Bridging Structure and Search: A Hybrid GNN-Constrained Optimization Approach for Real-World University Timetabling

Abstract

Efficient and equitable university course timetabling is a foundational component of higher education systems, with direct implications for student access, instructional quality, and optimal use of institutional resources. However, generating high-quality timetables remains a computationally challenging problem, requiring the assignment of time slots and rooms to courses while satisfying complex pedagogical and administrative constraints. To encourage practical solutions to this problem, the International Timetabling Competition (ITC 2019) was launched, providing real-world benchmark datasets to advance research in this domain. However, despite the availability of benchmark datasets and the growing interest in AI-based methods, efficiently generating high-quality solutions, particularly for large-scale timetabling instances, remains a significant open challenge. While deep learning techniques have shown promise in capturing structural regularities within scheduling data, purely neural approaches often fall short when confronted with the intricate combinatorial constraints that define real-world timetabling problems.

In this paper, we propose a novel hybrid AI approach that integrates Graph Neural Networks (GNNs) with a state-of-the-art Constraint Programming solver (CP-SAT) to improve solution quality and scalability for large-scale timetabling. Our method first trains a GNN on small, optimally solved subinstances to learn structural patterns of feasible schedules. These learned representations are then used to generate informative *hints* that guide the CP-SAT solver in solving full-scale problems. Experimental results show that our approach consistently outperforms the standalone CP-SAT solver in both solution quality and runtime. On the *pu-d9-fal19-late* instance, our method reduced the penalty score from 196,026 to 162,796 and improved computational efficiency from 3,204 to 2,981 seconds. Our results show how AI-driven timetabling can enhance access and equity in education by generating higher-quality schedules that better accommodate institutional constraints and diverse student needs.

Introduction

University course timetabling is a foundational yet highly complex problem in educational systems worldwide. It involves assigning courses, instructors, and rooms to specific time slots while satisfying a wide range of logistical, pedagogical, and institutional constraints. High-quality timetables are essential not only for ensuring smooth academic operations but also for promoting equitable access to education, enabling students to enroll in required courses, avoid scheduling conflicts, and graduate on time. As class sizes grow, course offerings diversify, and resources remain limited, manual scheduling becomes infeasible and risks exacerbating existing inequities in access and course availability.

From a computational perspective, university timetabling is a classic NP-hard combinatorial optimization problem (Błazewicz et al. 1994; Lewis and Thompson 2015), with an astronomically large solution space and numerous hard and soft constraints. The International Timetabling Competition (ITC 2019) has provided real-world benchmark datasets that reflect the complexity of institutional scheduling needs (Müller, Rudová, and Müllerová 2025). Traditional approaches, such as constraint programming and heuristic search (Lewis 2008), have achieved notable successes in solving small to medium-sized instances, but often struggle with scalability and efficiency in large-scale settings.

Google's CP-SAT solver (Perron and Furnon 2019), a state-of-the-art constraint programming engine, can systematically search large solution spaces to produce feasible or near-optimal schedules. However, when applied to highly constrained university timetabling problems, CP-SAT alone can suffer from long runtimes and limited scalability (Ceschia, Di Gaspero, and Schaerf 2023) . To overcome these challenges, recent research has turned toward combining symbolic solvers with learning-based components that guide search more intelligently (Khalil et al. 2016; Cappart et al. 2023).

In this paper, we propose a hybrid AI framework that integrates Graph Neural Networks (GNNs) with CP-SAT to tackle complex university timetabling problems. Our method first trains a GNN on small, optimally solved subinstances to learn the structural characteristics of feasible schedules. The GNN then generates informative *hints*, predictions about promising assignments, which are fed into the CP-SAT solver to guide its search process. This learned guidance improves both the quality of generated schedules and the solver's runtime performance. Our experiments on the ITC 2019 benchmark datasets demonstrate that our

hybrid GNN+CP-SAT approach consistently outperforms standalone CP-SAT solvers. On the *pu-d9-fal19-late* instance, our method reduced the penalty score from 196,026 to 162,796 and achieved a faster runtime (2,981 seconds vs. 3,204). This hybridization opens new avenues for AI-assisted planning in high-stakes, real-world domains like education.

Beyond technical performance, our work highlights the potential of AI-driven timetabling systems to improve educational equity. Poorly constructed schedules can disproportionately affect students with rigid life constraints, such as those working part-time, commuting long distances, or with caregiving responsibilities. By producing higher-quality schedules, our method helps institutions better accommodate diverse student needs, supporting fairer and more inclusive access to education.

