the previous rule. Specifically, the constant 0 denotes that a chair is required for each of these time slots. Rules r_8 and r_9 produce the atom res(ID,D,TS,1), which differs from the previous one for the constant 1, indicating the use of a tomograph. From rule r_8 , it is inferred that a tomograph is employed during phase 3, whereas rule $r₉$ indicates the tomograph's usage in phase 1 and 2, according to the atom timeOccupation. Rule *r*¹⁰ ensures that the limit of protocols that can be executed on a single tomograph is respected. Rule r_{11} produces the atom chair(C, ID,D) representing the assignment of a chair C on the day D to the patient ID when the protocol PrID requires

```
1 0 {x(ID, D, TS, PrID, 0) : avail(TS, D)} 1 :- reg(ID, D, PrID).
2 \{X(ID, D, START, PrID, P+1) : avail(START, D), START >= TS+NumTS, START <TS+NumTS+6} = 1 :- x(ID, D, TS, PrID, P), exam(PrID, P, NumTS), P \ge 0, P < 3.
3 : -x(ID, , S, PrID, 3), exam(PrID, 3, NumTS), TS + NumTS > 120.4 timeAnamnesis(ID, TS..TS+NumTS-1) :- x(ID, D, TS, PrID, 0), exam(PrID, 0, NumTS).
5 : - #count[ID: timeAnammesis(ID, TS) > 2, avail(TS,D).6 timeOccupation(ID, D, TS, END-1, PrID) :- x(ID, D, TS, PrID, 1), x(ID, D, END,
     PrID, 3).
7 res(ID, D, TS..END,0) :- timeOccupation(ID, D, TS, END, PrID),
     required_chair(PrID).
8 \text{ res(ID, D, TS.}.TS+NumTS-1,1) :- x(ID, D, TS. PrID, 3), exam(PrID, 3, NumTS),required_chair(PrID).
9 res(ID, D, TS..END+NumTS-1,1) :- timeOccupation(ID, D, TS, END, PrID), exam(PrID,
     3, NumTS), not required_chair(PrID).
10 :- #count{ID: tomograph(T, ID, D), x(ID, D, _, PrID, _)} > N, limit(PrID, N),
     tomograph(T, _-).
11 {chair(C, ID, D) : chair(C, -)} 1 :- x(ID, D, -, PrID, -), required_chair(PrID).
12 1 {tomograph(T, ID, D) : tomograph(T, _{})} 1 :- x(ID, D, _, PrID, _{}).
13 :- chair(C, ID, D), tomograph(T, ID, D), chair(C, R1), tomograph(T, R2), R1 != R2.
14 chair(C, ID, D, TS) :- chair(C, ID, D), res(ID, D, TS, 0).
15 tomograph(T, ID, D, TS) :- tomograph(T, ID, D), res(ID, D, TS, 1).
16 :- \#\text{count} {ID: tomograph(T, ID, D, TS)} > 1, tomograph(T, ), avail(TS,D).
17 : -\#count\{ID : chair(C, ID, D, TS)\} > 1, chair(C, _), avail(TS,D).18 :∼ not x(ID, D, _, _,0), reg(ID, D, _). [1@2, ID, D]
19 :∼ x(ID, _, START, PrID, 0), x(ID, _, END, _, 3), cost(PrID, NumTS), END - START
      - NumTS > = 0. [END - START - NumTS@1, ID]
```
Figure 1: ASP encoding of the NMS problem.

a chair. Rule r_{12} produces the atom tomograph(T, ID,D) similar to the previous but considering the tomographs. representing the assignment of a tomograph T on the day D to the patient ID. Rule r_{13} prevents the patient who moves from the chair to the tomograph from changing room. Rule *r*¹⁴ and r_{15} generate the atoms chair(C,ID,D,TS) and tomograph(T,ID,D,TS) respectively, indicating the time slots TS during which the chair C and tomograph T are utilized by the patients ID on the day D. Rule r_{16} and r_{17} ensure that at most one patient is assigned to each tomograph and chair in every time slot, respectively. Finally, the optimal solution is achieved through the application of rule r_{18} , which minimizes (with the highest priority) the number of registrations not assigned to a schedule, and rule r_{19} , which minimizes the duration of patient appointments beyond the time necessary to perform the test.

