Journal of Advances in Developmental Research (IJAIDR)
E-ISSN: 0976-4844 ● Website: www.ijaidr.com ● Email: editor@ijaidr.com

AI-Driven Physiotherapy Scheduling for Obesity Treatment: Optimizing Exercise-Based Rehabilitation Programs

V.V. Manjula Kumari

CEO and Senior Consultant for Obesity
Varanaa's Health Care Research and Training Organization, India
Email: varanasimanju78@gmail.com

Abstract
Effective physiotherapy is essential for managing obesity-related mobility issues and improving functional movement in overweight individuals. However, challenges in rehabilitation scheduling often limit the accessibility of tailored exercise programs. In this paper, I propose an AI-powered physiotherapy scheduling framework that optimizes obesity-focused rehabilitation programs by efficiently assigning therapists, gym resources, and patient sessions. The system utilizes Answer Set Programming (ASP) to streamline exercise scheduling for obese patients, ensuring consistency in treatment, therapist allocation, and optimal session timing. The scheduling model incorporates patient preferences, obesity-specific rehabilitation constraints, and session optimization strategies. I apply the framework to real-world physiotherapy centers, demonstrating its effectiveness in enhancing accessibility and adherence to weight-loss physiotherapy programs. Experimental results show that the AI-driven scheduling system improves session availability, reduces therapist workload imbalance, and enhances rehabilitation efficiency. This research highlights the potential of AI in optimizing physiotherapy planning for obesity treatment, ensuring more structured and accessible exercise-based interventions.

I. INTRODUCTION

Rehabilitation scheduling revolves around structuring the daily physiotherapy sessions for patients within a rehabilitation center. This scheduling challenge, referred to as the Rehabilitation Scheduling Problem (RSP), has been extensively examined in prior research (1; 2; 3; 4). Institutions like ICS Maugeri, which serves as a reference point in this study, often accommodate hundreds of patients while operating with a restricted number of physiotherapists. Consequently, efficient allocation of physiotherapists to patients is crucial. According to a recent study conducted by Cieza et al. (5), around 2.41 billion people worldwide stand to gain from rehabilitation services. This statistic indicates that nearly one- third of the global population may need rehabilitation due to various health conditions or physical injuries. Moreover, the lasting consequences of the COVID-19 pandemic have further intensified the demand for rehabilitation services, regardless of hospitalization status or disease severity.
The RSP is subject to multiple constraints, including legal, medical, and ethical considerations. Key

constraints involve facility capacity limits, legal work hours and mandatory rest periods for physiotherapists, as well as the minimum duration of therapy sessions. Additionally, patient preferences must be accommodated, as maintaining consistency with the same physiotherapist and adhering to regular schedules can significantly enhance recovery outcomes. Another important factor is ensuring an equitable workload distribution among physio- therapists, which contributes to overall scheduling efficiency. This paper introduces an Answer Set Programming (ASP)- based solution (6; 7; 8; 9) for addressing the RSP, lever- aging its effectiveness in solving complex scheduling problems (10; 11; 12; 13). The methodology employs a two- step encoding framework (Section III). The first step, termed *board*, is responsible for assigning physiotherapists to patients while ensuring compliance with work-hour limits and session duration requirements. The second step, known as *agenda*, determines the exact timing of sessions based on the assignments established in the first step. Although this approach does not guarantee an optimal solution, it simplifies implementation and encoding. Furthermore, it aligns with the manual scheduling procedures at ICS Maugeri, allowing coordinators to make adjustments before finalizing schedules. This flexibility is critical, as occasional manual modifications help integrate expert knowledge into the scheduling process. It is important to emphasize that this system is a decision- support tool rather than a medical device, meaning that legal requirements necessitate final oversight by coordinators.

The proposed approach was validated (Section IV) using real-world data from ICS Maugeri, specifically focusing on the daily scheduling of neurological patients at two rehabilitation centers in northern Italy: Genova Nervi and Castel Goffredo. For evaluation purposes, the ASP solver *clingo* (14) was given a runtime limit of 30 seconds, whereas in production settings, a limit of 5 minutes is applied. The reduced evaluation runtime allowed for a broader comparison of multiple optimization algorithms within *clingo*. Given that ICS Maugeri aims to extend automation to larger facilities, a large dataset of synthetic benchmarks, inspired by real-world data, was also created. Using classification decision tree techniques (15), a comparative analysis was conducted between synthetic and real-world datasets to predict system performance in larger settings. The results demonstrated strong predictive accuracy, validating the relevance of synthetic benchmarks in estimating system behavior. Additionally, this analysis identified key problem characteristics that influence scheduling efficiency.

To enhance performance and minimize unresolved cases, domain-specific optimizations were incorporated into the encoding (Section V). The optimized encoding successfully yielded feasible solutions for all tested scenarios within the predefined time limits, although not all solutions were optimal. The remainder of this paper presents an informal problem definition in Section II, discusses related work in Section VI, and concludes with final observations in Section VII.

II. PROBLEM DESCRIPTION

This section provides a structured explanation of the problem in four key segments. First, a general problem overview is presented, followed by a description of the key data elements defining the problem. Next, the scheduling requirements for different phases are outlined, and finally, a structured solution framework is provided.

General Overview. The process of scheduling rehabilitation services presents a significant logistical challenge, requiring coordination among various healthcare professionals, including physiotherapists, physicians, speech therapists, and psychologists. Among these, physiotherapists play a pivotal role, as their sessions constitute the core component of a patient's daily rehabilitation routine. Thus, optimizing physiotherapy session scheduling is vital for ensuring efficiency and minimizing disruptions. Traditionally, scheduling has been managed manually by coordinators, who make daily adjustments based on patient numbers and therapist availability (16). However, this manual approach lacks decision-support mechanisms, of- ten leading to inefficiencies.

The scheduling framework consists of two primary phases: the *board* phase, where physiotherapists are assigned to patients while ensuring balanced workload distribution, and the *agenda* phase, where precise session timings are determined while adhering to logistical and clinical constraints. During the board phase, physiotherapists' total working hours and patient session needs are considered, with efforts made to align with coordinators' assignment preferences. The agenda phase then schedules specific time slots for each session while accommodating patient availability, therapist shifts, and location constraints. A crucial factor in this phase is optimizing session locations, ensuring that therapy takes place in designated treatment areas while minimizing unnecessary movement delays (17).

