A Two-Phase ASP Encoding for Solving Rehabilitation Scheduling

A Two-Phase ASP Encoding for Solving Rehabilitation Scheduling *

Matteo Cardellini[1,4][0000−0003−3788−9475], Paolo De Nardi[2], Carmine Dodaro[3][0000−0002−5617−5286], Giuseppe Galatà[1][0000−0002−1948−4469], Anna Giardini[2], Marco Maratea[4][0000−0002−9034−2527]**, and Ivan Porro[1][0000−0002−0601−8071]

[1] SurgiQ srl, Italy; {name.surname}@surgiq.com
[2] ICS Maugeri, Italy; {name.surname}@icsmaugeri.it
[3] DeMaCS, University of Calabria, Rende, Italy; dodaro@mat.unical.it
[4] DIBRIS, University of Genova, Genova, Italy; marco.maratea@unige.it

Abstract. The rehabilitation scheduling process consists of planning rehabilitation physiotherapy sessions for patients, by assigning proper operators to them in a certain time slot of a given day, taking into account several requirements and optimizations, e.g., patient's preferences and operator's work balancing. Being able to efficiently solve such problem is of upmost importance, in particular after the COVID-19 pandemic that significantly increased rehabilitation's needs.
In this paper, we present a solution to rehabilitation scheduling based on Answer Set Programming (ASP), which proved to be an effective tool for solving practical scheduling problems. Results of experiments performed on both synthetic and real benchmarks, the latter provided by ICS Maugeri, show the effectiveness of our solution.

1 Introduction

The rehabilitation scheduling process [17–19, 24] (RSP) consists of planning patients' physiotherapy sessions inside a rehabilitation institute. Hospitals that may profitably make a practical use of such scheduling, including those managed by ICS Maugeri[5] that will provide benchmarks in this paper, deal with up to hundreds of patients with a team of just few tens of physiotherapists; so, it is of paramount importance to be able to assign patients to operators efficiently. A recent article [9] found that 2.41 billion people could benefit from rehabilitation services. This finding means that almost one third of the people in the world needs rehabilitation at some point during the course of their disease or injury; further, this number is predicted to trend upward given the current demographic and health shifts. In addition, there is emerging evidence that many of the people affected by the COVID-19 pandemic have long-term consequences regardless

* The paper will appear in the LNCS proceedings of the 5th International Joint Conference on Rules and Reasoning (RuleML+RR 2021).
** Corresponding author.
[5] https://www.icsmaugeri.it/.

of the disease severity or length of hospitalisation, thus further increasing the demand for rehabilitation services globally.

The RSP is subject to several constraints, i.e., legal, medical and ethical, that need to be taken into consideration in order to find a viable schedule. For example, the main constraints that have to be dealt with are the maximum capacity of rehabilitation gyms, the legal working time and rest periods for operators, and the minimum durations of physiotherapy sessions. Moreover, several preferences shall be considered, e.g., due to clinical and organizational reasons it is often best for the patient to be treated as often as possible by the same operator and defined slots for the rehabilitation sessions are to be preferred; also rehabilitation professionals' work balancing needs to be taken into proper account.

In this paper, we present a solution to the RSP based on Answer Set Programming (ASP) [16, 20, 6, 7], which proved to be an effective tool for solving practical scheduling problems [15, 22, 10, 4], thanks to efficient solvers (such as CLINGO [13] and WASP[1]; see, e.g., [14] for an overview). The solution is designed as a two-phase encoding (Section 3): The first phase, called *board*, deals with the problem of assigning a physiotherapist to every patient considering the total working time of the physiotherapist and the minimum mandatory time of rehabilitation sessions. In the second phase, called *agenda*, a start and end time of every rehabilitation session is defined given the assignments between patients and physiotherapists found in the first phase. Our two-phase solution is not guaranteed to find the best possible overall solution, but has been designed in this way because: (*i*) it simplifies the overall encoding and its practical use, and (*ii*) it mimics how schedules have been computed so far (in a non-automatic way) in ICS Maugeri and gives freedom to physiotherapists' coordinators to perform any desired manual change to the board, before planning the agenda. We first tested our encoding (Section 4) on real benchmarks from ICS Maugeri related to the daily scheduling of neurological patients from two of their rehabilitation institutes in the North of Italy, namely Genova Nervi and Castel Goffredo: Results using the ASP solver CLINGO [13], focused on understanding the percentage of the real benchmarks in which no solution can be found in very short time, i.e., much shorter than in production, show that this happens approximately for only less than one third of the instances. Then, given that ICS Maugeri is planning to instrument with automated techniques other, possibly larger, institutes in addition to Genova Nervi and Castel Goffredo, we generated a wide set of synthetic benchmarks, whose parameters are inspired by the real data. We made a wide experimental evaluation, and statistically confronted these results with those with real data using classification decision tree methods [21], with the aim of predicting the behavior of our solution on such larger institutes. Results show that the accuracy is high, so our synthetic benchmarks look significant to indicate a possible behavior on real data coming from other institutes with other parameters. Moreover, our analysis also outlined what are the features of the problems that affect the results mostly. The paper is then completed by problem description in Section 2, and related work discussion and conclusions in Section 5.

2 Problem description

The delivery of rehabilitation services is a complex task that involves many healthcare professions such as physicians, physiotherapists, speech therapists, psychologists and so on. In particular, physiotherapists spend most of their time with the patients and their sessions constitute the core of the daily agenda of the patient, around which all other commitments revolve. For this reason, this article is focused on scheduling the physiotherapy sessions in the most efficient way, optimising the overall time spent with the patient.