4. Nuclear Medicine Rescheduling

In this section, we explore scenarios where patient appointments need to be rescheduled. We start by describing the core aspects of the Nuclear Medicine Rescheduling (NMR) problem and, then, we introduce the encoding designed to model the problem.

4.1. NMR Problem Description

As described in Section [3.1,](#page-3-2) the NMS problem involves assigning patients to specific days for their medical checks, preparation, and image detection process. Given the complexity of protocols and the multiple phases each patient must undergo, the precise timing and sequence of these phases—especially the injection and bio-distribution periods—are critical for accurate imaging results. Hence, it is crucial to effectively navigate unforeseen challenges and adapt to changes. This is especially true when addressing rescheduling issues, where a previously viable solution may no longer be feasible. Common scenarios

that may disrupt a previously established schedule are:

- (*i*) the reallocation of resources due to emergencies that demand immediate access to equipment or require more time on the resources;
- (*ii*) unexpected unavailability of essential resources (e.g., an injection chair or a tomography);
- (*iii*) unavailability of a procedure room.

Moreover, these scenarios can produce two different types of rescheduling. If the reason, e.g., the unavailability of some resources or of some procedure room, is known prior, the rescheduling is called 'offline', since it does not affect the patients already in the hospital and it produces a new output before patients arrive in the structure. Instead, suppose the reason cannot be known prior: In this case, e.g., unavailability of the procedure room depends on a larger amount of time required by some patient or some emergency comes into the picture, the rescheduling is called 'online' since it is created while the patients are in the hospital but without affecting the patients already under treatment.

In general, these situations require rescheduling the patients to ensure the continuity of medical processes, safeguarding both the quality of patient care and the accuracy of diagnostic outcomes. Moreover, to ensure that all the patients are treated on the originally scheduled day, it is possible to allow overtime usage. To address these challenges, a responsive rescheduling solution is essential. Such a solution should be capable of quickly adjusting to changes by reallocating resources. In the following, we will present an online rescheduling solution for scenario (*i*). The solution, that is presented in Section [4.2,](#page-6-0) is developed to be general enough to be adapted to the other scenarios with relatively few modifications. More in detail, in scenario (*i*) registrations could require more time than expected to complete their phases. Additionally, new registrations having an emergency, not necessarily starting from phase 0, could arise and require immediate assignment. The resulting online rescheduling process should prioritize the following desiderata, in descending order:

- assigning as soon as possible the new registrations having an emergency;
- minimizing the difference between the old and the new starting time of the treatment for each rescheduled patient;
- minimizing the time slots assigned in overtime;
- minimizing the difference between the old and the new assigned resources to the patients.

4.2. ASP Encoding

Here we present the encoding of the NMR problem. The underlying encoding is based on the input language of CLINGO [\[10\]](#page-11-9).

Data Model. The input data comprises the input and output atoms detailed in Section [3.2,](#page-4-0) excluding the reg atoms, and supplemented by the following additional atoms:

- instances of new regs(ID,RequiredTS, P, PrID) represent new registrations to be assigned, where ID is the identification number, which should be scheduled as close as possible to the time slot RequiredTS, P is the required phase, and PrID is the protocol number to be followed;
- instances of new exam(ID,P,NumTS) denote the delay of a registration with identification number ID, where phase P requires an additional NumTS time slots to complete the treatment.