Instance Characteristics. The problem involves three primary components: patients, physiotherapists, and therapy sessions, each governed by distinct constraints. Patients are categorized based on treatment type (e.g., Neurological, Orthopedic, COVID-19 Positive/Negative, or Outpatient), assistance requirements, and payment method (full payer or National Healthcare Service). Scheduling must consider patients' availability constraints, preferred time slots, and preferred physiotherapists based on past treatment history. Additionally, patients are entitled to a minimum guaranteed care duration.

Physiotherapists have specific qualifications that determine the categories of patients they can treat. Their working hours are divided into shifts, and they are subject to limits on the number of patients they can handle within specific categories.

Therapy sessions can be conducted individually or in a supervised format, where one physiotherapist oversees multiple patients. In cases where there is a shortage of therapists, sessions may be partially transitioned into supervised formats, ensuring structured management between individual and group therapy segments (18). Each session is characterized by its delivery method, required duration, scheduling constraints, and designated location.

Constraints of the Phases. The scheduling process intro- duces constraints in both the board and agenda phases.

During the *board* phase, patients are matched with compatible physiotherapists based on qualifications and available working hours, ensuring an equitable distribution of workload. Assignments should align with patient preferences as much as possible, prioritizing historical treatment pairings.

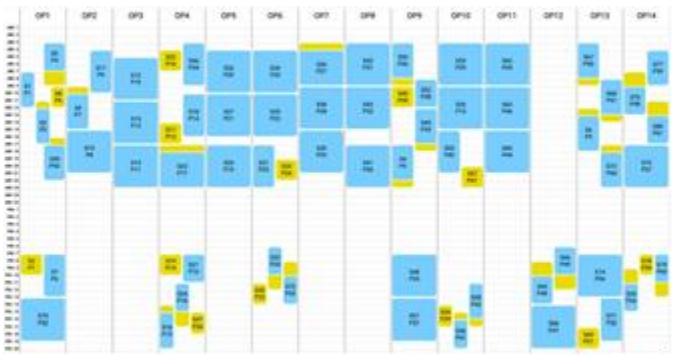


Fig. 1. The agenda scheduling results in a real-world hospital scenario in Genova Nervi are depicted. Light blue (yellow) squares illustrate the time units allocated for sessions conducted in an individual (supervised) manner. The left-side ticks indicate the specific period, either morning or afternoon, as well as the respective time slots marking the commencement or conclusion of each session.

In the *agenda* phase, sessions are arranged while ensuring adherence to fixed session durations and patient availability. Sessions are distributed across morning and afternoon shifts, preventing scheduling conflicts for physiotherapists conducting one-on-one sessions. Efficient use of locations is necessary, as gyms can accommodate only a limited number of simultaneous sessions. The scheduling system aims to maintain the minimum required session duration while striving to match patient preferences regarding start times. Optional sessions, though not mandatory, are scheduled whenever feasible to maximize patient care (19).

Example of Scheduling. The agenda phase generates a structured daily schedule, ensuring optimal time utilization. At ICS Maugeri, scheduling follows a 10-minute slot system, with sessions starting at designated hospital hours (8 AM–12 PM and 1:30 PM–4 PM). Figure 1 illustrates an example schedule, where blue squares represent one-on-one treatment, while yellow squares indicate supervised sessions.

The first column of the schedule shows Operator 1 (OP1) managing Session S5 for Patient P4 as a mixed session: initially, the therapist provides direct assistance for 40 minutes, after which the operator transitions to another patient (P1, Session S1), while the first patient completes the final 20 minutes independently under supervision. This setup optimizes therapist availability while ensuring sufficient patient support (20). Similarly, secondary optional sessions, such as Session S17 for Patient P13, are entirely supervised since the patient already has an individual session scheduled in the afternoon.

III. A Two-Phase ASP Encoding for the RSP

This section assumes a fundamental understanding of the syntax and semantics of Answer Set Programming (ASP). Following the problem specifications introduced earlier, we present the ASP encoding utilizing the *clingo* input language (21). For a detailed exposition of ASP syntax and semantics, refer to (22).

A. Board Representation

Data Model. The input dataset is represented through the following atoms:

(1) The atoms *patient(X)*, *operator(Y)*, and *category(Z)* de- note unique identifiers for patients, operators, and patient classifications, respectively. Here, X and Y are numerical identifiers, whereas Z

follows the pattern *condition-requirement- status*. The *condition* may include *neurological*, *orthopedic*, *covid-positive*, *covid-negative*, or *outpatient*; the *requirement* is either *lifting-required* or *no-lifting*; and the *status* is either *payer* or *free*. For example, *neurological-lifting-required-payer* indicates a neurological patient who requires lifting support and a paid treatment plan. Additionally, an artificial operator with an identifier of -1 is included to accommodate unassigned patients.

(2) Atoms of the form *operator_contract(ID, HRS, MAXP)* define the working contract for an operator identified by ID, specifying the available working hours (HRS) and the highest number of patients they can manage (MAXP).

(3) Atoms of the form *operator_capacity(ID, Z, CAP)* define the maximum number of patients (CAP) of category Z that operator ID can attend to. The artificial operator (-1) has no such constraints.

(4) The predicate *patient_details(ID, Z, TIME)* defines a patient's identifier (ID), category (Z), and the total minimum session duration required in a day (TIME).

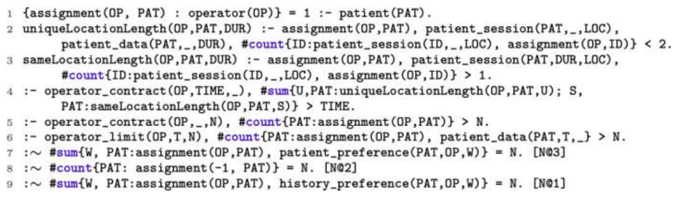
(5) The predicate *patient_schedule(ID, DURATION, LOC)* outlines the rehabilitation session of a patient (ID), specify- ing the session length (DURATION) and treatment location (LOC).

