

# Scheduling Periodic Treatments via Answer Set Programming

Simone Caruso<sup>1</sup>, Carmine Dodaro<sup>2</sup>, Giuseppe Galatà<sup>3</sup> and Marco Maratea<sup>2</sup>

<sup>1</sup>*DIBRIS, University of Genoa, Genova, Italy*

<sup>2</sup>*DeMaCS, University of Calabria, Rende, Italy*

<sup>3</sup>*SurgiQ srl, Genova, Italy*

## Abstract

The scheduling of periodic treatments consists of planning a care path over a period of several weeks, in which patients have to perform different treatments respecting a certain periodicity. Treatments must be assigned to a day taking into account patients' preferences, operators' availability, the possibility of needing instruments/machinery, and of having different facilities in which operators can work and move around during the working time. This problem has been formulated to have a certain degree of flexibility and to be suitable for different contexts where periodicity is required, e.g. rehabilitation or planning a cycle of drug therapies. In this paper, we present a solution to this problem based on Answer Set Programming (ASP). We consider two different possible scenarios: scheduling one patient at a time, and scheduling blocks of patients. We have also conducted an experimental analysis showing that ASP is a suitable solution to this problem.

## Keywords

Answer Set Programming, Scheduling, Digital Health

## 1. Introduction

In recent years, hospitals have faced an ongoing challenge to improve the efficiency of their operations, especially after the COVID-19 pandemic emergency. This is due to the increasing healthcare expenditures and growing demand for healthcare services [1], which have led to numerous attempts to develop new patient admission and planning techniques.

This paper focuses on the problem of computing a schedule of periodic treatments, where patients must follow a predetermined care path of treatments. For example, patients may need to attend a specific number of physiotherapy sessions followed by a certain number of minutes of exercise. The schedule must consider several requirements, such as patient preferences, operator availability, periodicity of treatments, and more.

Scheduling problems like the one considered in this paper are typically addressed using logic formalisms such as Answer Set Programming (ASP) [2], which is a declarative programming paradigm that allows users to specify a problem in terms of rules and constraints. ASP systems

---


*CILC'23: 38th Italian Conference on Computational Logic, June 21–23, 2023, Udine, Italy*

✉ [simone.caruso@edu.unige.it](mailto:simone.caruso@edu.unige.it) (S. Caruso); [carmine.dodaro@unical.it](mailto:carmine.dodaro@unical.it) (C. Dodaro); [giuseppe.galata@surgiq.com](mailto:giuseppe.galata@surgiq.com) (G. Galatà); [maratea@mat.unical.it](mailto:maratea@mat.unical.it) (M. Maratea)

🆔 0000-0002-2724-4342 (S. Caruso); 0000-0002-5617-5286 (C. Dodaro); 0000-0002-1948-4469 (G. Galatà); 0000-0002-9034-2527 (M. Maratea)



© 2023 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

then generate all possible solutions that satisfy these rules and constraints. The clarity of the ASP syntax and its intuitive semantics, combined with the availability of several efficient solvers [3], make ASP a powerful tool for solving complex combinatorial problems. Indeed, due to its ability to handle large-scale problems, ASP is becoming increasingly popular in both academic [4, 5, 6, 7] and industrial settings [8]. In particular, ASP has been successfully employed to solve various real-world scheduling problems in the healthcare context, including the Nurse Scheduling Problem [9, 10], the Operating Room Scheduling Problem [11, 12, 13], the Chemotherapy Treatment Scheduling Problem [14] (see [15] for a recent survey).

In this paper, we follow the trend of such previous work and propose an ASP encoding to compute the scheduling of periodic treatments over a fixed period of time. We then present the results of a preliminary experimental analysis, demonstrating that the ASP system CLINGO [16], when executed on the presented encoding, achieves good performance in solving randomly-generated instances.

## 2. Problem Description

The problem considered in this paper consists in assigning multiple appointments to patients according to a care path in a period of several weeks. These assignments should take into account the availability of any instrument needed and the presence of the medical staff, if necessary. The patient's care path consists of sets of treatments that the patient should receive in the span of the planning period. Any set of treatments should be performed in the same day, minimizing the waiting times between the various treatments. A typical use case is that, after a medical examination, the patient comes with a precise list of treatments, e.g. 10 sessions of physiotherapy followed up by 10 minutes of cardio training, and ask for the scheduling of a certain number of appointments based on their availability and on the availability of the care facility. Patients may require more than one type of treatment, in this case, the different treatments are typically scheduled on the same day, one immediately after the other, so that the patient can come just one time for all of them. In this case, it is possible that it is provided an order to be respected. Patients may express time and day of the week preferences for their treatments as well as unavailable times and days.

Treatments can also have a weekly periodicity e.g. every week or every two weeks and a maximum and minimum number of sessions per week e.g once or twice a week, they can also require machines/instruments and operators with a specific qualification. It is possible that operators are not necessary for the total duration of the session, e.g. the operator starts the machine and then he/she does other tasks. Moreover, care facilities can be distributed among multiple locations and some operators can move from one to another during the day. This means that when assigning appointments also the travel time of the operator should be taken into account. Every care location can feature the relevant instruments/machinery for the type of care supported in the facility and has a number of qualified operators assigned accordingly.