The output is the rescheduling, consisting of atoms similar to x, tomograph, and chair detailed in Section [3.2,](#page-4-0) but denoted as y, Y_tomograph, and Y_chair, respectively.

```
1 block(TS_MIN) :- #min{TS: new_reg(ID, D, TS, P, PrID); TS: x(ID, D, TS, PrID, P),
     exam_new(ID, P, NumTS) = TS_MIN.
2 block(RID, P) :- x(ID, D, TS, P, PrID), block(TS_MIN), not exam_new(ID, _, _), TS
     < TS_MIN.
3 \{y(ID, D, TS, PrID, P) : avail(TS, D), TS >= RequiredTS\} = 1 :- new_reg(ID, D,RequiredTS, P, PrID).
4 \{y(ID, D, TSN, PrID, 0) : avail(TSN, D), TSN \geq TS\} = 1 : - x(ID, D, TS, PrID,0), not block(ID, 0).
5 \{y(ID, D, START, PCID, P+1) : avail(START,D), START >= TS+NumTS\} = 1 :- y(ID, D,TS, PrID, P), exam(PrID, P, NumTS), P \ge 0, P \le 3, not block(ID, P).
6 \{y(ID, D, START, PCID, P+1) : avail(START, D), START >= TS+NumTS\} = 1 :- y(ID, D, D, T)TS, PrID, P), exam_new(ID, P, NumTS), P \ge 0, P < 3.
7 y(ID, D, TS, PrID, P) :- x(ID, D, TS, PrID, P), block(ID, P).
8 : -y(ID, D, TS_N, -, P), x(ID, D, TS_0, -, P), TS_N < TS_0.9 :∼ y(ID, D, TS, _, P), new_reg(ID, D, RequiredTS, _, P). [TS-RequiredTS@4, ID]
10 :∼ y(ID, D, TS_N, _, P), x(ID, D, TS_O, _, P). [TS_N-TS_O@3, ID]
11 :∼ res(ID, D, TS, _), TS > 120. [1@2, ID, TS]
12 :∼ not Y_tomograph(T, ID, _, _), tomograph(T, ID, _, _). [1@1, ID]
13 :∼ not Y_chair(C, ID, _, _), chair(C, ID, _, _). [1@1,ID]
```
Figure 2: ASP encoding of the rescheduling problem.

Encoding. The ASP encoding for the rescheduling problem comprises the thirteen rules shown in Figure [2,](#page-7-1) along with fourteen rules derived from the encoding detailed in Figure [1.](#page-5-1) The latter is obtained by creating new rules from the corresponding rules r_4, \ldots, r_{17} , by replacing x, tomograph, and chair with y, Y_tomograph and Y_chair, respectively. To simplify the description of encoding reported in Figure [2,](#page-7-1) we denote as r_i' the rule appearing at line *i* of Figure [2.](#page-7-1) Rule r_1' derives an auxiliary atom that gets the first time slot where a new registration or the delay of a registration appears. Then, in rule r'_2 another auxiliary atom is derived getting all the registrations and phases starting before the derived time slot. Rule r'_3 assigns a starting time for the first phase to all the new registrations, ensuring that the assigned starting time is after the required time slot. Rules r'_4 and r'_5 assign the first and the subsequent phases to the previously assigned patients that are not involved in delays and were assigned after the first delay or arrival of a new registration. Rule r'_6 assigns a starting time for all the phases of the registrations involved in the delays, ensuring that the subsequent phases are assigned following the new required time. Rule r'_7 assigns the same starting time as in the original schedule to the registrations and phases previously derived by the auxiliary atom. Rule r'_{8} ensures that all the starting times assigned in the reschedule are not before the previously assigned ones. Finally, rules from r'_9 to r'_{13} are weak constraints that are used to reach the optimal solution. In particular, the weak constraints, ensure that the optimal solution minimizes the same objectives as presented in the list in Section [4.1](#page-5-2) in a prioritized way.