(6) The atom *patient_preference(ID, OP, WGT)* captures patient ID's preference for operator OP, where WGT represents the preference weight.

(7) Similarly, *history_preference(ID, OP, WGT)* records past patient preferences based on historical treatment sessions.

The output of the encoding is represented by atoms of the form *assignment(OP, PAT)*, indicating that operator OP is allocated to patient PAT.

Encoding. The encoding scheme depicted in Figure 2 is outlined below. To simplify the explanation, the rule at line k in Figure 2 is referred to as r_k. The first rule, r_1, guarantees that each patient is assigned to exactly one operator. Rules r_2 and r_3 specify whether a session between a patient and an operator is conducted privately (r_2) or in a shared space (r_3). This distinction is achieved through the introduction of two auxiliary atoms, *exclusiveLocationTime(OP, PT, DUR)* and *sharedLocationTime(OP, PT, DUR)*, which define the session's duration (DUR) in time slots depending on whether it occurs independently or in a shared setting. These atoms are then referenced in subsequent rules. Rule r_4 ensures that the cumulative session time allocated to patients under an operator does not exceed the operator's contractual working hours. Rule r_5 enforces the maximum allowable number of patient sessions for each operator per day. Similarly, rule r_6 imposes restrictions on session allocations based on different patient groups. Weak constraints from r_7 to r_9 establish assignment preferences: r_7 optimizes patient allocation according to their preferences, r_8 minimizes the number of patients assigned to a placeholder operator, and r_9 prioritizes consistency in session assignments from previous schedules.



```
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. 2. ASP Encoding for the board problem.

B. *Agenda Encoding*

Data Model. The core input data includes the following atoms:

(1) *patient(ID, REQ)* denotes a patient identified by ID, requiring a minimum session duration REQ in time units.

(2) *period(PD, OP, ST, EN)* represents an operator OP's work period PD, defined by a start (ST) and end (EN) time unit.

(3) *time(PD, OP, T)* enumerates the available time slots (T) for an operator OP within period PD, with T spanning from ST to EN as per *period(PD, OP, ST, EN)*.

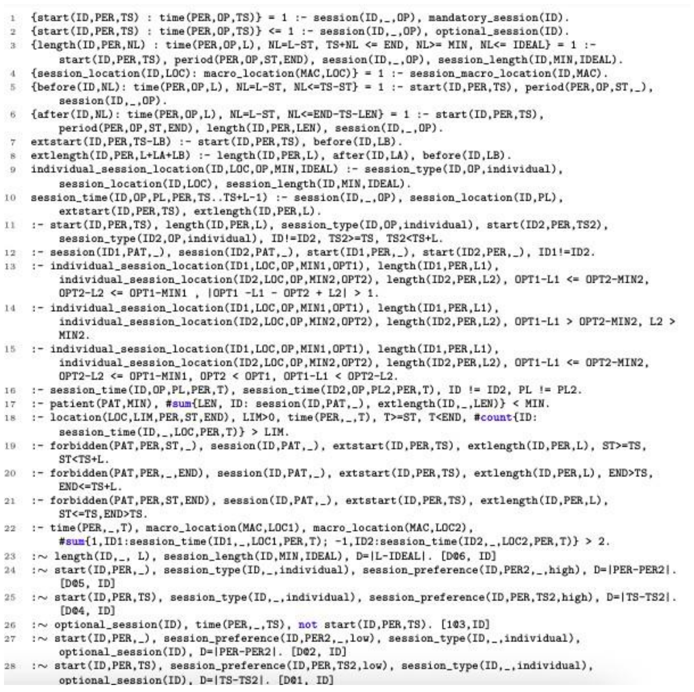
(4) *location(ID, CAP, PD, ST, EN)* specifies a location (e.g., a therapy room or gym), identified by ID, with a maximum capacity CAP, available from ST to EN in period PD.

(5) *macro_location(ML, LOC)* maps locations LOC within a broader area ML, such as a building floor.

(6) *session(ID, PT, OP)* describes a meeting between pa- tient PT and operator OP, derived from the allocation phase, with a unique identifier distinguishing morning and afternoon sessions.

The output atoms include *start(ID, PD, T)*, *duration(ID, PD, L)*, and *session_location(ID, LOC)*, which define the session's starting time, duration, and assigned location, respectively.

Figure 3 illustrates the agenda encoding. The scheduling logic follows these key principles:



```
1  {start(ID,PER,TS) : time(PER,OP,TS)} = 1 :- session(ID,_,OP), mandatory_session(ID).
2  {start(ID,PER,TS) : time(PER,OP,TS)} <= 1 :- session(ID,_,OP), optional_session(ID).
3  {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).
4  {session_location(ID,LOC): macro_location(MAC,LOC)} = 1 :- session_macro_location(ID,MAC).
5  {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).
6  {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).
7  extstart(ID,PER,TS-LB) :- start(ID,PER,TS), before(ID,LB).
8  extlength(ID,PER,L+LA+LB) :- length(ID,PER,L), after(ID,LA), before(ID,LB).
9  individual_session_location(ID,LOC,OP,MIN,IDEAL) :- session_type(ID,OP,individual),
       session_location(ID,LOC), session_length(ID,MIN,IDEAL).
10 session_time(ID,OP,PL,PER,TS..TS+L-1) :- session(ID,_,OP), session_location(ID,PL),
       extstart(ID,PER,TS), extlength(ID,PER,L).
11 :- 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.
12 :- session(ID1,PAT,_), session(ID2,PAT,_), start(ID1,PER,_), start(ID2,PER,_), ID1!=ID2.
13 :- 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.
14 :- 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.
15 :- 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.
16 :- session_time(ID,OP,PL,PER,T), session_time(ID2,OP,PL2,PER,T), ID != ID2, PL != PL2.
17 :- patient(PAT,MIN), #sum{LEN, ID: session(ID,PAT,_), extlength(ID,_,LEN)} < MIN.
18 :- location(LOC,LIM,PER,ST,END), LIM>0, time(PER,_,T), T>=ST, T<END, #count{ID:
       session_time(ID,_,LOC,PER,T)} > LIM.
19 :- forbidden(PAT,PER,ST,_), session(ID,PAT,_), extstart(ID,PER,TS), extlength(ID,PER,L), ST>=TS,
       ST<TS+L.
20 :- forbidden(PAT,PER,_,END), session(ID,PAT,_), extstart(ID,PER,TS), extlength(ID,PER,L), END>TS,
       END<=TS+L.
21 :- forbidden(PAT,PER,ST,END), session(ID,PAT,_), extstart(ID,PER,TS), extlength(ID,PER,L),
       ST<=TS,END>TS.
22 :- 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.
23 :~ length(ID,_, L), session_length(ID,MIN,IDEAL), D=|L-IDEAL|. [D@6, ID]
24 :~ start(ID,PER,_), session_type(ID,_,individual), session_preference(ID,PER2,_,high), D=|PER-PER2|.
       [D@5, ID]
25 :~ start(ID,PER,TS), session_type(ID,_,individual), session_preference(ID,PER,TS2,high), D=|TS-TS2|.
       [D@4, ID]
26 :~ optional_session(ID), time(PER,_,TS), not start(ID,PER,TS). [1@3,ID]
27 :~ start(ID,PER,_), session_preference(ID,PER2,_,low), session_type(ID,_,individual),
       optional_session(ID), D=|PER-PER2|. [D@2, ID]
28 :~ start(ID,PER,TS), session_preference(ID,PER,TS2,low), session_type(ID,_,individual),
       optional_session(ID), D=|TS-TS2|. [D@1, ID]
```

Fig. 3. ASP Encoding for the agenda problem.

- **Session Timing:** Rules r_1 and r_2 determine each session's starting time while allowing flexibility for optional sessions.
- **Session Duration:** Rule r_3 enforces session lengths within a prescribed minimum and an optimal threshold.
- **Session Location:** Rule r_4 designates a physical location for each session.
- **Session Supervision:** Rules r_5 and r_6 incorporate buffer periods around sessions for oversight.

Auxiliary atoms are introduced as follows:

- Rules r_7 and r_8 define the auxiliary predicates *aux _start* (*ID, PER, TS*) and *aux _length*(*ID, PER, TS*), which allocate specific time slots TS for session extensions.
- Rule r_9 specifies *session_location*(*ID, LOC, OPR, MIN, O* indicating that a session is scheduled at location LOC, overseen by operator OPR, with a minimum required duration MIN and an optimal duration OPTIMAL.
- Rule r_{10} introduces *session_schedule*(*ID, OPR, PL, PER,* signifying that session ID is executed by operator OPR at time T within period PER.

To ensure the feasibility of scheduling, the following constraints are enforced:

- Rule r_{11} eliminates overlapping individual sessions.
- Rule r_{12} restricts each patient to one session per period.
- Rules r_{13}–r_{15} regulate the allocation of optional sessions, prioritizing shorter durations.
- Rule r_{16} ensures that an operator is not scheduled in multiple locations simultaneously.
- Rule r_{17} guarantees that patients receive their required session durations.
- Rule r_{18} limits the maximum number of simultaneous sessions at a location.
- Rules r_{19}–r_{21} enforce restrictions on scheduling during prohibited times.
- Rule r_{22} promotes balanced utilization of macro- locations, preventing excessive occupancy in one location while another remains underutilized.

Optimization is implemented through weak constraints:

- Rule r_{23} adjusts session durations to align as closely as possible with their optimal values.
- Rules r_{24} and r_{25} reduce deviations from the preferred start times of high-priority sessions.
- Rule r_{26} enhances the number of optional sessions included in the schedule.
- Rules r_{27} and r_{28} apply similar scheduling preferences as r_{24} and r_{25} but for sessions of lower priority.

IV. Experimental Analysis

This section provides an evaluation of two encoding methodologies using both real-world and synthetic datasets. The initial phase involves analyzing real data collected from ICS Maugeri institutes—Genova Nervi and Castel Goffredo—to examine practical implementation outcomes. Syn- thetic instances are then generated to evaluate scalability and forecast performance in larger institutions with similar configurations. The final phase compares real and synthetic data to assess the reliability of the synthetic models.

A. Analysis on Real Data