An optimal solution assigns the patients' appointments in their desired days of the week, as close as possible from a given target week i.e. the week prescribed by the doctor and minimizes waiting time between treatments, while the operators' working time is daily maximized and weekly balanced among the available operators.

The main use case that our algorithm aims to solve is the situation when a patient calls a healthcare provider and receives a list of appointments in order to fulfill their care path. Then the patient can accept the proposed appointments or reject them and negotiate for other times, dates and locations.

However another use case is when the healthcare provider collects a waiting list of patients and assign their care paths at the same time. This approach has the advantage of being able to better optimize the working times of the operators and the instruments, however it makes the negotiating phase with the patients more difficult.

### 3. ASP Encoding

In this section, we first present the data model and the expected output (Section 3.1), then we describe the encodings for scheduling a single patient (Section 3.2) and multiple patients (Section 3.3). In the following, we assume the reader to be familiar with the ASP syntax and semantics (for more details we refer the reader to [17]).

#### 3.1. Data Model

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

- Instances of `reg(RID, CPID, PR, F)` represent the registrations characterized by an id (RID), the id of the associated care path (CPID), a priority level (PR) and a flag (F) that can be either `def` or `all`. The latter indicates that the patient requires that all the appointments must be scheduled in the same day of the week and all with the same starting time.
- Instances of `carePath(CPID, PER, NMINW, NMAXW, TAR)` represent a care path characterized by an id (CPID), a periodicity i.e. every how many weeks the treatments must be scheduled (PER), a minimum number of treatments per week (NMINW) and a maximum number of treatments per week (NMAXW) and the optimal week in which to start the care path (TAR).
- Instances of `treatment(TID, PCID, Q, TST, NOP, DORD, GORD)` represent a treatment characterized by an id (TID), the id of the associated care path (PCID), the necessary operator qualification (Q), the desirable starting time (TST), the number of necessary operators (NOP), an order value identifying the order between the treatments that have to be scheduled in the same day (DORD) and a global order identifying the order between the treatments to be scheduled in different appointments (GORD).
- Instances of `treatmentRequires(TID, PID, IID)` represent the place (PID) and the instrument (IID) required by a treatment (TID).
- Instances of `treatmentRequiresPlace(TID, PID)` represent the case in which a treatment (TID) requires only a place (PID) and no instruments.
- Instances of `treatmentDuration(TID, TODDUR, OPDUR, OPS, IDUR, IS)` represent the duration of a treatment characterized by the treatment id (TID), the total duration of the treatment (TODDUR), the duration with the operator(s) (OPDUR), the number of time slots that must pass from the start of the treatment before the operator(s) is needed (OPS), the duration with the instrument (IDUR) and the instrument starting time (IS).

- Instances of `place`(PID, ST, ET, D, W, MAXPAT) represent the places characterized by an id (PID), a starting time (ST), an ending time (ET), a day (D), the corresponding week (W) and the maximum capacity of the place (MAXPAT).
- Instances of `instrument`(IID, PID, ST, ET, D, W, MAXPAT) represent the instruments characterized by an id (IID), the place in which the instrument is located (PID), a starting time (ST), an ending time (ET), a day (D), the corresponding week (W) and the maximum capacity of the instrument (MAXPAT).
- Instances of `operator`(OPID, ST, ET, D, W, Q, MAXPAT) represent an operator characterized by an id (OPID), a starting time (ST), an ending time (ET), a day (D) and the corresponding week (W), the qualification/task assigned to the operator in that day (Q) and the maximum patients the operator can assist at the same time (MAXPAT).
- Instances of `travellingDuration`(PID1, PID2, DUR) represent the duration (DUR) of an operator movement from one place (PID1) to another (PID2).
- Instances of `compatible`(PID, IID1, IID2) represent the compatibility for an operator to stay at the same time on two different instruments (IID1 and IID2) and the place in which the instruments are located (PID).
- Instances of `desirableDays`(RID, D) represent the patients' (RID) preference of a day (D).
- Instances of `forbidden`(RID, ST, ET, D, W) represent a forbidden time for a patient characterized by a patient id (RID), a starting time (ST), an ending time (ET), the day (D) and the corresponding week (W).
- Instances of `time`(T, D, W) represent the available time slots (T) in a day (D) of a week (W).
- All the atoms presented in the next paragraph representing the patients and operators already scheduled.

**Output.** The output is specified by means of the following atoms:

- Instances of `x`(RID, TID, ST, ET, D, W, DORD, GORD) represent an assignment of a treatment characterized by a patient id (RID), a treatment id (TID), a day (D) and the corresponding week (W), a starting time (ST), an ending time (ET), the daily order of the treatment (DORD) and the global order value (GORD).
- Instances of `assignedPlace`(PID, TID, RID, ST, ET, D, W) represent the assignment of a patient to a place characterized by a place id (PID), a treatment id (TID), a patient id (RID), a day (D) and the corresponding week (W), a starting time (ST) and an ending time (ET).
- Instances of `assignedInstrument`(IID, PID, TID, RID, ST, ET, D, W) represent the assignment of a patient to an instrument/machine characterized by an instrument id (IID), a place id (PID), a patient id (RID), a treatment id (TID), an instrument id (IID), a day (D) and the corresponding week (W), a starting time (ST) and an ending time (ET).
- Instances of `assignedOpR`(OPID, RID, TID, PID, IID, D, W) represent the assignment of an operator to the resources of a treatment characterized by an operator id (OPID), a patient id (RID), a treatment id (TID), a place id (PID), an instrument id (IID), a day (D) and the corresponding week (W).