5. NMR Preliminary Experimental Results

In this section, we report the results of an empirical analysis of the NMR problem via ASP. We performed experiments on an Apple M1 CPU @ 3.22 GHz machine with 8 GB of physical RAM. The ASP system used was CLINGO [\[10\]](#page-11-9) 5.6.2, using parameters *--opt--strategy=usc* for faster optimization and *--parallelmode 4* for parallel execution: We conducted a preliminary analysis with various options, and these parameters were to be the most effective. The ASP encoding and the instances employed in this section can be found at: [https://github.com/MarcoMochi/HC2024NMR.](https://github.com/MarcoMochi/HC2024NMR)

5.1. NMR benchmarks

We used real data coming from a medium size hospital, provided by Medipass, to compute a solution for the NMS problem. More in detail, we tested instances of more than a year of daily exams. In particular,

Table 2

Analysis of the time required to compute rescheduling considering an increasing number of new_regs (rows) and new_exam (columns). Each cell displays the time required, measured in seconds, for a scenario with a low/medium/high number of registrations in input.

we tested 366 instances, each corresponding to weekdays, resulting in a total of 72 weeks. The solution schedules the patients in a day in a range of 10 hours, split into 120 time slots of 5 minutes. Each patient is linked to one of the possible exams. In particular, protocol "823" is required by more than 85% of the patients, thus, the majority of the patients need an exam protocol that requires 2 time slots for the anamnesis and other 2 time slots for medical preparation, 10 time slots for the drug injection and the bio-distribution time and, at last, the image detection requires 7 time slots. The other patients can be associated with one of the other 11 possible protocols. The schedule is done having as available resources two rooms for the radiotherapy. In each room, there is a tomograph and 3 chairs. The number of patients requiring exam changes every day but, on average, there are 29 patients to be scheduled, with a maximum of 37 patients.

These scheduling solutions were then utilized as part of the input for the NMR problem. We randomly selected three schedules representing three different scenarios that we called: low, medium, and high. These scenarios represent three situations in which the schedule assigned the treatments to a low, a medium, or a high number of registrations, meaning that the reschedule has fewer or more time slots to rearrange the registrations as needed. For each scenario, we generated 9 instances in total: 3 instances with an increasing number of new registrations with an emergency, 3 with an increasing number of patients with delays, and 3 instances with both kinds, for a total of 27 instances. To simulate the factors that require a rescheduling, we proceed as follows: For a new registration with an emergency, we randomly select a protocol among the possible ones, a preferred time slot, and the starting phase of the registration. For patients with delay, we randomly select a scheduled patient and a required phase and assign a new treatment duration needed to complete the phase.

5.2. Results

We present the results obtained from testing our solution to the NMR problem, obtained through the usage of the ASP encoding presented in Section [4.2](#page-6-0) and using the instances generated as discussed in Section [5.1.](#page-7-2) Our analysis comprises an examination of the time required to compute the solution and an evaluation of its quality.

Time efficiency. We report the analysis of the performances of our solution in Table [2,](#page-8-0) which details the time required to optimally solve all the considered instances. Each row indicates the number of new registrations, while the columns represent the number of patients experiencing delays. As shown in the table, our solution is able to optimally solve all the instances within the timeout. This result is of particular importance because, in an online scheduling scenario, the operator requesting the reschedule can not afford to wait for a long time before the final result. Thus, having an optimal solution in less than 20 seconds is crucial. A closer examination of the instances reveals that those involving only one type of issue–either emergencies or delays–can be solved in less than 10 seconds. When increasing the complexity of the problem, mixing the different kinds of issues, and considering the low and medium scenarios, all the instances still reach an optimal solution in less than 5 seconds. Only in the scenario with a high number of registrations and an almost full schedule could the rescheduling process take more than 10 seconds to find the optimal solution. Notably, this longer runtime occurs just in the instance with

Table 3

Analysis of three different scenarios in terms of low, medium, and high number of registrations in input. Each scenario comprises the analysis in terms of the sum of time slots required to suddenly assign the new registrations with emergency (Emergency Wait) and the difference between the scheduling and rescheduling (Change) of the patients without delay in each instance. The instances are characterized by the number of new registrations (NR) and the number of registrations with delay (D).

Instance	Low		Medium		High	
	EMERGENCY WAIT	CHANGE	EMERGENCY WAIT	CHANGE	EMERGENCY WAIT	CHANGE
1 NR						
2 NR						
3 NR						
$1 NR + 1D$	0					
$2 NR + 2D$						
$3 NR + 3D$						

3 new registrations and 3 patients with delays. This instance is particularly complex since it means that approximately 20% of the patients are either new or have encountered an issue.