The agenda for the physiotherapy sessions is computed by the coordinator of the physiotherapists. This process is repeated on a daily basis in order to take into account any change in the number and type of patients to be treated, the number of operators available, and, until recently, it has been performed manually, without any automation. In the following, the main elements and constraints of the problems are described.

The usual scheduling practice, entails two subsequent phases resulting in the computation of a board and an agenda, that we herewith describe. In short, the first phase, called board, deals with the problem of assigning a physiotherapist to every patient, keeping track of the total working time of the operator and the minimum mandatory time of rehabilitation sessions. In the second, consequential phase, called agenda, a start and end time of every rehabilitation session is searched given the assignments between patients and operators found in the first phase.

Going more in details, in the board phase, the working hours of operators are simplified by counting their total working time, in minutes, and assigning patients to each operator in order to keep the cumulative time of all the sessions in which the operators are involved underneath their total working time. In this phase, patient-operator assignment preferences, expressed by the coordinator before the start of the scheduling procedure, are taken into account and respected as far as possible. In the agenda phase, given an assignment found by the board, every patient-operator session is assigned a starting and ending time, respecting the more granular working hours of the operators and the times in which the patients are unavailable. At this stage, the location in which the rehabilitation session is performed, is also considered: a gym is assigned to every session, keeping into consideration the maximum number of simultaneous sessions allowed inside the gym. The choice of the gym has also to be made between a subset of gyms that are located at the same floor of the room of the patient in order to avoid elevators and stairs that can result in discomfort to patients and can quickly congest the hospital. In this phase, time preferences for each patient are also considered: in fact, plans in which the sessions are performed nearer the desired time of the patients are preferred to the others.

In the next paragraphs, we describe more in details the main elements of our encoding, namely patients, operators and sessions, as well as the constraints and preferences entailed by the board and agenda phases.

Patients. Patients are characterized by their:

- type (Neurological, Orthopaedic, Alcoholic, COVID-19 Positive, COVID-19 Negative, Outpatient),
- aid needs, i.e., if they need specific care or not,
- payment status (full payer or in charge of the National Healthcare Service),
- forbidden times, i.e., the time intervals when the patient cannot be scheduled,
- ideal time, i.e., the preferred scheduled time expressed by the coordinator,
- preferred operators, i.e., the list of physiotherapists, ordered by priority, the patient can be assigned to,
- overall minimum length, i.e., the minimum amount of care time that the patient is guaranteed to be scheduled,
- sessions, i.e., the list of sessions to be scheduled.

Operators. Physiotherapists, that will be called operator from now on, are characterized by their:

- qualifications, i.e., patient's types which the operator can treat,
- operating times, i.e., the part of the operator's working times dedicated to the direct care of the patients. The operating times are usually split in morning and afternoon shifts.

Sessions. The coordinator, in accordance with the rehabilitation program set by the physician, determines the daily activities of the patient. These activities can be performed in one or two therapy sessions, in the latter case one session will be scheduled in the morning and the other in the afternoon shift.

Each session can be delivered to patients either by individualized ("one-on-one" sessions) or supervised (one therapist supervising more patients at the same time, each patient carrying out their personal activity independently). It must be noted that, while operators are delivering one-on-one therapy to patients, they can supervise other patients but cannot deliver one-to-one therapy to another patient. When the operators are particularly overbooked, their one-to-one sessions can be partially converted to supervised ones. These mixed sessions can either start with a supervised part and then continue with the one-on-one part, or vice-versa, or even start and end with a supervised part with a middle one-on-one session. Obviously, an operator can supervise different patients only if their sessions are located at the same place. The characteristics are:

- delivery mode (one-on-one, supervised),
- minimum one-on-one length, i.e., the minimum length of the session guaranteed to be delivered one-on-one,
- ideal overall length, i.e., the overall length of the session including the one-on-one and supervised parts,
- optional status, i.e., if the session can be left out of the schedule in case of overbooked operators,
- forced time, i.e., the time when the session must be scheduled; if empty, the session is placed as close as possible to the patient's preferred time,
- location, i.e. the place where the session must be delivered.

Board. In the board phase all patients are assigned to an available operator, according to the following criteria:

- compatibility between patient and operator, depending on the patient's type and operator qualifications, the patient's forced time, if any, and the operator working times, by also checking if the operator has enough time to provide the guaranteed overall minimum length and minimum one-to-one length to each patient and session,
- forced assignments of a patient to an operator: In special cases, the coordinator can override the preferred operators list and force an assignment regardless of all other considerations,
- the patients should be fairly distributed among all available operators, taking into account their type, aid needs and payment status,
- the patients should be assigned to the operators respecting as much as possible their preferred operators list, which considers primarily the choices of the coordinator and secondarily the history of the past assignments.

Agenda. The results of the board phase can be revised and, if necessary, manually modified by the coordinator. Once the coordinator is satisfied with the board, it is possible to proceed to the agenda scheduling, using the approved board as input data. The criteria for the agenda phase are:

- compliance with the forced time of the session, if specified,
- two sessions of the same patient must be assigned in different shifts,
- compliance with the minimum one-on-one length of the session,
- no overlap between two one-on-one sessions (or their one-on-one sections if the sessions are mixed) assigned to the same operator,
- observance of the maximum capacity of the locations (1 for each room, varying for the gyms),
- respect of the overall minimum length of the patient,
- respect of the one-on-one minimum session length,
- compliance with the forbidden times of the patient,
- sessions can only be scheduled within the working times of the operator,
- the start time of each session should be as close as possible to the preferred time, either specified by the coordinator or inferred from previous schedules,
- for mixed sessions, the one-on-one part should be maximized,
- the largest possible number of optional sessions should be included,
- the overall length, including the one-on-one and supervised parts in case of mixed sessions, should be as close as possible to the ideal overall length specified by the coordinator.