- Instances of `assignedOpT(OPID,RID,TID,ST,ET)` represent the assignment of an operator (OPID) to a patient's (RID) treatment (TID) with a starting time (ST) and an ending time (ET).
- Instances of `prevWork(OPID,W,S)` represent the sum of all the working hours (S) of the operators (OPID) for each week (W).

### 3.2. Encoding for Single Patient Scheduling

The encoding for the scheduling of a single patient is described in the following.

As first operation, we assign a day and a week to the first treatment of the new registration using the following rule:

```
1      {assignedDay(RID,TID,D,W,0,0): time(_,D,W)} = 1 :-
      reg(RID,CPID,_,def), carePath(CPID,_,_,_,_),
      treatment(TID,CPID,_,_,_,0,0).
```

Then, for each treatment, we compute the assignable weeks using the following rule:

```
2      assignableWeek(RID,TID,W) :- reg(RID,CPID,_,_),
      assignedDay(RID,_,_,SW,0,0), time(_,_,W),
      treatment(TID,CPID,_,_,_,0,GORD), carePath(PCID,PER,NMIN,NMAX,_) ,
      (SW-W)\PER = 0, W >= SW + (GORD/NMAX), W <= SW + (GORD/NMIN).
```

Assignable weeks range between the minimum and the maximum number of weekly visits, and they are used by the following rule:

```
3      {assignedDay(RID,TID,D,W,0,GORD) : assignableWeek(RID,TID,W),
      time(_,D,W), W >= PW} <= 1 :- reg(RID,CPID,_,def),
      carePath(CPID,_,_,_,_), treatment(TID,CPID,_,_,_,0,GORD), GORD >
      0, assignedDay(RID,_,_,PW,_,GORD-1).
```

The above rule assigns, for each initial daily treatment (different treatments can be needed in the same day) a day and a week (among the ones in the assignable weeks).

We then handle the case where the patient requires that all the appointments must be scheduled in the same day of the week and all with the same starting time (expressed by the latest parameter of the registration set to `all`). In this the case, the following rules are used to first select the days of the week, and the starting times of the treatments, and then, to assign the day and the week for each treatment:

```
4      {chosenDays(RID,D): time(_,D,_) } = NMAX:-
      carePath(PCID,PER,NMIN,NMAX,_) , reg(RID,PCID,_,all).
5      {chosenDays(RID,D,T): time(T,D,_) } = 1 :- chosenDays(RID,D).
6      {assignedDay(RID,TID,D,W,0,0): time(_,D,W), chosenDays(RID,D,T),
      availableTime(RID,TID,T,D,W)} = 1 :- reg(RID,CPID,_,all),
      carePath(CPID,_,_,_,_) , treatment(TID,CPID,_,_,_,0,0).
7      {assignedDay(RID,TID,D,W,0,GORD) : assignableWeek(RID,TID,W),
      time(_,D,W), chosenDays(RID,D,T), availableTime(RID,TID,T,D,W), W
      >= PW } <= 1 :- reg(RID,CPID,_,all), carePath(CPID,_,_,_,_),
      treatment(TID,CPID,_,_,_,0,GORD), assignedDay(RID,_,_,PW,_,GORD-1).
```

Subsequently, if there are multiple treatments to be performed in the same day, they are assigned to the initial daily treatment using the following rule:

```

8      assignedDay(RID,TID,D,W,DORD,GORD) :- reg(RID,CPID,_,_),
      carePath(CPID,_,_,_), assignedDay(RID,_,D,W,0,GORD),
      treatment(TID,CPID,_,_,DORD,GORD), DORD > 0.

```

Then, we have to ensure that (i) between two weeks there is not a skipped week, taking into account that the periodicity of the care path; (ii) the order between the treatments is respected, therefore the treatments with greater global order are assigned after those with lower global order; (iii) the minimum and maximum numbers of weekly treatments are fulfilled. The following constraints are used in this respect:

```

9      :- assignedDay(RID,_,_,W1,_,_), assignedDay(RID,_,_,W2,_,_), not
      assignedDay(RID,_,_,W1+PER,_,_), W2 > W1, carePath(CPID,PER,_,_,_).
10     :- assignedDay(RID,TID1,D1,W,0,GORD1),
      assignedDay(RID,TID2,D2,W,0,GORD2), GORD2 > GORD1, D2 <= D1.
11     :- assignedDay(RID,_,_,W,_,_), #max{WM: assignedDay(RID,_,_,WM,_,_)} >
      W, reg(RID,CPID,_,_), #count{D: assignedDay(RID,_,D,W,_,_)} = N, N
      < NMIN, carePath(CPID,_,NMIN,_,_).
12     :- assignedDay(RID,_,_,W,_,_), reg(RID,CPID,_,_), #count{D:
      assignedDay(RID,_,D,W,_,_)} = N, N > NMAX,
      carePath(CPID,_,_,NMAX,_) .