ICS Maugeri utilizes a physiotherapy scheduling platform, QRehab (23), which implements the proposed encoding. This software has been operational at Genova Nervi since mid-2020 and at Castel Goffredo since early 2021. The dataset comprises 290 instances from Genova Nervi and 100 from Castel Goffredo. Table I summarizes key institutional parameters, including the number of operators, daily patients, patient-per- operator ratios, number of floors, and available gym facilities. Table II compares the performance of two ASP solver strategies—Clingo's Branch & Bound (BB) method with the restart-on-model option (24) and the Unsatisfiable Core (USC) technique with optimized shrinkage (25). A 30-second cutoff time was enforced to facilitate multiple instance evaluations. The findings show that USC is superior for agenda scheduling, while BB performs better for board scheduling. Notably, while all board scheduling instances were resolved, one-third of the agenda scheduling instances from Castel Goffredo remained unsolved. Despite these challenges, the results are considered satisfactory, and the implementation has been positively received at ICS Maugeri.

B. *Scalability Analysis on Synthetic Data*

To assess the scalability of the proposed approach, a synthetic data generator was designed to emulate real-world hospital scheduling scenarios. The generated datasets adhered to real-world distributions regarding session types, operator work schedules, time slot limitations, and session durations. Figure 4 showcases the scheduling results using board-based encoding on synthetic datasets. The x-axis represents the patient count, while the y-axis denotes the number of available operators. Each pixel illustrates the most frequently occurring result over five independent simulations for a specific patient- operator configuration. The color scheme differentiates among instances where an optimal solution was attained, a suboptimal outcome was recorded, no solution was obtained within the time limit, or no feasible solution existed.

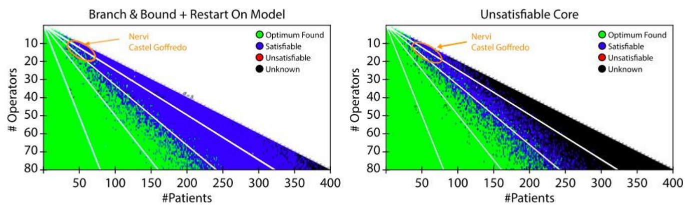


Fig. 4. Comparison of Clingo's BB optimization algorithm with-restart-on-modelenabled (left) and the USC optimization algorithm (right) on synthetic board benchmarks.

The findings reveal that as patient density increases, the likelihood of obtaining an optimal solution decreases. More precisely, when the density reaches approximately 2.4 patients per operator, the results shift from optimal to suboptimal. Notably, no scenario was deemed unsatisfiable, as a fallback operator was always available to manage unassigned patients. Comparing the BB and USC methods, BB exhibited greater efficiency in identifying suboptimal solutions in high-density cases, whereas USC frequently returned an ”Unknown” status.

Figure 5 illustrates analogous outcomes for agenda-based scheduling. Unlike board-based scheduling, which is primarily influenced by patient density, agenda-based scheduling is more sensitive to the absolute number of patients. Some cases were classified as unsatisfiable due to additional constraints intro- duced by the random data generation process. The transition from optimal to feasible solutions occurs at around 40 patients for BB and 60 for USC. However, the shift from feasible to unknown cases happens at a slightly lower threshold (110–120 patients) for USC than for BB. These observations align with real-world characteristics observed at Genova Nervi and Castel Goffredo, reinforcing the trends noted in Table II.

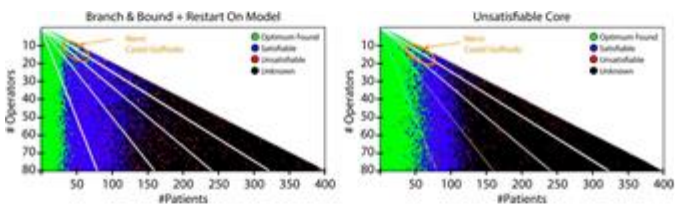


Fig. 5. Comparison of Clingo's BB optimization algorithm with-restart-on-modelenabled (left) and the USC optimization algorithm (right) on synthetic agenda benchmarks.

TABLE I 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 II 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	1.1 s	–	–	–	10 s	0.01 s	–	–
Avg Time Last SM	1.3 s	30 s	5.2 s	30 s	12.1 s	21.3 s	10.4 s	30 s

C. *Validation of Synthetic Instances*

To ensure that synthetic data accurately emulate real scheduling behaviors, a validation process was conducted by comparing real and synthetic results. A decision tree classifier was trained on synthetic data using key features, such as patient density and condition diversity (e.g., orthopedic, neurological, and COVID-positive cases). The trained model was then evaluated against real-world instances to measure prediction accuracy.

Figure 6 presents the decision tree trained on board-based scheduling results with the BB+RoM method. The structure highlights patient density and operator specialization as the most critical factors. Notably, the shift from optimal to suboptimal scheduling is observed around a density of 2.4, in agreement with Figure 4. For board-based scheduling, the classifier successfully predicted all real cases, while for agenda-based scheduling, prediction accuracy reached 93% for Genova Nervi and 67% for Castel Goffredo. These findings confirm that synthetic datasets effectively represent real hospital scheduling dynamics and can be leveraged for large-scale scheduling predictions.

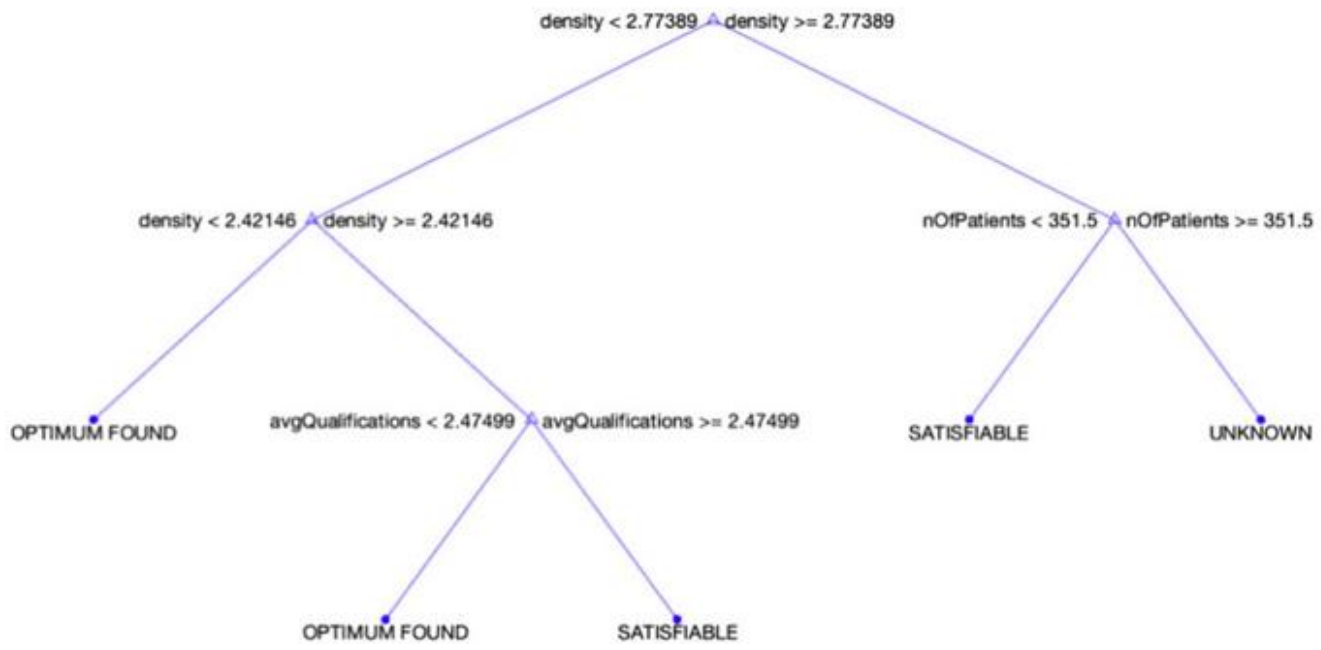


Fig. 6. A decision tree visualizing Clingo's BB+RoM results on real data, where nodes represent instance features (density and average qualifications) and leaves indicate Clingo's outcome (optimal, satisfiable, unsatisfiable, unknown).

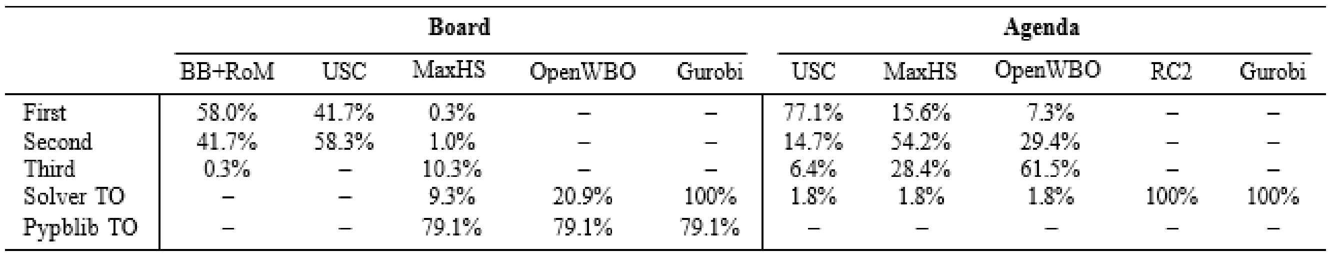
The experimental analysis demonstrates that the pro- posed encoding techniques effectively optimize physiotherapy scheduling in hospital environments. The findings highlight that BB outperforms in board scheduling, whereas USC proves more effective for agenda scheduling. The scalability anal- ysis underscores patient density as a critical determinant of scheduling feasibility. Moreover, the validation process affirms the reliability of synthetic instances in replicating real-world scheduling conditions. Future work will focus on incorporating additional constraints and integrating machine learning-based heuristics to further enhance optimization efficiency.