3 A Two-Phase ASP Encoding

In the following, 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 [11]. For details about syntax and semantics of ASP programs, we refer the reader to [8].

3.1 Board encoding

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

- Instances of `patient(P)`, `operators(O)`, and `type(T)` represent the identifiers of patients, operators, and the different types of patients that can be visited, respectively, where `P` and `O` are numbers, whereas `T` can be: *neurologic*, *neurologic-lifter*, *orthopaedic*, *orthopaedic-lifter*, *covid-19-positive*, *covid-19-negative*, or *outpatient*. Moreover, a fictitious operator with `ID` equals to -1 is included in the list of all the operators, and it is needed to intercept all patients that cannot be assigned to other operators.
- Instances of `operator_contract(ID,TIME,MAX)` represent the contract of the operator with the identifier `ID`, and include the quantity of time (in time units) the operator works in a day (`TIME`), and the maximum number of patients the operator can visit during the day (`MAX`).
- Instances of `operator_limit(ID,T,VALUE)` represent the maximum number of patients (`VALUE`) of type `T` the operator with identifier `ID` can visit. The operator with `ID` equals to -1 has no patients limits.
- Instances of `patient_data(ID,T,MIN)` represent the data associated to the patient with the identifier `ID`, and include the type of the patient (`T`), and the minimum cumulative time of all sessions of the patient during the day (`MIN`).
- Instances of `patient_session(ID,MIN,LOC)` represent a rehabilitation session that the patient with identifier `ID` needs to perform during the day. The session is characterized by a minimum length for the session in time units (`MIN`), and the location of the session (`LOC`).
- Instances of `patient_preference(ID,OP,W)` represent the preference of the patient with identifier `ID` to be treated by the operator with identifier `OP`, where `W` specifies the weight of the preference.
- Similarly, instances of `history_preference(ID,OP,W)` represent the preference of the patient based on the history of previous sessions.

The output is an assignment represented by atoms of the form `assignment(OP, PAT)` stating that the patient `PAT` will be treated by the operator `OP`.

Encoding. The related encoding is shown in Figure 1, and is described in the following. To simplify the description, the rule appearing at line i in Figure 1 is denoted with r_i. Rule r_1 ensures that each patient is assigned to exactly one operator. Rules r_2 and r_3 are used to define if the session between a patient and an operator will be performed individually in a single location (r_2), or it will be executed in the same location of another session (r_3). Rule r_4 ensures that the time required by the patients assigned to an operator does not exceed the maximum time of her/his contract. Rule r_5 ensures that each operator does not exceed the maximum number of patients to visit during the day. Rule r_6 is similar to the previous one, but in this case the limits are imposed according to the type of the patient.

```
1 {assignment(OP, PAT) : operator(OP)} = 1 :- patient(PAT).
2 uniqueLocationLength(OP,PAT,DUR) :- assignment(OP,PAT), patient_session(PAT,_,LOC),
      patient_data(PAT,_,DUR), #count{ID:patient_session(ID,_,LOC), assignment(OP,ID)} < 2.
3 sameLocationLength(OP,PAT,DUR) :- assignment(OP,PAT), patient_session(PAT,DUR,LOC),
      #count{ID:patient_session(ID,_,LOC), assignment(OP,ID)} > 1.
4 :- operator_contract(OP,TIME,_), #sum{U,PAT:uniqueLocationLength(OP,PAT,U); S,
      PAT:sameLocationLength(OP,PAT,S)} > TIME.
5 :- operator_contract(OP,_,N), #count{PAT:assignment(OP,PAT)} > N.
6 :- operator_limit(OP,T,N), #count{PAT:assignment(OP,PAT), patient_data(PAT,T,_} > N.
7 :~ #sum{W, PAT:assignment(OP,PAT), patient_preference(PAT,OP,W)} = N. [N@3]
8 :~ #count{PAT: assignment(-1, PAT)} = N. [N@2]
9 :~ #sum{W, PAT:assignment(OP,PAT), history_preference(PAT,OP,W)} = N. [N@1]
```

Fig. 1: ASP Encoding for the allocation problem.

Weak constraints from r_7 to r_9 are then used to provide preferences among different assignments. In particular, r_7 is used to maximize the assignments that fulfil the preferences of each patient. Then, r_8 is used to minimize the number of patients that are assigned to the fictitious operator. Finally, r_9 is used to maximize the solutions that preserve assignments dictated by the history of previous sessions.

3.2 Agenda encoding

Data Model. The following atoms constitute the input data:

- Instances of `patient(ID,MIN)` represent a patient identified by `ID`, and a minimum rehabilitation session of `MIN` length in time units that the patient has to undertake during the day.
- Instances of `period(PER,OP,STA,END)` define the start (`STA`) and end (`END`) time in the period `PER` (which can be *morning* or *afternoon*), which corresponds to the shift, of the operator with identifier `OP`.
- Instances of `time(PER,OP,T)` define the time slots (`T`) during the period `PER` where the operator `OP` works. In particular, `T` ranges from `STA` to `END` defined for instances of `period(PER,OP,STA,END)`.
- Instances of `location(ID,CAP,PER,STA,END)` represent a location, with an identifier `ID`, a maximum capacity of `CAP`, and during the period `PER` is open from the time unit `STA` until `END`.
- Instances of `macro_location(MLOC,LOC)` define that the location `LOC` is inside the macro-location `MLOC`.
- Instances of `session(ID,PAT,OP)` represent a session between the patient `PAT` and the operator `OP`, coming from the `assignment(OP,PAT)` output of the board phase to which a unique `ID` is added (to discriminate between *morning* and *afternoon* shifts).
- Instances of `session_type(ID,OP,TYPE)` represent that the session with identifier `ID` assigned to operator `OP` is of type `TYPE` (which can be *individual* or *supervised*).