```

Then, we define a support atom, namely availableTime(RID,TID,T,D,W), representing the time slots in which each treatment can be scheduled:

```

13     forbiddenT(RID,STF..ETF,D,W) :- reg(RID,_,_,_),
      forbidden(RID,D,STF,ETF,W).
14     maxStartingTime(RID,TID1,eot-TOTDUR1-M) :- reg(RID,CPID,_,_),
      treatment(TID1,CPID,_,_,DORD1,GORD),
      treatmentDuration(TID1,TOTDUR1,_,_,_), #sum{TOTDUR2,TID2:
      treatment(TID2,CPID,_,_,DORD2,GORD),
      treatmentDuration(TID2,TOTDUR2,_,_,_), DORD2 > DORD1} = M.
15     minStartingTime(RID,TID1,M) :- reg(RID,CPID,_,_),
      treatment(TID1,CPID,_,_,DORD1,GORD), #sum{TOTDUR2,TID2:
      treatment(TID2,CPID,_,_,DORD2,GORD),
      treatmentDuration(TID2,TOTDUR2,_,_,_), DORD2 < DORD1} = M.
16     availableTime(RID,TID,T,D,W) :- maxStartingTime(RID,TID,MAX),
      minStartingTime(RID,TID,MIN), time(T,D,W),
      treatmentDuration(TID,DUR,_,_,_), T >= MIN - DUR, T <= MAX.

```

This represents a domain specific optimization, since there is an order between the treatments we know the first and last possible assignable time slot for each treatment, so that the treatments that come before and/or after have enough space.

The defined support atom is then used to assign a starting time to each treatment by means of the following rules:

```

17     {x(RID,TID,ST,ST+TOTDUR,D,W,0,GORD) : availableTime(RID,TID,ST),
      time(ST,D,W), treatmentDuration(TID,TOTDUR,_,_,_) } = 1 :-
      reg(RID,_,_,def), assignedDay(RID,TID,D,W,0,GORD).
18     x(RID,TID,ST,ST+TOTDUR,D,W,0,GORD) :- reg(RID,_,_,all),
      availableTime(RID,TID,ST,D,W), chosenDays(RID,D,ST),
      treatmentDuration(TID,TOTDUR,_,_,_),
      assignedDay(RID,TID,D,W,0,GORD).

```

```

19      {x(RID, TID, ST, ST+TOTDUR, D, W, DORD, GORD) : availableTime(RID, TID, ST),
        time(ST, D, W), treatmentDuration(TID, TOTDUR, _, _, _, _), ST >= ET } =
        1 :- reg(RID, _, _, _), assignedDay(RID, TID, D, W, DORD, GORD),
          x(RID, _, _, ET, D, W, DORD-1, GORD).

```

In particular, it is first assigned the starting time of the first treatment, i.e. the one with daily order equal to zero and then, it is assigned the starting time to the following treatments ensuring that they are assigned after the ending time of the one before.

Then, we compute the starting and the ending time of the usage of an instrument by a patient and we ensure that, for each time slot, the capacity of the instruments is not exceeded.

```

20      assignedInstrument(IID, PID, TID, RID, ST+MIS, ST+MIS+MIDUR, D, W) :-
        reg(RID, _, _, _), instrument(IID, PID, STM, ETM, D, W, _), ST+MIS >= STM,
        ST+MIS+MIDUR <= ETM, x(RID, TID, ST, ET, D, W, _, _),
        treatment(TID, _, _, _, _, _), treatmentRequires(TID, PID, IID),
        treatmentDuration(TID, _, _, _, MIDUR, MIS).
21      :- #count{RID: assignedInstrument(IID, PID, TID, RID, ST, ET, D, W), T >= ST,
        T < ET} > MAXPAT, instrument(IID, PID, _, _, D, W, MAXPAT), time(T, D, W).

```

Similarly, we compute the starting time and ending time of the assignment of a patient to a place and we ensure that, for each time slot, the capacity of the place is not exceeded:

```

22      assignedPlace(PID, TID, RID, ST, ET, D, W) :- reg(RID, _, _, _),
        place(PID, STL, ETL, D, W, _), ST >= STL, ET <= ETL,
        x(RID, TID, ST, ET, D, W, _, _), treatment(TID, _, _, _, _, _),
        treatmentRequires(TID, PID, IID).
23      assignedPlace(PID, TID, RID, ST, ET, D, W) :- reg(RID, _, _, _),
        place(PID, STL, ETL, D, W, _), ST >= STL, ET <= ETL,
        x(RID, TID, ST, ET, D, W, _, _), treatment(TID, _, _, _, _, _),
        treatmentRequiresPlace(TID, PID).
24      :- #count{RID: assignedPlace(PID, _, RID, ST, ET, D, W), T >= ST, T < ET} >
        MAXPAT, place(PID, _, _, D, W, MAXPAT), time(T, D, W).

```

Then, we assign to each treatment, the required operators:

```

25      {assignedOp(OPID, RID, TID, STO, ETO) : operator(OPID, STO, ETO, D, W, Q, _) } =
        NOP :- reg(RID, _, _, _), treatment(TID, _, Q, _, _, NOP, _, _).