Related Work

University timetabling is a longstanding and extensively studied problem in combinatorial optimization, with diverse formulations and benchmark datasets developed over the years. Recent work by Ceschia et al. (Ceschia, Di Gaspero, and Schaerf 2023) provides a comprehensive taxonomy and analysis of educational timetabling problems, highlighting six standard formulations and the benchmark datasets that support them. Among these, the ITC-2019 formulation is considered one of the most realistic, combining course scheduling with student sectioning, and incorporating real-world institutional features such as heterogeneous class structures, dynamic weekly timetables, and complex distribution constraints.Unlike earlier benchmarks (e.g., PE-CTT (Lewis, Paechter, and McCollum 2007) and CB-CTT (Bonutti et al. 2012)), which are more simplified and less reflective of operational realities, ITC-2019 enables rigorous evaluation of algorithms under practical conditions. The survey also emphasizes the importance of reproducibility, standardized evaluation, and solution verification principles we adopt by testing our method on public ITC-2019 instances and reporting performance using established objective metrics. Our work contributes to this growing body of literature by proposing a hybrid GNN + CP-SAT framework that addresses the scalability challenges highlighted in the survey while remaining grounded in real-world institutional constraints.

Traditional optimization techniques, such as Constraint Programming (CP), Integer Programming, and metaheuristic approaches (e.g., Tabu Search, Simulated Annealing), have achieved significant success in solving small to medium-sized instances (Burke et al. 1997; Lewis 2008; Abdullah and Turabieh 2012; Abdullah et al. 2012). In particular, Google's CP-SAT solver has become a popular choice for solving large-scale discrete optimization problems (Perron and Furnon 2019). However, real-world instances of university timetabling, such as those found in ITC-2019, present significant scalability challenges for pure CP solvers due to their combinatorial complexity and large, heterogeneous constraint sets (Müller et al. 2018). While traditional approaches continue to dominate educational timetabling research, recent studies have begun to explore the integration

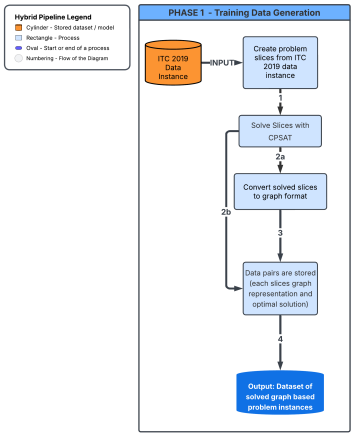

Figure 1: Training pipeline: Sub-problems extracted from the ITC 2019 dataset are solved using CP-SAT and encoded as graphs to generate supervised training data for the GNN.

of machine learning techniques to improve scalability and solution quality. In particular, graph-based representations and Graph Neural Networks (GNNs) have shown promise in capturing structural dependencies among scheduling entities and guiding optimization solvers more effectively (Cappart et al. 2023; Khalil et al. 2017; Gasse et al. 2019). This emerging class of ML-augmented optimization frameworks has demonstrated potential in improving solver efficiency and solution quality, particularly for large, constraint-rich scheduling problems.

Our work builds upon this line of research by proposing a novel hybrid approach that combines a GNN trained on small, optimally solved subinstances with a CP-SAT solver. By leveraging the GNN to generate informative search hints, our method addresses the scalability bottlenecks of classical solvers while preserving solution interpretability and constraint satisfaction guarantees. To the best of our knowledge, this is the first work to apply a GNN-based hint generation strategy within a CP-SAT framework for university course timetabling. We evaluate our approach on real-world university timetabling instances from ITC-2019, aligning with the community's call for reproducible, benchmark-driven comparisons (Ceschia, Di Gaspero, and Schaerf 2023).