- Instances of `session_macro_location(ID,MLOC)` represent that the session with identifier `ID` has to be held in the macro-location `MLOC`.
- Instances of `session_length(ID,MIN,IDEAL)` represent that the session `ID` has a minimum length (`MIN`) that has to be performed in individual, and an ideal length (`IDEAL`) that would be beneficial to the patient, but it is not mandatory to perform.
- Instances of `mandatory_session(ID)` and `optional_session(ID)` identify sessions that are mandatory and optional, respectively.
- Instances of `forbidden(PAT,PER,STA,END)` represent an unavailability of the patient `PAT` in the period `PER` from the time unit `STA` to `END`.
- Instances of `session_preference(ID,PER,START,TYPE)` represent the preference of the patient, stating that session should be held during the period `PER` and it must start at the time unit `START`, where `TYPE` indicates if the preference is *high* or *low*.

The output is represented by atoms `start(ID,PER,T)`, `length(ID,PER,L)`, and `session_location(SES,LOC)`, which indicate the start, length and location of each session, respectively.

Encoding. In Figure 2 the encoding for the agenda is presented.

Rules r_1 and r_2 assign a start time to every session; for the optional session, the start atom can be unassigned. Rule r_3 defines a length for all the sessions: the session length cannot be lower than the minimum time of the session and cannot be greater than the ideal time the session should take. Rule r_4 assigns a location for each session. Rules r_5 and r_6 reserve to each session slots of time before it starts and after it ends, in which the session can be performed in a supervised fashion. These extensions cannot be longer than the difference between the maximum and the minimum length of the session.

Then, rules r_7 and r_8 define auxiliary atoms `ext_start` and `ext_length` using the slots of times reserved for the extensions. Rule r_9 defines an auxiliary atom of the form `individual_session_location(ID,LOC,OP,MIN,IDEAL)` which represents that an individual session `ID` is in the location `LOC`, is assigned to the operator `OP`, and its minimum and ideal lengths are equal to `MIN` and `IDEAL`, respectively. Rule r_{10} defines `session_time(ID,OP,PL,PER,T)` which states that during time `T` of period `PER` the session `ID` is being performed by operator `OP`.