```

Subsequently, we compute the place and the instrument assigned to the operator in a given day and the corresponding patient, and the starting and ending time of the operator assignment:

```

26      assignedOpR(OPID, RID, TID, PID, IID, D, W) :- assignedOp(OPID, RID, TID, _, _),
        x(RID, TID, _, _, D, W, _, _), reg(RID, _, _, _),
        treatmentRequires(TID, PID, IID).
27      assignedOpR(OPID, RID, TID, PID, -1, D, W) :- assignedOp(OPID, RID, TID, _, _),
        x(RID, TID, _, _, D, W, _, _), reg(RID, _, _, _),
        treatmentRequiresPlace(TID, PID).
28      opidSt(RID, TID, STX+OPS, STX+OPS+OPDUR) :- reg(RID, _, _, _),
        treatmentDuration(TID, _, OPDUR, OPS, _, _), x(RID, TID, STX, ETX, _, _, _, _).
29      assignedOpT(OPID, RID, TID, ST, ET) :- assignedOp(OPID, RID, TID, _, _),
        opidSt(RID, TID, ST, ET).
30      :- assignedOpT(OPID, RID, TID, ST, ET), assignedOp(OPID, RID, TID, STO, ETO),
        ST < STO.

```

```

31     :- assignedOpT(OPID,RID,TID,ST,ET), assignedOp(OPID,RID,TID,STO,ETO),
      ET > ETO.

```

Then, we ensure that the operators do not move in different places in less than the necessary travelling time and that their capacity is not exceeded for each time slot:

```

32     :- assignedOpR(OPID,RID1,TID1,PID1,_,D,W),
      assignedOpT(OPID,RID1,TID1,_,ET1),
      assignedOpR(OPID,RID2,TID2,PID2,_,D,W),
      assignedOpT(OPID,RID2,TID2,ST2,_), ST2 >= ET1,
      travellingDuration(PID1,PID2,DUR), ST2 < ET1 + DUR.
33     :- #count{RID: support(OPID,RID,T,_,_,D,W)} > MAXPAT,
      operator(OPID,_,_,_,_,MAXPAT), time(T,D,W).

```

Moreover, we also ensure that an operator is not assigned in two different places at the same time and to two different instruments that are not compatible at the same time:

```

34     support(OPID,RID,ST,ET-1,PID,IID,D,W) :-
      assignedOpT(OPID,RID,TID,ST,ET),
      assignedOpR(OPID,RID,TID,PID,IID,D,W).
35     :- support(OPID,_,T,LID1,_,D,W), support(OPID,_,T,LID2,_,D,W), LID1 <
      LID2.
36     :- support(OPID,_,T,PID,MID1,D,W), support(OPID,_,T,PID,MID2,D,W), not
      compatible(PID,MID1,MID2), MID1 < MID2.

```

Finally, we are ready to express the preferences among different solutions by means of the following rules:

```

37     ~ assignedDay(RID,_,_,W,_,0), carePath(CPID,_,_,_,TAR). [|W-TAR|@10]
38     ~ reg(RID,_,_,_), treatment(TID,_,_,_,_,_), not
      assignedDay(RID,TID,_,_,_,_). [1@9,TID]
39     ~ reg(RID,_,_,_), assignedDay(RID,_,_,W,_,_). [1@8,RID,W]
40     ~ reg(RID,_,_,_), assignedDay(RID,TID,D,_,_,_), not
      desirableDays(RID,D). [1@7,TID]
41     ~ reg(RID,_,_,_), x(RID,TID1,_,ET1,_,_,_,_),
      x(RID,TID2,ST2,_,_,_,_,_), treatment(TID1,_,_,_,DORD,GORD),
      treatment(TID2,_,_,_,DORD+1,GORD), ST2-ET1 > 0.
      [ST2-ET1@6,TID1,TID2]
42     ~ reg(RID,_,_,_), x(RID,TID,ST,_,_,_,_,_),
      treatment(TID,_,_,TST,_,_). [|ST-TST|@5,TID]
43     ~ reg(RID,_,_,_), assignedOp(OPID1,RID,TID1,_,_),
      assignedOp(OPID2,RID,TID2,_,_), TID1 < TID2, OPID1 != OPID2.
      [1@4,TID1,TID2]
44     ~ time(_,D,W), #count{OPID: assignedOpR(OPID,_,_,_,_,D,W)}=N.
      [N@3,D,W]
45     movement(OPID,LID1,LID2,ET1,D,W) :- reg(RID1,_,_,_),
      treatment(TID1,_,_,_,_,_,_), RID1 != RID2,
      assignedOpR(OPID,RID1,TID1,LID1,_,D,W),
      assignedOpT(OPID,RID1,TID1,ST1,ET1),
      assignedOpR(OPID,RID2,TID2,LID2,_,D,W),
      assignedOpT(OPID,RID2,TID2,ST2,_), ST2 > ST1, LID2 != LID1,