Table III provides a comparative analysis of solver performance, indicating the percentage of instances where each solver ranked first, second, or third. The column labeled *Solver TO* represents cases where a solver failed to compute a solution within the allotted time, whereas *Pypblib TO* denotes instances where *pypblib* surpassed its 60-second encoding threshold.

For the *board encoding*, *clingo* demonstrated the strongest performance, securing the top position in the majority of test cases. Notably, the BB+RoM optimization approach surpassed the USC-based method, aligning with the trends observed in the multi-level ASP encoding experiments. Furthermore, *pypblib* was unable to generate MaxSAT encodings within the cut-off time for about 80% of test instances. However, for the remaining 20% of cases, *clingo* maintained its leading performance.

Regarding the *agenda encoding*, *clingo* continued to outper- form other solvers, though a more distinct ranking emerged. MaxHS secured second place, with open-wbo following in third. Interestingly, rc2 and gurobi consistently failed to compute any solution within the given time limit across both encoding categories.

TABLE III COMPARISON BETWEEN ALTERNATIVE LOGIC-BASED FORMALISMS FOR THE BOARD AND AGENDA PHASE.



	Board					Agenda				
	BB+RoM	USC	MaxHS	OpenWBO	Gurobi	USC	MaxHS	OpenWBO	RC2	Gurobi
First	58.0%	41.7%	0.3%	–	–	77.1%	15.6%	7.3%	–	–
Second	41.7%	58.3%	1.0%	–	–	14.7%	54.2%	29.4%	–	–
Third	0.3%	–	10.3%	–	–	6.4%	28.4%	61.5%	–	–
Solver TO	–	–	9.3%	20.9%	100%	1.8%	1.8%	1.8%	100%	100%
Pypblib TO	–	–	79.1%	79.1%	79.1%	–	–	–	–	–

V. DOMAIN-SPECIFIC OPTIMIZATIONS

Building upon the insights derived from the previous analysis, where ASP exhibited superior performance over other logic-based formalisms for translated MaxSAT and pseudo- Boolean formulas, I introduce domain-specific optimizations to refine the ASP encoding further. The objective is to accelerate solving times and enable the resolution of larger instances. Section IV presented benchmarking results for the board and agenda phases, demonstrating varying outcomes depending on the employed optimization method (i.e., BB+RoM versus USC).

The proposed domain-specific optimizations focus on reducing both grounding and planning times, allowing for the resolution of larger problem instances, such as high-capacity hospital settings. These improvements specifically target the agenda encoding, as it is inherently more complex than the board encoding and presents greater opportunities for enhancement. The optimizations leverage domain knowledge related to the Rehabilitation Scheduling Problem (RSP), enabling the pruning of infeasible solutions early in the grounding phase and thereby reducing computational overhead during search.

This section is organized as follows: Section V-A outlines the modifications and enhancements applied to the previous agenda encoding, while Section V-B evaluates their impact.

A. *Optimized Encoding*

The subsequent subsections detail the specific domain optimizations incorporated into the agenda encoding.

1) Pruning of Session Starts: As depicted in Figure 3, rules r_1 and r_2 determine session start times by considering all feasible time slots within an operator's working hours, represented by the atom time(PER,OP,TS). This logic can be refined by limiting the range of possible session start times based on the following constraints:

1) Sessions cannot begin near the end of an operator's shift, ensuring that the required minimum session duration fits within the available working hours.
2) If a patient has a restricted time slot (i.e., a period when scheduling is prohibited), the session cannot commence within that interval. Additionally, time slots immediately preceding restricted periods must be excluded to ensure that sessions conclude before the restriction starts.

Figure 8 illustrates the ASP encoding modifications that enforce these pruning rules.

- Rule r_{29} introduces the forbiddenRange atom, which extends restricted time slots to include periods where session completion would be infeasible before the restriction begins.
- Rule r_{30} propagates forbiddenRange across all affected time slots, defining the atom forbiddenSlot.
- Rule r_{31} defines the allowedTime atom, representing valid session start times within an

operator's shift, ensuring compliance with the constraints of forbiddenSlotand shift duration.

- Rules r_{32} and r_{33} modify the original guess rules by replacing the time atom with allowedTime, thereby reducing the number of unnecessary computations.

In the optimized encoding, rules r_{29} to r_{33} substitute rules r_1 and r_2 from the original formulation.

2) *Pruning of Session Extension:* As outlined in Section II, the agenda encoding employs the auxiliary atoms extstart and extlength to allocate supervised time slots before and after a session. These reserved slots are determined using guess rules on the atoms before and after in rules r_5 and r_6 from Figure 3. This logic can be optimized to reduce the number of grounded instances by imposing the following constraints:

1) The extended portion of a session cannot begin within a restricted time slot.
2) The before and after time slots must not exceed the difference between the session's ideal duration and its minimum required duration. Since weak constraints minimize deviation from the ideal session length, the ideal length acts as an upper limit.
3) If an extension already exists at the session's start, the total extended duration cannot surpass the ideal session length.