Rule r_{11} states that two individual assignments shall not overlap. Rule r_{12} imposes that each patient is assigned to at most one session per period. Rules r_{13} trough r_{15} impose that the optional individual time (i.e., the difference between the minimum length of the session and the planned length) is added fairly to all individual sessions, starting with shorter ones. Rule r_{16} imposes that for each time slot, the operator is not in two different places. Rule r_{17} states that patients must have their minimum time reserved. Rule r_{18} imposes a limit on the concurrent use of locations with limited capacity. Rules r_{19} trough r_{21} impose that a session cannot happen during a forbidden time. Rule r_{22} avoids that, during a time slot, the distribution of sessions between each pair of locations inside the same macro location is unfair (i.e., a location is at its full capacity while another is empty).

```
{start(ID,PER,TS) : time(PER,OP,TS)} = 1 :- session(ID,_,OP), mandatory_session(ID).
{start(ID,PER,TS) : time(PER,OP,TS)} <= 1 :- session(ID,_,OP), optional_session(ID).
{length(ID,PER,NL) : time(PER,OP,L), NL=L-ST, TS+NL <= END, NL>= MIN, NL<= IDEAL} = 1 :-
    start(ID,PER,TS), period(PER,OP,ST,END), session(ID,_,OP),
    session_length(ID,MIN,IDEAL).
{session_location(ID,LOC): macro_location(MAC,LOC)} = 1 :- session_macro_location(ID,MAC).
{before(ID,NL): time(PER,OP,L), NL=L-ST, NL<=TS-ST} = 1 :- start(ID,PER,TS),
    period(PER,OP,ST,_), session(ID,_,OP).
{after(ID,NL): time(PER,OP,L), NL=L-ST, NL<=END-TS-LEN} = 1 :- start(ID,PER,TS),
    period(PER,OP,ST,END), length(ID,PER,LEN), session(ID,_,OP).
ext_start(ID,PER,TS-LB) :- start(ID,PER,TS), before(ID,LB).
ext_length(ID,PER,L+LA+LB) :- length(ID,PER,L), after(ID,LA), before(ID,LB).
individual_session_location(ID,LOC,OP,MIN,IDEAL) :- session_type(ID,OP,individual),
    session_location(ID,LOC), session_length(ID,MIN,IDEAL).
session_time(ID,OP,PL,PER,TS..TS+L-1) :- session(ID,_,OP), session_location(ID,PL),
    ext_start(ID,PER,TS), ext_length(ID,PER,L).
:- start(ID,PER,TS), length(ID,PER,L), session_type(ID,OP,individual), start(ID2,PER,TS2),
    session_type(ID2,OP,individual), ID!=ID2, TS2>=TS, TS2<TS+L.
:- session(ID1,PAT,_), session(ID2,PAT,_), start(ID1,PER,_), start(ID2,PER,_), ID1!=ID2.
:- individual_session_location(ID1,LOC,OP,MIN1,OPT1), length(ID1,PER,L1),
    individual_session_location(ID2,LOC,OP,MIN2,OPT2), length(ID2,PER,L2), OPT1-L1 <=
    OPT2-MIN2, OPT2-L2 <= OPT1-MIN1 , |OPT1 -L1 - OPT2 + L2| > 1.
:- individual_session_location(ID1,LOC,OP,MIN1,OPT1), length(ID1,PER,L1),
    individual_session_location(ID2,LOC,OP,MIN2,OPT2), length(ID2,PER,L2), OPT1-L1 >
    OPT2-MIN2, L2 > MIN2.
:- individual_session_location(ID1,LOC,OP,MIN1,OPT1), length(ID1,PER,L1),
    individual_session_location(ID2,LOC,OP,MIN2,OPT2), length(ID2,PER,L2), OPT1-L1 <=
    OPT2-MIN2, OPT2-L2 <= OPT1-MIN1, OPT2 < OPT1, OPT1-L1 < OPT2-L2.
:- session_time(ID,OP,PL,PER,T), session_time(ID2,OP,PL2,PER,T), ID != ID2, PL != PL2.
:- patient(PAT,MIN), #sum{LEN, ID: session(ID,PAT,_), ext_length(ID,_,LEN)} < MIN.
:- location(LOC,LIM,PER,ST,END), LIM>0, time(PER,_,T), T>=ST, T<END, #count{ID:
    session_time(ID,_,LOC,PER,T)} > LIM.
:- forbidden(PAT,PER,ST,_), session(ID,PAT,_), ext_start(ID,PER,TS), ext_length(ID,PER,L),
    ST>=TS, ST<TS+L.
:- forbidden(PAT,PER,_,END), session(ID,PAT,_), ext_start(ID,PER,TS), ext_length(ID,PER,L),
    END>TS, END<=TS+L.
:- forbidden(PAT,PER,ST,END), session(ID,PAT,_), ext_start(ID,PER,TS),
    ext_length(ID,PER,L), ST<=TS,END>TS.
:- time(PER,_,T), macro_location(MAC,LOC1), macro_location(MAC,LOC2),
    #sum{1,ID1:session_time(ID1,_,LOC1,PER,T); -1,ID2:session_time(ID2,_,LOC2,PER,T)} > 2.
:~ length(ID,_, L), session_length(ID,MIN,IDEAL), D=|L-IDEAL|. [D@6, ID]
:~ start(ID,PER,_), session_type(ID,_,individual), session_preference(ID,PER2,_,high),
    D=|PER-PER2|. [D@5, ID]
:~ start(ID,PER,TS), session_type(ID,_,individual), session_preference(ID,PER,TS2,high),
    D=|TS-TS2|. [D@4, ID]
:~ optional_session(ID), time(PER,_,TS), not start(ID,PER,TS). [1@3,ID]
:~ start(ID,PER,_), session_preference(ID,PER2,_,low), session_type(ID,_,individual),
    optional_session(ID), D=|PER-PER2|. [D@2, ID]
:~ start(ID,PER,TS), session_preference(ID,PER,TS2,low), session_type(ID,_,individual),
    optional_session(ID), D=|TS-TS2|. [D@1, ID]
```

Fig. 2: ASP Encoding for the timetable problem.

The weak constraint r_{23} states that each session duration should be as close as possible to the ideal duration. Rules r_{24} and r_{25} minimize the distance between the actual and the preferred starting time for the *high* session priority preferences. Rule r_{26} maximizes the number of optional sessions included in the scheduling. Rules r_{27} and r_{28} are similar to r_{24} and r_{25}, respectively, but for the *low* session priority preferences.

Table 1: Dimensions of the ICS Maugeri's institutes.

Institute	# Operators	# Patients	Density	# Floors	# Gyms
Genova Nervi	[9,18]	[37,67]	[2.4,5.2]	1	1
Castel Goffredo	[11,17]	[51,78]	[3.5, 6.4]	2	3

Table 2: Results on ICS Maugeri institutes.

	Branch & Bound + RoM				Unsatisfiable Core			
	Genova Nervi		Castel Goffredo		Genova Nervi		Castel Goffredo	
	Board	Agenda	Board	Agenda	Board	Agenda	Board	Agenda
% Optimum	35%	0%	0%	0%	22%	45%	0%	0%
% Satisfiable	65%	100%	100%	67%	78%	55%	100%	70%
% Unknown	0%	0%	0%	33%	0%	0%	0%	30%
Avg Time for opt	1.1s	-	-	-	10s	0.01s	-	-
Avg Time Last SM	1.3s	30s	5.2s	30s	12.1s	21.3s	10.4s	30s

4 Experimental Analysis

In this section, analyses performed on the two encodings are presented. The first part of our analysis is performed on real data (that of course can encapsulate also forced assignments and timings, and revisions between phases, if any) coming from the institutes of Genova Nervi and Castel Goffredo; then, in order to evaluate the scalability of the approach and to analyse how our solution would behave in larger institutes, an analysis is performed on synthetic instances with increasing dimensions, but considering real parameters. A comparison between the real and synthetic instances validates the approach and demonstrates that synthetic instances can reasonably model the problem at hand. Encodings and benchmarks used in the experiments can be found at: `http://www.star.dist.unige.it/~marco/RuleMLRR2021/material.zip`.