```



```

#count{ST3: assignedOpR(OPID,RID3,TID3,_,_,D,W), RID1 != RID3,
assignedOpT(OPID,RID3,TID3,ST3,_) , ST3 > ST1, ST3 < ST2} < 1.
46 movement(OPID,LID2,LID1,ET2,D,W) :- reg(RID1,_,_,_),
    treatment(TID1,_,_,_,_,_), RID1 != RID2,
    assignedOpR(OPID,RID1,TID1,LID1,_,D,W),
    assignedOpT(OPID,RID1,TID1,ST1,_) ,
    assignedOpR(OPID,RID2,TID2,LID2,_,D,W),
    assignedOpT(OPID,RID2,TID2,ST2,ET2), ST1 > ST2, LID2 != LID1,
    #count{ST3: assignedOpR(OPID,RID3,TID3,LID3,_,D,W), RID1 != RID3,
    assignedOpT(OPID,RID3,TID3,ST3,_) , ST3 > ST2, ST3 < ST1} < 1.
47 :~ movement(OPID,LID1,LID2,T,D,W) . [1@3,OPID,LID1,LID2,T,D,W]
48 :~ time(_,D,W), support(OPID,_,T,_,_,D,W), not
    support(OPID,_,T+1,_,_,D,W) . [1@2,OPID,T,D,W]
49 totWorkOp(OPID,W,0..S+T) :- time(.,.,W), prevWork(OPID,W,S),
    #sum{OPDUR,TID: assignedOpR(OPID,RID,TID,_,_,_,W), reg(RID,_,_,_),
    treatmentDuration(TID,_,OPDUR,_,_,_)} = T,
    operator(OPID,_,_,_,_,_,_).
50 :~ time(.,.,W), totWorkOp(OPID1,W,T), operator(OPID1,_,_,_,_,_,_),
    not totWorkOp(OPID2,W,T), operator(OPID2,_,_,_,_,_,_) . [1@1,W]

```

In particular, rule 37 is used to minimize the distance in absolute value between the week in which the care path actually starts and the optimal one. Then, since it is possible that not all the treatments are scheduled (for instance when there are not enough operators), we have to maximize the number of assigned treatments (rule 38), and we also maximize the number of weekly treatments (rule 39), this is obtained by minimizing the total weeks. In rule 40, it is penalized the case where a treatment is not assigned to a day preferred by the patient and then, in the subsequent rule, it is minimized the waiting time between the treatments. Rule 42 is used to minimize the distance between the starting time of a treatment and the one desired by the patient, whereas rule 43 enforces, as much as possible, to assign the same operator to a patient for each treatment. Then, in rules 44 and 47 we have the two possible operator optimization strategies, the care facility can choose the one to adopt. In particular, rule 44 minimizes the number of operators working for each day, whereas rule 47 minimizes the daily movements of the operators. Finally, rules 48 and 50 balance the total working times of the operators for each week.

### 3.3. Encoding for Multiple Patients Scheduling

For scheduling multiple patients at the same time, we require as input a list of patients with the corresponding care pathway, available instruments, operators and facilities, and information inherent to instrument compatibility, duration moves and patient preferences.

The encoding is then the same presented in Section 3.2, where only the following rules are modified to be adapted to the presence of multiple patients.

In particular, rule 9 is replaced as follows:

```

9 :- reg(RID,CPID,_,_), assignedDay(RID,_,_,W1,_,_),
    assignedDay(RID,_,_,W2,_,_), not assignedDay(RID,_,_,W1+PER,_,_),
    W2 > W1, carePath(CPID,PER,_,_,_).

```

In this case, we ensure that, for each patient, a week of treatment is not skipped.

Moreover, rule 32 is replaced as follows:

```
32      :- support(OPID,RID1,T1,PID1,_,D,W), support(OPID,RID2,T2,PID2,_,D,W),
        RID1 != RID2, T1 < T2, T2 < T1 + DUR + 1,
        travellingDuration(PID1,PID2,DUR).
```

As in the previous encoding, we ensure that the operators do not move in different places in less than the necessary travelling time.

Then, rules 37, 38, and 42 are slightly modified as follows to be suitable for the scheduling of a list of patients:

```
37      :~ reg(RID,CPID,_,_), assignedDay(RID,_,_,W,_,0),
        carePath(CPID,_,_,_,TAR). [|W-TAR|@10,RID]

38      :~ reg(RID,CPID,_,_), treatment(TID,CPID,_,_,_,_), not
        assignedDay(RID,TID,_,_,_,_). [|1@9,TID]

42      :~ reg(RID,CPID,_,_), x(RID,TID,ST,_,_,_,_),
        treatment(TID,CPID,_,TST,_,0,_) . [|ST-TST|@5,TID]
```

## 4. Preliminary Experimental Analysis

In this section we report the results of an empirical analysis conducted to assess the performance of the proposed encoding. The experiments were run on a AMD Ryzen 5 3600 CPU @ 3.60GHz with 16 GB of physical RAM. As ASP system, we used CLINGO [16] version 5.4.1, configured with the option `--parallel-mode 6` for parallel execution. This setting is the result of a preliminary analysis done on the same instances where we tested also other parameters, e.g., `--opt-strategy=usc` for optimization. The time limit was set to 60 seconds for the scheduling of single patients and 300 seconds for the scheduling of multiple patients. Encodings and benchmarks employed in this section can be found at: [https://drive.google.com/file/d/1OhYaP\\_7dfSRzZ8GyWzRgAdb84SRWtkOp/view](https://drive.google.com/file/d/1OhYaP_7dfSRzZ8GyWzRgAdb84SRWtkOp/view).