Solutions quality. To assess the quality of the rescheduling, we focus on two key criteria: the number of time slots a new registration must wait before starting the treatment (*emergency wait*) and the difference in starting times between the original schedule and the rescheduled one (*change*). These criteria are detailed in Table [3,](#page-9-0) which presents the results for all instances involving at least one new registration. The table highlights the total difference in time slots between the required and assigned starting time slots for new registrations, as well as the difference between the scheduling and rescheduling for patients without delays. In a simpler initial scenario, where a low number of patients are already scheduled, the results show that it is possible to schedule the new registrations as requested without unnecessarily modifying the original schedule. Notably, similar results can be achieved even when the original schedule has a higher number of patients. Indeed, in the medium scenario, all the solutions assign the new registrations at the required time slot and only in the most complex instance the obtained reschedule differs from the original schedule by 12 time slots. In the high scenario, finding a solution that perfectly aligns with the original schedule becomes more challenging. Specifically, for new registrations, the rescheduled solution fails to assign the required time slot in 2 out of the 6 tested instances. In these instances, the new registrations have to wait, in total, 5 time slots, thus, with an average waiting time per patient of 12.5 and 8.3 minutes, respectively. However, in instances where optimal solutions include patients with delays, new registrations are assigned to the required time slots. This derives from the randomness of the inputs. In these instances, even if there are more new registrations, the rescheduler can assign them to the required time slots since these new random patients are required to be assigned to a later phase, reducing the total required occupations of the resources. Even in the high scenario, the approach is still able to reach an optimal solution that is very similar to the original schedule. Indeed, in the instances without any delays, the new solution differs by at most 2 time slots from the original one. Only in the most complex instance, comprising 3 new registrations and 3 patients with delays, the new solution differs by more than 10 time slots. This is due to the high number of registrations already assigned in the original schedule and the fact that all the new registrations are assigned to the required time slot, meaning that the registrations previously assigned in that time slots are forced to be moved. Finally, it is interesting to analyze the usage of overtime. Overtime refers to scheduling beyond the regular time slots to fit in additional or rescheduled patients, which can affect efficiency and resource allocation. Across all scenarios and instances, the majority of the instances are solved without using overtime, demonstrating the effectiveness of the rescheduling approach. However, in the most complex cases, there may be a limited use of overtime to achieve an optimal solution. Even in these cases, the amount of overtime needed never exceeds 6 time slots. This indicates that the rescheduling process remains efficient, reducing both changes to the original schedule and the need for overtime.

6. Related Work

In this section we first analyse papers that consider approaches related to the NMR problem, then we mention works in which ASP has been employed for solving rescheduling problems.

In [\[19\]](#page-12-3) the authors report their experience with rescheduling nonurgent imaging and procedures during the pandemic. To this aim, the authors conducted daily virtual huddles with discussions of rescheduling strategies and issue tracking, reviewing all cases using radiologists, schedulers, residents, and administrative leadership. In [\[20\]](#page-12-4) the authors conducted a cross-sectional study on data obtained via magnetic resonance imaging schedule reviews and self-administrated questionnaires to estimate the rate of "No-Shows" or "Reschedule" magnetic resonance imaging appointments and investigate the correlating factors. Even if in these studies emerged the need for a rescheduling solution to overcome emergencies and issues in the planned schedule, and in some works such as [\[21,](#page-12-5) [22,](#page-12-6) [23,](#page-12-7) [24,](#page-12-8) [25\]](#page-12-9) the scheduling problem is studied and solved, from our understanding no works proposed and evaluated a solution to the rescheduling.