Real data. ICS Maugeri utilizes, in its daily activity of scheduling the rehabilitation session of its patients, a web-based software called QRehab [23], developed by SurgiQ, which is built on top of the specified encoding; thus, analysis can be performed on real data coming from the institutes of Genova Nervi and Castel Goffredo which tested and used this software since mid 2020 for Genova Nervi and the beginning of 2021 for Castel Goffredo. This allowed us to access 290 instances for Genova Nervi and 100 for Castel Goffredo. Table 1 provides an overview of the dimension of the instances in the two institutes in terms of number of physiotherapists, number of daily patients, density of patients per operator, number of floors (i.e., macro-locations) and number of total gyms (i.e., locations). In Table 2, the results obtained by the two encodings are presented in terms of percentage of instances for which an optimal/satisfiable/no solution is computed. Last two rows report the mean time of instances solved optimally and of the last computed solution for all satisfiable instances, respectively. The scheduling was performed using the ASP solver CLINGO [13] with a cut-off of $30s$

using two different optimization methods: the first is the default Branch&Bound (BB) optimization method [12] with the option `--restart-on-model` enabled; the second instead leverages the Unsatisfiable Core (USC) algorithm [3, 5] with the clingo options `--opt-strategy=usc,k,0,4` and `--opt-usc-shrink=bin` enabled. As it can be seen in Table 2, results are mixed: the USC algorithm performs better in the agenda encoding while BB algorithm is better on the board scheduling: moreover, 100% of the board instances are solved, while for approximately one third of the agenda instances a solution can not be found. Considering these are hard real instances, results are positive and highly appreciated by ICS Maugeri members.

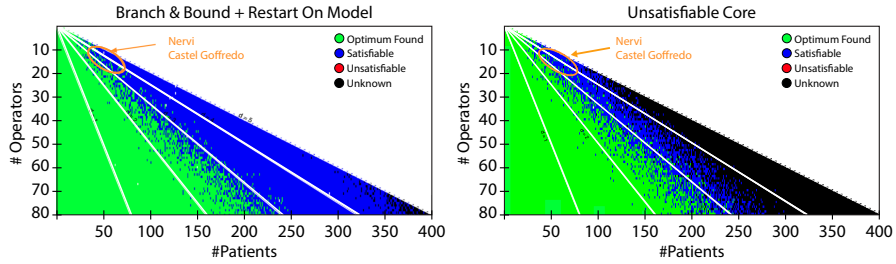

Fig. 3: Results of CLINGO using the BB optimization algorithm (left) and the USC optimization algorithm (right) on synthetic benchmarks of the board.

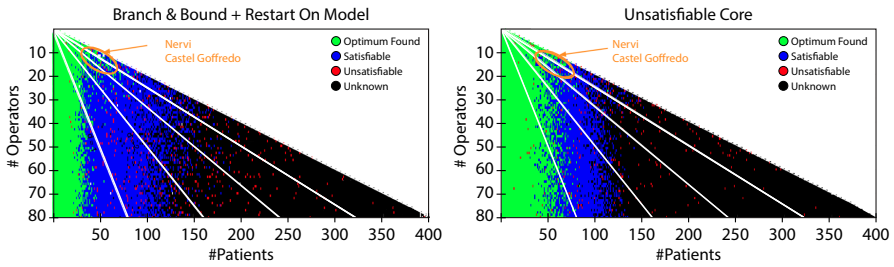

Fig. 4: Results of the synthetic benchmarks of the agenda produced by CLINGO with the BB optimization algorithm and the option `--restart-on-model` enabled (left) and the USC optimization algorithm (right).

Synthetic data. In order to understand how the system scales to larger institutes, that ICS Maugeri plans to instrument soon with such solution, a simulated approach is needed. For this reason, a generator able to produce random

instances with features as close as possible to the ones of real hospitals was developed. Some examples of real data utilized are: the percentage of individual and supervised sessions, the medium length of operator's shifts, the occurrence of forbidden time slots for patients, and the ideal length of sessions. For every new instance created, each feature was extracted from a random distribution which was modelled from the real data coming from the hospitals or from the knowledge of institute administrators and managers. In Figure 3 results of the scheduling of the board encoding, computed from the synthetic data, are presented. The x-axis defines the number of patients and the y-axis the number of operators; white lines represent points in which the density is an integer. Every pixel of the image depicts the mode of the results of 5 simulations executed with the corresponding number of patients and operators with a cut-off of $30s$ using the BB optimization algorithm and with the `--restart-on-model` option enabled (left) and the USC optimization algorithm (right). The colour of a pixel thus signals if the majority of instances with that particular number of operators and patients resulted in: (i) *Optimum Found*, signalling that the optimal stable model was found, (ii) *Satisfiable*, when at least one sub-optimal stable model was found, but the solution is not guaranteed to be optimal, (iii) *Unknown*, if no stable model could be found before the cut-off, (iv) *Unsatisfiable*, when no stable model exists which can satisfy all the constraints. As it can be seen from the figure, the results of the scheduling are directly proportional to the density (i.e., the average number of patients for every operator), changing from *Optimum Found* to *Satisfiable* when reaching a density of approximatively 2.9 patients per operator. Notably, despite the use of random instances, no instance results *Unsatisfiable* since the fictitious operator can always catch the patients which cannot be assigned to any operator (due to all the operators reaching full capacity). The position of the hospitals of Genova Nervi and Castel Goffredo are highlighted with a circle. In this figure it can be noted how the BB gives better results than USC, by being able to find, before the cut-off, at least a sub-optimal stable model for instances of higher densities, while, instead, the USC algorithm returns *Unknown*.