**Benchmarks.** Data used in the experiments were randomly generated. In particular, we considered a period of 10 weeks, each with 7 working days and, for each day, we considered 60 time slots. Moreover, we considered a scenario with 4 operators that can move between two facilities, one with 3 instruments and the other with 2 instruments. Then, for each patient we generated a minimum and maximum number of weekly treatments. For each patient, we considered two possible ranges of such values, i.e. 1 (min) and 2 (max), or 2 (min) and 3 (max). The optimal week in which start the treatments was set in the first 4 weeks and the number of treatments per appointment, instead, varied between 2 and 3 treatments and with a duration ranging from 3 to 12 time slots.

While these parameters were held constant, patient preferences and their number of treatments were varied. Due to the high flexibility of the problem several scenarios are possible:

1. Scenario 1: we have a scheduling of a single patient at a time with a variable number of treatments, and a daily minimization of the total number of operators, and for each patient we have at least 1 and at most 2 weekly treatments.

**Table 1**

Results of the experiments in the Scenario 1.

Number of treatments	Avg time
8 treatments (in 4 appointments)	3.39s
12 treatments (in 5 appointments)	4.95s
18 treatments (in 6 appointments)	7.51s

**Table 2**

Results of the experiments in the Scenario 2.

Number of treatments	Avg time
12 treatments (in 6 appointments)	5.18s
18 treatments (in 7 appointments)	8.1s
24 treatments (in 12 appointments)	10.77s
27 treatments (in 9 appointments)	13.44s
36 treatments (in 12 appointments)	18.84s

**Table 3**

Results of the experiments in the Scenario 3.

Number of treatments	Avg time
12 treatments (in 6 appointments)	5.91s
18 treatments (in 7 appointments)	10.49s
27 treatments (in 9 appointments)	20.4s

**Table 4**

Results of the experiments in the Scenario 4.

Number of treatments	Avg time
12 treatments (in 6 appointments)	6.5s
18 treatments (in 7 appointments)	10.32s
27 treatments (in 9 appointments)	15.25s

2. Scenario 2: we have a scheduling of a single patient at a time with a variable number of treatments, and a daily minimization of the total number of operators, and for each patient we have at least 2 and at most 3 weekly treatments.
3. Scenario 3: as in Scenario 2, where patients require the same weekly day and start time.
4. Scenario 4: as in Scenario 2, where instead of minimizing the number of daily working operators, it is minimized the number of operators' displacements.
5. Scenario 5: as in Scenario 1 (Scenario 5a) and in Scenario 2 (Scenario 5b), where we consider blocks of patients of sizes 5 and 10 per instance.

**Table 5**

Results of the experiments in the Scenario 5a and 5b.

Scenario	N.Pat/instance	Avg.Treat/instance	Avg.Time	% OPT
5a	5 patients	57	48.52s	100%
5a	10 patients	125	246.54s	60%
5b	5 patients	93	126.94s	90%
5b	10 patients	188	299s	10%

**Results with Patients Scheduled One by One.** Tables 1, 2, and 3 show the results when it is required to minimize the daily number of working operators. Table 1 shows the average solving time for the instances with the minimum and the maximum number of weekly appointments for each patient set to 1 and 2, while Table 2 shows the average solving time for the instances with the minimum and the maximum number of weekly appointments for each patient set to 2 and 3.

It is possible to observe that the solving time depends on the number of treatments. Table 3 shows the results obtained by generating patients who always require to keep the weekly days and start times fixed. In table 4 are shown the result using the operator optimization strategy: minimize daily movements. Results show that the optimum is always found with average time less than 20 seconds. Patients are always assigned in their ideal starting week, respecting weekly day preferences, and in most cases all treatments are scheduled, however as the number of patients increases and the agenda becomes saturated some treatments are not assigned. In addition, waiting times between treatments are minimal, increasing as the agenda becomes thicker due to the fact that greater priority is given to respect the target start week and patient day preferences. Moreover, the analysis shows that the operators' weekly workloads are correctly distributed, with an average distance, for each operator, from the weekly average of less than 15 time slots for all the instances.

**Results Scheduling Multiple Patients at the Same Time.** Scheduling blocks of patients allows us to study the scalability of the proposed solution and can be used to optimize the work time of operators and instruments. The results are shown in Table 5. We used blocks of 5 and 10 patients and varied the average number of total treatments per instance, i.e, the average number of treatments per patient. The results show that for 5 patients the solver in the majority of the cases finds the optimum solution. However, the first levels are minimized also in the non-optimal solutions ensuring a certain quality of the scheduling so that patient preferences and waiting times are respected.

For blocks of 10 patients, on the other hand, as the input grows, more time is spent in finding solutions and several instances do not reach the optimum but a (sub-optimal) solution is still found before the timeout. Only in one case the optimization stops at the first level while in the other sub-optimal results the first levels are minimized respecting patient preferences, assigning all assignable visits and minimizing waiting times.