As we mentioned in the introduction, ASP has been successfully used for solving hard combinatorial and application scheduling problems in several research areas, including the Healthcare domain (see, e.g., [\[26\]](#page-12-10) for a survey). Focusing on the problems for which a rescheduling solution has been devised, the first solved problem was the *Nurse Scheduling Problem* [\[27,](#page-12-11) [28\]](#page-12-12), where the goal is to create a scheduling for nurses working in hospital units: The rescheduling problem [\[29\]](#page-12-13) deals with the sudden absences of some nurses. Then, the problem of assigning operating rooms to patients, denoted as *Operating Room Scheduling*, has been treated [\[30\]](#page-12-14), further extended to include bed management [\[31\]](#page-12-15) and then, given the availability of datasets (e.g.,[\[32\]](#page-12-16)) tested on real data [\[33\]](#page-12-17): The rescheduling problem here [\[34\]](#page-13-0) considers cases in which some patients could not be operated in their assigned slot. More recent problems include the *Chemotherepy Treatment Scheduling* problem [\[3\]](#page-11-2), in which patients are assigned a chair or a bed for their treatments, the *Rehabilitation Scheduling Problem* [\[2\]](#page-11-1), which assigns patients to operators in rehabilitation sessions, and the *Master Scheduling Problem* [\[35\]](#page-13-1), which consists of scheduling different specialties to the operating rooms. The rescheduling solution for the first considers the case of patients unavailability, the second deals with the unavailability of operators and/or the absence of patients [\[36\]](#page-13-2), while the last [\[37\]](#page-13-3) considers the unavailability of operating rooms or limitations/changes related to specialties.

7. Conclusion

In this paper we have presented a solution based on Answer Set Programming for the Nuclear Medicine Rescheduling problem as defined in the paper. Out of various scenarios that can motivate the need to reschedule an already computed schedule, we have presented and evaluated a solution to the online reallocation of resources due to emergencies that demand immediate access to equipment or require more time on the resources. Experiments employing real data from a medium size hospital in Italy have shown that our rescheduling solution provides satisfying results even when the concurrent number of emergencies and unavailability is significant.

As future work, we plan to provide rescheduling solutions also to the other two scenarios mentioned in Section [4,](#page-5-0) and to provide a practical web application to support the easy usage of such solutions.

Acknowledgments

This research has been partially supported by Italian Ministry of Research (MUR) (*i*) under PNRR project FAIR "Future AI Research", CUP H23C22000860006, (*ii*) with the European Union - NextGenerationEU, National Recovery and Resilience Plan (NRRP), Mission 4, Component 2, Investment 1.5, project "RAISE - Robotics and AI for Socio-economic Empowerment" (ECS00000035) under the project "Gestione e Ottimizzazione di Risorse Ospedaliere attraverso Analisi Dati, Logic Programming e Digital Twin (GOLD)", CUP H53C24000400006, and (*iii*) under PRIN project PINPOINT, CUP H23C22000280006;

and by Italian Ministry of Enterprises and Made in Italy (MIMIT) under project EI-TWIN "Concezione, progettazione e sviluppo di soluzioni di Energy digital TWIN" Prog n. F/310168/05/X56 - CUP: B29J24000680005 - COR: 22338845. The research of Giuseppe Galatà is partially funded by the "POR FESR Liguria 2014-2020" public grant scheme. Carmine Dodaro is member of Gruppo Nazionale Calcolo Scientifico (GNCS) of INdAM.