In Figure 4 the results of the agenda encoding, scheduled with the BB optimization algorithm (left) and USC algorithm (right), are presented in the same format as the previous experiment. The instances for this experiment are the same as the previous one, but are augmented with the assignments between patients and operators found by CLINGO with the board encoding and other needed parameters. As previously stated, each pixel represent 5 instances and its color represents the mode of the CLINGO results. Here two things can be noted: (i) unlike the board results, which showed a proportionality with the density, these results show a correlation only with the number of patients, and (ii) some red dots scattered in the image indicate that some instances result *Unsatisfiable*; this can happen since the random data could create some instances with features that cause an impossibility to plan. With the BB optimization algorithm, the transition between the *Optimum Found* results and *Satisfiability* is located near 40 patients and near 120 patients for the transition between *Satisfiability*

and *Unknown*. As it can be seen in Figure 4 (right), the USC algorithm, instead, performs better and moves the transition between the *Optimum found* results and *Satisfiable* from 40 to 60 patients but, on the other hand, the transition between *Satisfiable* and *Unknown* slightly decreases from 120 patients to 110. The improvements on the transition between the *Optimum found* results and *Satisfiable* is very important in our setting, since Genova Nervi and Castel Goffredo fall into this area, confirming the improvements obtained in Table 2. The loss with USC could be resolved by launching the two algorithms in two different threads, so the USC algorithm will perform better on instances with fewer patients while the BB algorithm will at least return a sub-optimal stable model for instances with more patients.

Validation of synthetic instances. In order to understand if the simulated instances correctly represent the real data and can be therefore used to predict the behavior of the system in larger institutes, a validation is needed to compare the results on real and synthetic instances. Intuitively, we have considered the data presented in Table 2 and compared it to the result of the instances within the circles around Genova Nervi and Castel Goffredo in Figures 3 and 4, to check if they "coincide". For doing so, a decision tree was trained, taking as dataset all the features of the simulated instances, some of them listed in the previous paragraph. Then, a test dataset with features extracted from the real instances was produced and given as an input to the decision tree, and the predicted result was then compared to the result given by CLINGO on the real instances. This test showed that for the board encoding, all the results on real instances were equal to the predicted ones for both institutes; the agenda encoding produced the same results in 93% of the cases for Genova Nervi and in 67% of the cases for Castel Goffredo, thus showing that overall the synthetic data behaves similarly as the real one and can be used for predicting the behavior of instances in larger institutes. Finally, the computed decision trees also confirm what are the most relevant features outlined above by inspecting the graphs in the figures.

5 Related Work and Conclusions

There have been few attempts to solve rehabilitation scheduling, since most hospitals are still doing it in a manual way. Among the automated solutions, often they are applied to real world data. However, their results are not directly comparable to ours, since their constraints and objective functions are different from the ones that emerged from our meetings with the physiotherapists and management at ICS Maugeri. In particular, to our knowledge, no other solution takes into account several aspects like the preferred time for the session scheduling and the preferences in the assignment of the patient to the operator. Huang, Zheng and Chien [17] developed a system, equipped with a Graphical User Interface, which can generate the optimal schedules for rehabilitation patients to minimize waiting time and thus enhance service quality and overall resource effectiveness of rehabilitation facilities. More recently, Huyinh, Huang and Chien [18] further

refined the algorithm in order to develop a hybrid genetic algorithm (GASA) that integrates genetic algorithm (GA) and simulated annealing (SA). Recently, Li and Chen [19] designed a genetic algorithm based on Waiting Time Priority Algorithm (WTPA) which was tested on a rehabilitation department. Schimmelpfeng, Helber and Kasper [24] developed a decision support system for the scheduling process based on mixed-integer linear programs (MILPs) to determine appointments for patients of rehabilitation hospitals, subject to numerous constraints that are often found in practice. We already mentioned in the introduction that ASP has been already successfully used for solving application problems in several research areas (see, e.g., [11, 22]), including the Healthcare domain (see, e.g., [2] for an overview). Differently from this set of papers in the same domain, the current work designs a two-phase encoding rather than a direct encoding, and evaluates the solution on real benchmarks.

In this paper, we have presented a two-phase ASP encoding for solving rehabilitation scheduling. Our solution has been tested with CLINGO and both real and synthetic benchmarks, the former provided by ICS Maugeri while the latter created with real parameters and employed to understand a possible behavior of the solution on upcoming institutes where the solution will be employed. Results are positive for the institutes employed at the moment and give some indications on the upcoming, e.g., there are few institutes that may fall close to the transition between satisfiable and unknown instances. Thus, despite the current positive results, a possible topic for future research is to improve the current encoding, as well as combining the strengths of the optimizations algorithms employed. Another interesting direction is to design also rescheduling solutions, to be applied in case of unavailability of operators and/or patients.