## 5. Related Work

In [1] a review of the literature on multi-appointment scheduling problems in hospitals, i.e., problems in which patients need appointments for different resources, is presented and shows how research interest in these problems has grown only in recent years.

Problems similar to the one presented in this paper are found only in particular use cases such as rehabilitation, chemotherapy and radiotherapy. In [18] a solution to the problem of chemotherapy treatments is proposed. Different treatments are considered for each patient but the days between one appointment and the following are fixed. Deciding the starting day, therefore, also fixes all the subsequent visits. The problem is modeled as an integer linear program and tests were conducted using CPLEX solver.

In [19], a model for scheduling radiotherapy patients is presented. In this problem a certain number of weekly treatments must be scheduled for each patient. An Integer program formulation tested with LINGO is proposed.

Several studies have also been conducted in the field of physical therapy and different models have been proposed as in [20] and [21]. A series of appointments are required in this problem that also involve different resources and disciplines.

However, there are no general solutions for these kind of problems as the one presented in this paper that is designed to be flexible and able to adapt to different contexts where different appointments, involving different operators and resources, are required with a certain frequency and in a time horizon of several weeks.

## 6. Conclusion

In this paper, we provided an ASP solution for the problem of computing a schedule of periodic treatments for patients who must follow a predetermined care path of treatments. The results of a preliminary experimental analysis show that CLINGO executed on the presented encoding achieves good performance in solving randomly-generated instances. Concerning future work, we aim at finding a simpler encoding without sacrificing the performance, and at extending the experimental analysis by considering a larger numbers of operators and patients, or by considering alternative ASP solving techniques [22].

## References

- [1] J. Marynissen, E. Demeulemeester, Literature review on multi-appointment scheduling problems in hospitals, *European Journal of Operational Research* 272 (2019) 407–419. URL: <https://www.sciencedirect.com/science/article/pii/S0377221718302108>. doi:<https://doi.org/10.1016/j.ejor.2018.03.001>.
- [2] G. Brewka, T. Eiter, M. Truszczynski, Answer set programming at a glance, *Communications of the ACM* 54 (2011) 92–103.
- [3] 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](http://ijcai.org), 2018, pp. 5450–5456.

- [4] E. Erdem, M. Fidan, D. F. Manlove, P. Prosser, A general framework for stable roommates problems using answer set programming, *Theory Pract. Log. Program.* 20 (2020) 911–925.
- [5] E. Erdem, V. Patoglu, Applications of ASP in robotics, *Künstliche Intell.* 32 (2018) 143–149.
- [6] A. D. Palù, A. Dovier, A. Formisano, E. Pontelli, ASP applications in bio-informatics: A short tour, *Künstliche Intell.* 32 (2018) 157–164.
- [7] M. Gavanelli, M. Nonato, A. Peano, An ASP approach for the valves positioning optimization in a water distribution system, *Journal of Logic and Computation* 25 (2015) 1351–1369.
- [8] G. Grasso, N. Leone, M. Manna, F. Ricca, ASP at work: Spin-off and applications of the DLV system, in: *Logic Programming, Knowledge Representation, and Nonmonotonic Reasoning*, volume 6565 of *LNCS*, Springer, 2011, pp. 432–451.
- [9] C. Dodaro, M. Maratea, Nurse scheduling via answer set programming, in: *LPNMR*, volume 10377 of *LNCS*, Springer, 2017, pp. 301–307.
- [10] 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.
- [11] 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.
- [12] 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.
- [13] C. Dodaro, G. Galatà, M. K. Khan, M. Maratea, I. Porro, Operating room (re)scheduling with bed management via ASP, *Theory Pract. Log. Program.* 22 (2022) 229–253.
- [14] 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.
- [15] 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. URL: <http://ceur-ws.org/Vol-2745/paper7.pdf>.
- [16] 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.
- [17] 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 Pract. Log. Program.* 20 (2020) 294–309.
- [18] A. Condotta, N. Shakhlevich, Scheduling patient appointments via multilevel template: A case study in chemotherapy, *Operations Research for Health Care* 3 (2014) 129–144. URL: <https://www.sciencedirect.com/science/article/pii/S221169231400006X>. doi:<https://doi.org/10.1016/j.orhc.2014.02.002>.
- [19] D. Conforti, F. Guerriero, R. Guido, Optimization models for radiotherapy patient scheduling, *4or* 6 (2008) 263–278.

- [20] K. Schimmelpfeng, S. Helber, S. Kasper, Decision support for rehabilitation hospital scheduling, *OR spectrum* 34 (2012) 461–489.
- [21] A. Braaksma, N. Kortbeek, G. Post, F. Nollet, Integral multidisciplinary rehabilitation treatment planning, *Operations Research for Health Care* 3 (2014) 145–159. URL: <https://www.sciencedirect.com/science/article/pii/S2211692314000058>. doi:<https://doi.org/10.1016/j.orhc.2014.02.001>.
- [22] G. Mazzotta, F. Ricca, C. Dodaro, Compilation of aggregates in ASP systems, in: *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022*, AAAI Press, 2022, pp. 5834–5841. URL: <https://ojs.aaai.org/index.php/AAAI/article/view/20527>.