References

- [1] C. Dodaro, G. Galatà, C. Marte, M. Maratea, M. Mochi, Nuclear medicine scheduling via answer set programming, in: E. D. Angelis, M. Proietti (Eds.), Proceedings of the 39th Italian Conference on Computational Logic (CILC 2024), volume 3733 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2024.
- [2] M. Cardellini, P. D. Nardi, C. Dodaro, G. Galatà, A. Giardini, M. Maratea, I. Porro, A twophase ASP encoding for solving rehabilitation scheduling, in: S. Moschoyiannis, R. Peñaloza, J. Vanthienen, A. Soylu, D. Roman (Eds.), Proceedings of the 5th International Joint Conference on Rules and Reasoning (RuleML+RR 2021), volume 12851 of *LNCS*, Springer, 2021, pp. 111–125.
- [3] C. Dodaro, G. Galatà, A. Grioni, M. Maratea, M. Mochi, I. Porro, An ASP-based solution to the chemotherapy treatment scheduling problem, Theory and Practice of Logic Programming 21 (2021) 835–851.
- [4] E. Erdem, M. Gelfond, N. Leone, Applications of answer set programming, AI Magazine 37 (2016) 53–68.
- [5] P. Cappanera, M. Gavanelli, M. Nonato, M. Roma, Logic-based Benders decomposition in answer set programming for chronic outpatients scheduling, Theory and Practice of Logic Programming 23 (2023) 848–864.
- [6] M. Gebser, P. Obermeier, T. Schaub, M. Ratsch-Heitmann, M. Runge, Routing driverless transport vehicles in car assembly with answer set programming, Theory and Practice of Logic Programming 18 (2018) 520–534.
- [7] A. A. Falkner, G. Friedrich, K. Schekotihin, R. Taupe, E. C. Teppan, Industrial applications of answer set programming, Künstliche Intelligenz 32 (2018) 165–176.
- [8] 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.
- [9] G. Brewka, T. Eiter, M. Truszczynski, Answer set programming at a glance, Communications of the ACM 54 (2011) 92–103.
- [10] 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.
- [11] M. Gebser, N. Leone, M. Maratea, S. Perri, F. Ricca, T. Schaub, Evaluation techniques and systems for answer set programming: a survey, in: J. Lang (Ed.), IJCAI, ijcai.org, 2018, pp. 5450–5456.
- [12] 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.
- [13] M. Alviano, C. Dodaro, Unsatisfiable core analysis and aggregates for optimum stable model search, Fundamenta Informaticae 176 (2020) 271–297.
- [14] 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.
- [15] P. Bruno, F. Calimeri, C. Marte, M. Manna, Combining deep learning and ASP-based models for the semantic segmentation of medical images, in: S. Moschoyiannis, R. Peñaloza, J. Vanthienen,

A. Soylu, D. Roman (Eds.), Proceedings of the 5th International Joint Conference on Rules and Reasoning (RuleML+RR 2021), volume 12851 of *LNCS*, Springer, 2021, pp. 95–110.