References

1. Alviano, M., Amendola, G., Dodaro, C., Leone, N., Maratea, M., Ricca, F.: Evaluation of disjunctive programs in WASP. In: Balduccini, M., Lierler, Y., Woltran, S. (eds.) Proceedings of the 15th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR 2019). Lecture Notes in Computer Science, vol. 11481, pp. 241–255. Springer (2019)
2. Alviano, M., Bertolucci, R., Cardellini, M., Dodaro, C., Galatà, G., Khan, M.K., Maratea, M., Mochi, M., Morozan, V., Porro, I., Schouten, M.: Answer set programming in healthcare: Extended overview. In: Joint Proceedings of the 8th IPS Workshop and the 27th RCRA Workshop co-located with AIxIA 2020. CEUR Workshop Proceedings, vol. 2745. CEUR-WS.org (2020)
3. Alviano, M., Dodaro, C.: Unsatisfiable core analysis and aggregates for optimum stable model search. Fundamenta Informaticae **176**(3-4), 271–297 (2020)
4. Alviano, M., Dodaro, C., Maratea, M.: An advanced answer set programming encoding for nurse scheduling. In: Esposito, F., Basili, R., Ferilli, S., Lisi, F.A. (eds.) Advances in Artificial Intelligence - Proceedings of the 16th International Conference of the Italian Association for Artificial Intelligence (AI*IA 2017). Lecture Notes in Computer Science, vol. 10640, pp. 468–482. Springer (2017)
5. Alviano, M., Dodaro, C., Marques-Silva, J., Ricca, F.: Optimum stable model search: algorithms and implementation. Journal of Logic and Computation **30**(4), 863–897 (2020)

6. Baral, C.: Knowledge Representation, Reasoning and Declarative Problem Solving. Cambridge University Press (2003). https://doi.org/10.1017/CBO9780511543357
7. Brewka, G., Eiter, T., Truszczynski, M.: Answer set programming at a glance. Communications of the ACM **54**(12), 92–103 (2011)
8. Calimeri, F., Faber, W., Gebser, M., Ianni, G., Kaminski, R., Krennwallner, T., Leone, N., Maratea, M., Ricca, F., Schaub, T.: ASP-Core-2 input language format. Theory and Practice of Logic Programming **20**(2), 294–309 (2020)
9. Cieza, A., Causey, K., Kamenov, K., Hanson, S.W., Chatterji, S., Vos, T.: Global estimates of the need for rehabilitation based on the Global Burden of Disease study 2019: a systematic analysis for the Global Burden of Disease Study 2019. The Lancet **396**(10267), 2006–2017 (2020), publisher: Elsevier
10. Dodaro, C., Maratea, M.: Nurse scheduling via answer set programming. In: Balduccini, M., Janhunen, T. (eds.) Proceedings of the 14th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR 2017). Lecture Notes in Computer Science, vol. 10377, pp. 301–307. Springer (2017)
11. Gebser, M., Kaminski, R., Kaufmann, B., Ostrowski, M., Schaub, T., Wanko, P.: Theory solving made easy with clingo 5. In: Carro, M., King, A., Saeedloei, N., Vos, M.D. (eds.) Proceedings of ICLP (Technical Communications). OASICS, vol. 52, pp. 2:1–2:15. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik (2016)
12. Gebser, M., Kaminski, R., Kaufmann, B., Romero, J., Schaub, T.: Progress in clasp Series 3. In: LPNMR. LNCS, vol. 9345, pp. 368–383. Springer (2015)
13. Gebser, M., Kaufmann, B., Schaub, T.: Conflict-driven answer set solving: From theory to practice. Artificial Intelligence **187**, 52–89 (2012)
14. Gebser, M., Maratea, M., Ricca, F.: The design of the seventh answer set programming competition. In: Balduccini, M., Janhunen, T. (eds.) LPNMR. Lecture Notes in Computer Science, vol. 10377, pp. 3–9. Springer (2017)
15. Gebser, M., Obermeier, P., Schaub, T., Ratsch-Heitmann, M., Runge, M.: Routing driverless transport vehicles in car assembly with answer set programming. Theory Practice of Logic Programming **18**(3-4), 520–534 (2018)
16. Gelfond, M., Lifschitz, V.: Classical Negation in Logic Programs and Disjunctive Databases. New Generation Computing **9**(3/4), 365–386 (1991)
17. Huang, Y.C., Zheng, J.N., Chien, C.F.: Decision support system for rehabilitation scheduling to enhance the service quality and the effectiveness of hospital resource management. J. of the Chinese Inst. of Industrial Engineers **29**, 348 – 363 (2012)
18. Huynh, N.T., Huang, Y.C., Chien, C.F.: A hybrid genetic algorithm with 2D encoding for the scheduling of rehabilitation patients. Computers & Industrial Engineering **125**, 221–231 (2018)
19. Li, X., Chen, H.: Physical therapy scheduling of inpatients based on improved genetic algorithm. Journal of Physics: Conference Series **1848**(1), 012009 (apr 2021)
20. Niemelä, I.: Logic Programs with Stable Model Semantics as a Constraint Programming Paradigm. AMAI **25**(3-4), 241–273 (1999)
21. Quinlan, J.R.: Induction of decision trees. Machine learning **1**(1), 81–106 (1986)
22. Ricca, F., Grasso, G., Alviano, M., Manna, M., Lio, V., Iiritano, S., Leone, N.: Team-building with answer set programming in the Gioia-Tauro seaport. Theory and Practice of Logic Programming **12**(3), 361–381 (2012)
23. Saverino, A., Baiardi, P., Galata, G., Pedemonte, G., Vassallo, C., Pistarini, C.: The challenge of reorganizing rehabilitation services at the time of covid-19 pandemic: A new digital and artificial intelligence platform to support team work in planning and delivering safe and high quality care. Frontiers in neurology **12**, 643251 (2021)
24. Schimmelpfeng, K., Helber, S., Kasper, S.: Decision support for rehabilitation hospital scheduling. OR Spectrum **34**(2), 461–489 (Apr 2012)