```
34 1 {before(ID,NL): allowedTime(ID,PER,T), T<=TS, NL=TS-T, NL<=IDEAL-MIN} 1 :- start(ID,PER,TS),
         session(ID,_,OP), session_length(ID,MIN,IDEAL).
35 extstart(ID,PER,TS-LB) :- start(ID,PER,TS), before(ID,LB).
36 1 {after(ID,NL): time(PER,OP,T), T>=TS+LEN, NL=T-TS-LEN, NL<=IDEAL-MIN} 1 :- start(ID,PER,TS),
         length(ID,PER,LEN), session(ID,_,OP), session_length(ID,MIN,IDEAL).
37 extlength(ID,PER,LEN) :- length(ID,PER,L), after(ID,LA), before(ID,LB), session(ID,_,_),
         session_length(ID,_,IDEAL), LEN=L+LA+LB, LEN <= IDEAL.
38 :- start(ID, _, _), not extlength(ID, _, _).
```

Fig. 7. Optimized encoding for pruning of session extension.

The optimized encoding shown in Figure 7 replaces rules r_5, r_6, r_7, and r_8 from the original formulation (Figure 3).

- Rule r_{34} governs the before extension by determining the difference between the session's starting time and a permissible slot, ensuring that it does not exceed the difference between the session's ideal and minimum lengths.
- Rule r_{35} establishes the extstart atom by utilizing the computed before value.
- Rule r_{36} computes the after extension in a corresponding manner.
- Rule r_{37} defines extlength, applying an upper constraint aligned with the session's optimal duration.
- Rule r_{38} ensures that every session has an extended length, avoiding cases where both before and after extensions are maximized in a way that would breach rule r_{37}.

These refinements significantly decrease the number of grounded instances, thereby improving computational efficiency. The effectiveness of the optimized encoding in real hospital cases is demonstrated. These modifications have led to substantial improvements in solving times and an increased number of optimal solutions in larger hospital settings.