- [16] P. Bruno, F. Calimeri, C. Marte, Dedudeep: An extensible framework for combining deep learning and asp-based models, in: G. Gottlob, D. Inclezan, M. Maratea (Eds.), Proceedings of the 16th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR 2022), volume 13416 of *LNCS*, Springer, 2022, pp. 505–510.
- [17] F. Buccafurri, N. Leone, P. Rullo, Enhancing Disjunctive Datalog by Constraints, IEEE Transactions on Knowledge and Data Engineering 12 (2000) 845–860.
- [18] W. Faber, G. Pfeifer, N. Leone, Semantics and complexity of recursive aggregates in answer set programming, Artificial Intelligence 175 (2011) 278–298.
- [19] A. Vagal, M. Mahoney, B. Allen, S. Kapur, G. Udstuen, L. Wang, S. Braley, A. Makramalla, S. Chadalavada, K. A. Choe, et al., Rescheduling nonurgent care in radiology: implementation during the coronavirus disease 2019 (covid-19) pandemic, Journal of the American College of Radiology 17 (2020) 882–889.
- [20] M. O AlRowaili, A. E. Ahmed, H. A. Areabi, Factors associated with no-shows and rescheduling mri appointments, BMC Health services research 16 (2016) 1–7.
- [21] W. R. Reinus, A. Enyan, P. Flanagan, B. Pim, D. S. Sallee, J. Segrist, A proposed scheduling model to improve use of computed tomography facilities, Journal of Medical Systems 24 (2000) 61–76.
- [22] S. E. Seltzer, J. T. Rhea, J. H. Thrall, S. Saini, J. Sumner, Improving the efficiency and service of computed tomographic scanning, Academic Radiology 1 (1994) 164–170.
- [23] E. Pérez, L. Ntaimo, W. E. Wilhelm, C. Bailey, P. McCormack, Patient and resource scheduling of multi-step medical procedures in nuclear medicine, IIE Transactions on Healthcare Systems Engineering 1 (2011) 168–184.
- [24] Q. Xiao, L. Luo, S. Zhao, X. bin Ran, Y. bing Feng, Online appointment scheduling for a nuclear medicine department in a chinese hospital, Computational and Mathematical Methods in Medicine 2018 (2018). URL: [https://api.semanticscholar.org/CorpusID:5061754.](https://api.semanticscholar.org/CorpusID:5061754)
- [25] F. Akhavizadegan, J. Ansarifar, F. Jolai, A novel approach to determine a tactical and operational decision for dynamic appointment scheduling at nuclear medical center, Computers & Operations Research 78 (2017) 267–277.
- [26] M. Alviano, R. Bertolucci, M. Cardellini, C. Dodaro, G. Galatà, M. K. Khan, M. Maratea, M. Mochi, V. Morozan, I. Porro, M. Schouten, Answer set programming in healthcare: Extended overview, in: IPS and RCRA 2020, volume 2745 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2020.
- [27] C. Dodaro, M. Maratea, Nurse scheduling via answer set programming, in: LPNMR, volume 10377 of *LNCS*, Springer, 2017, pp. 301–307.
- [28] 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.
- [29] M. Alviano, C. Dodaro, M. Maratea, Nurse (re)scheduling via answer set programming, Intelligenza Artificiale 12 (2018) 109–124.
- [30] 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.
- [31] C. Dodaro, G. Galatà, M. K. Khan, M. Maratea, I. Porro, An ASP-based solution for operating room scheduling with beds management, in: P. Fodor, M. Montali, D. Calvanese, D. Roman (Eds.), Proceedings of the Third International Joint Conference on Rules and Reasoning (RuleML+RR 2019), volume 11784 of *Lecture Notes in Computer Science*, Springer, 2019, pp. 67–81.
- [32] L. Maier-Hein, M. Wagner, T. Roß, A. Reinke, S. Bodenstedt, P. M. Full, H. Hempe, D. Mîndroc-Filimon, P. Scholz, T. N. Tran, P. Bruno, A. Kisilenko, B. Müller, T. Davitashvili, M. Capek, M. Tizabi, M. Eisenmann, T. J. Adler, J. Gröhl, M. Schellenberg, S. Seidlitz, T. Y. E. Lai, V. Roethlingshoefer, F. Both, S. Bittel, M. Mengler, M. Apitz, S. Speidel, H. G. Kenngott, B. P. Müller-Stich, Heidelberg colorectal data set for surgical data science in the sensor operating room, CoRR abs/2005.03501 (2020). URL: [https://arxiv.org/abs/2005.03501.](https://arxiv.org/abs/2005.03501) [arXiv:2005.03501](http://arxiv.org/abs/2005.03501).
- [33] C. Dodaro, G. Galatà, M. Gebser, M. Maratea, C. Marte, M. Mochi, M. Scanu, Operating room scheduling via answer set programming: Improved encoding and test on real data, Journal of Logic

and Computation. To appear (2024).

- [34] C. Dodaro, G. Galatà, M. Maratea, I. Porro, An ASP-based framework for operating room scheduling, Intelligenza Artificiale 13 (2019) 63–77.
- [35] M. Mochi, G. Galatà, M. Maratea, Master surgical scheduling via answer set programming, Journal of Logic and Computation 33 (2023) 1777–1803.
- [36] M. Cardellini, C. Dodaro, G. Galatà, A. Giardini, M. Maratea, N. Nisopoli, I. Porro, Rescheduling rehabilitation sessions with answer set programming, Journal of Logic and Computation 33 (2023) 837–863.
- [37] G. Galatà, M. Maratea, M. Mochi, C. Marte, Rescheduling master surgical schedules via answer set programming, Progress in Artificial Intelligence. To appear (2024).