Methodology

We propose a hybrid framework that integrates Graph Neural Networks (GNNs) with a constraint programming solver (CP-SAT) to address the scalability challenges of university course timetabling. The goal is to leverage the learning capacity of GNNs to guide the symbolic search process, thereby improving solution quality and efficiency on real-world instances. Our GNN-guided solver is benchmarked against two baselines: a standalone CP-SAT solver representing the state-of-the-art, and a standalone GNN. Our methodology consists of the following key components: (1) problem formalization, (2) dataset selection, (3) baseline models, and (4) hybrid solver pipeline. The development

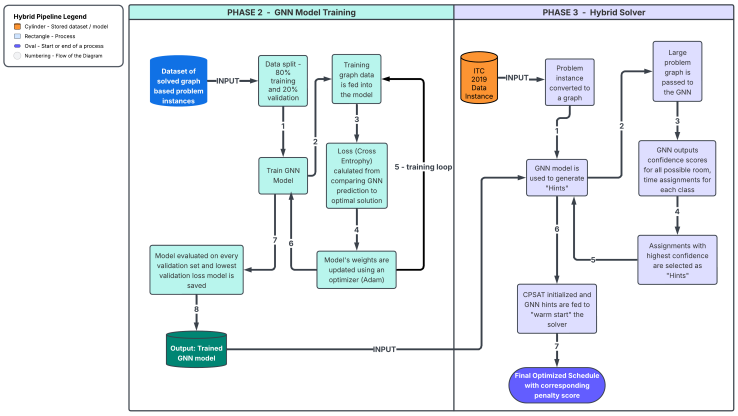

Figure 2: Hybrid solver pipeline: A pre-trained GNN generates high-confidence hints from an unsolved problem graph, which are used to warm-start the CP-SAT solver and improve schedule quality.

and evaluation of the hybrid model followed a structured and multi-phase pipeline as detailed below and illustrated in Figures 1 and 2.

Problem Formulation

Given a set of university courses, rooms, and available time slots, the task is to assign each course to a valid room-time pair while satisfying hard constraints and minimizing penalties from soft constraint violations. The formulation follows the ITC-2019 specification, which models both logistical feasibility and pedagogical preferences. Let $\mathcal{C}$ be the set of courses, $\mathcal{R}$ the set of rooms, and $\mathcal{T}$ the set of time slots. The objective is to find a mapping $f : \mathcal{C} \to \mathcal{R} \times \mathcal{T}$ that satisfies *Hard* and *Soft* constraints.

Hard constraints capture feasibility requirements that must not be violated. These include:

- **RoomNoOverlap:** No two classes may be scheduled in the same room at the same time.
- **SameRoom:** All sessions of a given course must take place in the same room.
- **SameStart:** Certain co-requisite classes are required to start simultaneously.
- **RoomCapacity:** The assigned room must accommodate the expected number of enrolled students.

Soft constraints encode scheduling preferences and pedagogical quality guidelines. While violations of soft constraints do not invalidate a schedule, they incur penalties that contribute to the overall cost. These include:

- **Soft SameRoom constraints:** Classes that should ideally be held in the same room across different days or sessions.
- **Time and Room Penalties:** Classes need to try and be scheduled in their optimum time slots and prefered rooms. If they cannot be, then a penalty is applied.

The optimization objective is to find a feasible schedule that minimizes the total weighted sum of soft constraint penalties. The complexity arises from the exponential number of possible combinations and the interactions between constraints, making the problem NP-hard. This motivates the use of hybrid, learning-augmented solvers that can efficiently explore promising regions of the search space while ensuring constraint satisfaction.

Dataset

We use the `pu-d9-fal19_late` instance from the ITC-2019 benchmark (itc 2019), a large-scale and highly constrained dataset representing a real university scenario (Müller, Rudová, and Müllerová 2025). The instance includes 2,798 classes, over 200 rooms, and a weekly schedule with dozens of hard and soft constraints spanning multiple departments. This benchmark reflects the complexity of course planning in higher education and serves as a rigorous testbed for scalable timetabling algorithms, particularly those intended for deployment in socially impactful educational institutions.

Baseline Models

To evaluate the effectiveness of our hybrid solver, we compare it against two baselines: a standalone CP-SAT solver and a standalone Graph Neural Network (GNN) model. These baselines represent traditional symbolic optimizat ion and purely neural approaches, respectively.

CP-SAT Solver (Cold Start): We use Google's OR-Tools CP-SAT solver (Perron and Furnon 2019) as a strong baseline for symbolic optimization. The solver is initialized without any prior knowledge or guidance ("cold start") and attempts to construct a valid schedule by systematically exploring the search space. CP-SAT is known for its robust constraint satisfaction capabilities and has been widely adopted to solve NP-hard scheduling problems. However, its performance degrades on large-scale instances due to the exponential growth of the solution space and the complexity of the interacting constraints.