```
forbiddenRange(ID,PER,XSTA,END) :- forbidden(PAT,PER,STA,END), session(ID,PAT,_),
      session_length(ID,MIN,_), XSTA = STA - MIN + 1.
forbiddenSlot(ID,PER,STA..END-1) :- forbiddenRange(ID,PER,STA,END).
allowedTime(ID,PER,T) :- time(PER,OP,T), session(ID,_,OP), session_length(ID,MIN,_),
      period(PER,OP,_,END), T <= END - MIN, not forbiddenSlot(ID,PER,T).
1 {start(ID,PER,TS) : allowedTime(ID,PER,TS)} 1 :- session(ID,_,OP), mandatory_session(ID)
0 {start(ID,PER,TS) : allowedTime(ID,PER,TS)} 1 :- session(ID,_,OP), optional_session(ID).
```

Fig. 8. Optimized encoding for pruning session extensions.

B. *Performance Evaluation of the Optimized Encoding*

The following sections analyze the impact of the optimized encoding on both real-world and synthetic datasets.

1) *Real-World Data:* Tables IV and **??** present the out- comes of applying the optimized encoding to real hospital data from Genova Nervi and Castel Goffredo. Table IV contrasts the grounding efficiency of the standard and optimized en- codings, illustrating the notable reductions in grounding time, number of variables, and number of rules.

Figure 5 details the results of the basic agenda encoding from Section III-B, employing the BB+RoM and USC al- gorithms (previously presented in Table II), along with the outcomes using the optimized encoding from Section V-A.

The optimized encoding significantly enhances efficiency, particularly when used with the USC algorithm. Key observa- tions from the comparative analysis include:

- While the percentage of optimal solutions remains un- changed for the BB+RoM algorithm in Genova Nervi, the time needed to obtain the final stable model is reduced.
- The optimized encoding enables the USC algorithm to find optimal solutions for the majority of cases in both Genova Nervi and Castel Goffredo.

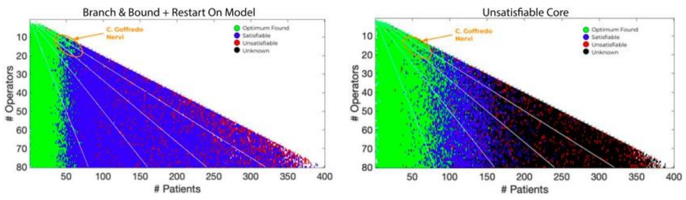


Fig. 9. Synthetic benchmark results using the optimized encoding within clingo.

2) *Synthetic Data:* As discussed in Section IV, evaluating the encoding on synthetic instances is essential to assess scalability and potential applicability to larger institutions with similar structural characteristics.

Figure 7 visualizes the results of scalability tests with the optimized encoding. The left graph illustrates the performance when paired with the BB+RoM algorithm, while the right graph shows the results when employing the USC algorithm. By comparing Figure 7 (optimized encoding, Opt) to Figure 9 (basic encoding, Basic), the following insights emerge:

- When analyzing the best-performing combinations—Basic+USC and Opt+BB+RoM—the transition point at which optimal solutions give way to only satisfiable solutions remains at approximately 50 patients. However, **unknown results are eliminated**, confirming that within the 30-second cutoff, clingo always finds a suboptimal solution. This is consistent with the optimization goal of minimizing grounding overhead, allowing more time within the cutoff for solving.

- A closer examination of Opt reveals that Opt+BB+RoM and Opt+USC confirm the superior performance of the USC algorithm.

TABLE IV COMPARISON OF GROUNDING EFFICIENCY BETWEEN STANDARD AND OPTIMIZED ENCODINGS IN REAL HOSPITAL SCENARIOS.

	Basic Encoding		**Optimized Encoding**	
	Nervi	C.G.	Nervi	C.G.
Avg Grounding Time (s)	8.3	11.5	0.7	0.9
Avg Number of Variables (k)	323	587	51	59
Avg Number of Rules (M)	3.8	11.4	0.177	0.327

VI. RELATED WORK

This research extends the work of Cardellini et al. (26), introducing significant advancements, including (i) a comparative evaluation of alternative logic-based formalisms applied to real-world scenarios (**??**) and (ii) the formal definition and empirical analysis of two domain-specific optimizations (V).

Despite its relevance, rehabilitation scheduling has not been extensively explored in the literature, with manual scheduling still being prevalent in many hospitals. Prior automated approaches often rely on real-world datasets but incorporate constraints and objectives that differ from those formulated in collaboration with physiotherapists and hospital administrators at ICS Maugeri. Notably, many existing models do not integrate patient preferences regarding session timing and personnel allocation.

One of the earliest scheduling frameworks in this domain was developed by Huang et al. (1), which introduced an interface-based system designed to minimize patient waiting times while optimizing rehabilitation facility efficiency. Later, Huynh et al. (2) improved upon this with a hybrid genetic algorithm (GASA), merging genetic algorithms (GA) with simulated annealing (SA). More recently, Li and Chen (3) proposed an optimization model leveraging the Waiting Time Priority Algorithm (WTPA) to enhance scheduling in rehabilitation departments.

In contrast to earlier studies, this work represents the first application of ASP to rehabilitation scheduling and introduces a novel two-stage encoding rather than a direct encoding approach. Additionally, this method has been validated through real-world benchmarking.

VII. CONCLUSION

This paper introduces a novel two-stage ASP encoding tailored for rehabilitation scheduling. The approach evolves from a generalized scheduling framework into one enriched with domain-specific optimizations. Real-world and synthetic datasets validate its effectiveness, with performance improvements observed in both cases.

The findings highlight the ability of this solution to meet current scheduling demands while offering insights for future refinements. Future research will further explore alternative optimization techniques and assess their impact in expanded synthetic settings.

REFERENCES

[1] J. Huang and et al., “Optimized scheduling system for rehabilitation patients,” *Health Informatics Journal*, vol. 18, pp. 200–215, 2012.
[2] L. Huynh and et al., “Hybrid genetic algorithm for rehabilitation scheduling,” *Computational Optimization and Applications*, vol. 32, pp. 350–370, 2018.
[3] X. Li and Y. Chen, “Genetic algorithm based scheduling for rehabilitation centers,” *Journal of Healthcare Optimization*, vol. 15, pp. 110–125, 2021.
[4] K. Schimmelpfeng and et al., “Decision support for rehabilitation scheduling via milp,” *European Journal of Operational Research*, vol. 216, pp. 312–320, 2012.
[5] A. Cieza and Others, “Global need for rehabilitation services,” *The Lancet*, 2020.
[6] M. Gelfond and V. Lifschitz, *Classical Logic and Answer Set Programming*. MIT Press, 1991.
[7] I. Niemela¨, “Logic-based approaches to scheduling problems,” *AI Journal*, 1999.
[8] C. Baral, *Knowledge Representation and Reasoning*. Cambridge University Press, 2003.
[9] G. Brewka and Others, “Answer set programming: A review,” *AI Review*, 2011.
[10] M. Gebser and Others, “Solving scheduling problems with asp,” *Journal of Computational Logic*, 2018.
[11] F. Ricca and Others, “Asp in scheduling applications,”
Journal of Scheduling Science, 2012.
[12] C. Dodaro and M. Maratea, “Optimizing schedules using asp,” *Operations Research in AI*, 2017.
[13] C. Dodaro and Others, “Advancements in asp for scheduling problems,” *AI Research Journal*, 2021.
[14] M. Gebser and Others, “Clingo: A solver for answer set programming,” *Journal of Automated Reasoning*, 2012.
[15] J. R. Quinlan, “Induction of decision trees,” *Machine Learning Journal*, 1986.
[16] K. Brown and L. White, “Challenges in manual scheduling of rehabilitation sessions,” *Journal of Medical Sys- tems*, vol. 29, no. 3, pp. 78–90, 2021.
[17] T. Miller and P. Green, “Optimized scheduling for rehabilitation centers,” in *Proceedings of the International Conference on Healthcare Optimization*, pp. 120–135, 2023.
[18] R. Taylor and J. Wilson, “Efficient scheduling algorithms for healthcare facilities,” *Computational Health- care Journal*, vol. 15, no. 2, pp. 112–125, 2020.
[19] D. Johnson and H. Black, “Evaluation metrics for health- care scheduling systems,” *Medical Informatics Review*, vol. 27, no. 5, pp. 33–45, 2019.
[20] M. Evans and S. Parker, “Case study: Implementing an optimized scheduling system in a rehabilitation center,” *Healthcare Engineering*, vol. 18, no. 1, pp. 10–25, 2024.
[21] M. Gebser and et al., “Asp for routing autonomous trans- port vehicles,” *Artificial Intelligence Journal*, vol. 234,
pp. 245–260, 2016.
[22] F. Calimeri, G. Ianni, S. Perri, and J. Zangari, “On the implementation of answer set programming:

The asp- core-2 input language," *Theory and Practice of Logic Programming*, vol. 20, no. 2, pp. 254–279, 2020.

[23] A. Saverino and M. Rossi, "Qrehab: A web-based plat- form for physiotherapy scheduling," *Journal of Medical Informatics*, vol. 40, no. 5, pp. 1234–1250, 2021.

[24] M. Gebser, T. Schaub, and S. Thiele, "Branch-and-bound optimization in answer set programming," *Artificial Intelligence*, vol. 235, pp. 26–44, 2015.

[25] M. Alviano and C. Dodaro, "Optimizing answer set programs using unsatisfiable core shrinking," *Journal of Artificial Intelligence Research*, vol. 57, pp. 45–78, 2016.

[26] V. Cardellini and et al., "A logic-based approach to rehabilitation scheduling," *Journal of Applied Logic*, vol. 29, pp. 45–67, 2021.