GNN-Only Model: To test the limits of a fully neural approach, we train a standalone GNN to directly predict room and time assignments for each class. The model uses the same heterogeneous graph structure as our hybrid method and is trained using supervision from optimally solved sub-instances. While the GNN achieves high prediction accuracy at the node level, it fails to produce globally feasible schedules on the full ITC-2019 instance. In particular, its outputs violate thousands of hard constraints, confirming that unconstrained learning alone is insufficient for complex timetabling tasks. This highlights the need for integrating neural predictions with a constraint-aware solver, motivating our hybrid design.

These baselines provide a clear contrast in capabilities: while CP-SAT ensures feasibility but suffers from inefficiency on large instances, the GNN captures structural regularities but lacks constraint enforcement. Our hybrid model aims to combine the strengths of both.

Hybrid GNN + CP-SAT Model

The core of our approach is a hybrid model that integrates a Graph Neural Network (GNN) with a Constraint Programming solver (CP-SAT). This architecture is designed to exploit the learning capabilities of neural networks and the rigorous constraints satisfaction guarantees of symbolic solvers. The integration follows a structured three-phase pipeline illustrated in Figures 1 and 2.

Training Data Generation To generate supervision signals for the GNN, we sample 100 sub-problems ("slices") from the full `pu-d9-fal19_late` dataset. Each slice includes 50 to 100 classes and their associated constraints, forming smaller problems that are still representative of the global structure. These are solved to optimality using CP-SAT. The resulting *problem-solution* pairs constitute the training set used to learn structural scheduling (See Figure 1).

We represent each sub-instance as a heterogeneous graph using PyTorch Geometric (Fey and Lenssen 2019). The graph comprises three node types: *class*, *room*, and *time*. Edges capture constraint relationships such as room compatibility, class-time conflicts, and synchronization requirements. This structure enables the GNN to reason about the problem's topology and relational constraints.

GNN Architecture and Training The GNN model is composed of multiple GATv2Conv layers, which apply attention mechanisms to selectively aggregate information from neighboring nodes. The network predicts room and time assignments for each class node. We train the GNN using a cross-entropy loss function over 200 epochs with the Adam optimizer. The objective is to minimize prediction error with respect to the optimal solutions from the sub-instances. As illustrated in the left panel of Figure 2, the training loop includes validation-based checkpointing to ensure model generalization.

Hinting Mechanism Once training is complete, the GNN is deployed to generate high-confidence predictions for the full scheduling instance. Each class node's output consists of raw scores (*logits*) for all possible room and time assignments, which are converted into probability distributions using softmax functions. For every class, we compute a composite *confidence score* as the product of the highest room and time probabilities, quantifying the model's certainty in its top prediction. The top 20% of class assignments with the highest confidence scores are then selected as *hints*.

These hints serve as a partial solution and are passed to the CP-SAT solver via its `AddHint` API, effectively "warm-starting" the solver with a strong initial configuration. This targeted initialization narrows the solver's search space, reducing computation time while preserving feasibility and improving solution quality.

The right panel of Figure 2 illustrates this inference and integration phase of the hybrid pipeline. A detailed outline of the hint selection process is provided in Algorithm 1.

Algorithm 1: GNN-Powered Hint Generation

Input: A large, unsolved timetabling problem instance P_{large}, a pre-trained GNN model M_{GNN}, and a hint selection percentage K

▷ Convert the problem instance into a graph representation

$G \leftarrow \text{ConvertToGraph}(P_{\text{large}})$

▷ Perform inference with the trained GNN

$M_{\text{GNN}}.\text{eval}()$

$(\text{scores}_{\text{room}}, \text{scores}_{\text{time}}) \leftarrow M_{\text{GNN}}(G)$

▷ Calculate confidence scores for each class assignment

$\text{probs}_{\text{room}} \leftarrow \text{softmax}(\text{scores}_{\text{room}})$

$\text{probs}_{\text{time}} \leftarrow \text{softmax}(\text{scores}_{\text{time}})$

$\text{confidence} \leftarrow \max(\text{probs}_{\text{room}}) \times \max(\text{probs}_{\text{time}})$

▷ Select the top-K most confident classes to generate hints for

$\text{top\_k\_indices} \leftarrow \text{TopK}(\text{confidence}, K \cdot |\text{classes}|)$

▷ Generate the final set of hints

$H_{\text{GNN}} \leftarrow \{\}$ ▷ Initialize empty set of hints

for each index i in top_k_indices **do**

  $c \leftarrow$ class at index i

  $\text{pred\_room\_idx} \leftarrow \arg\max(\text{scores}_{\text{room}}[i])$

  $\text{pred\_time\_idx} \leftarrow \arg\max(\text{scores}_{\text{time}}[i])$

  $(\text{time\_dict}, \text{room\_dict}) \leftarrow \text{TranslateIndices}(c, \text{pred\_room\_idx}, \text{pred\_time\_idx})$

  $H_{\text{GNN}} \leftarrow H_{\text{GNN}} \cup \{(c, (\text{time\_dict}, \text{room\_dict}))\}$

end for

Output: A set of high-quality hints H_{GNN} for the CP-SAT solver

Evaluation and Experimental Results

We evaluate the performance of our proposed hybrid solver through comprehensive experiments on university course timetabling benchmarks. Specifically, we compare it to two baselines: (1) a standalone CP-SAT solver that has no learned guidance, and (2) a standalone Graph Neural Network (GNN) trained to directly predict complete schedules. The objective is to determine whether the integration of structure-aware learned representations into a symbolic

solver pipeline can yield superior results in terms of solution quality and efficiency.

Evaluation Metrics. We use two primary metrics throughout our experiments:

- **Penalty Score:** A weighted sum of soft constraint violations in the generated schedule. Lower values reflect higher solution quality.
- **Solver Time:** The wall-clock time (in seconds) taken to produce the best solution found within a fixed timeout. Lower times indicate greater efficiency.

Training Behavior of the GNN

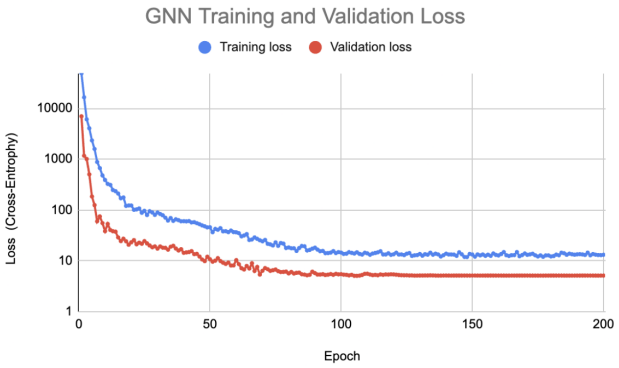

Figure 3: Training and validation loss per epoch for the GNN model. Loss convergence indicates stable generalization.

To facilitate effective scheduling guidance, we first trained the GNN model using small, fully solvable subinstances derived from the main scheduling problem. These subinstances are representative of localized structures within the larger instance and include complete, optimal labels to ensure high-quality supervision. Training was conducted for 200 epochs using standard stochastic optimization. Figure 3 illustrates the training and validation loss curves.

The training trajectory indicates a steady decrease in both training and validation loss, eventually plateauing after approximately 150 epochs. This behavior suggests that the model successfully captured generalizable structural patterns without overfitting. The stability of the validation loss reinforces the suitability of the GNN as a heuristic predictor within the hybrid framework (Fey and Lenssen 2019).

Performance on the Main Benchmark

Our primary benchmark uses the `pu-d9-fal19_late` dataset from ITC 2019, which contains 2,798 courses and numerous soft constraints. Table 1 presents comparative results for the standalone GNN, CP-SAT, and the hybrid GNN+CP-SAT models.

The standalone GNN, although successfully trained, failed to construct a feasible schedule. It covered only 2,690 out of 2,798 classes and resulted in more than 20,000 hard constraint violations. This confirms the limitations of direct neural solvers in handling complex, hard-constrained combinatorial domains.

In contrast, both the CP-SAT baseline and the hybrid model produced complete and feasible schedules. However, the quality of the solutions diverged significantly. The CP-SAT solver, operating without any prior knowledge, achieved a final penalty score of 196,026 after 3,182.44 seconds. The hybrid model, guided by GNN-generated hints, attained a substantially lower penalty of 162,796 using nearly identical compute time (3,181.27 seconds). This constitutes a **16.95%** reduction in penalty, underscoring the effectiveness of learned structural priors in reducing search overhead and improving solution quality.

Importantly, this improvement was achieved without modifying the CP-SAT engine or adding any post-processing steps. The sole intervention was the provision of warm-start hints derived from the GNN's output, demonstrating that learned representations can meaningfully accelerate symbolic search even when used in a lightweight integration.

Model	Classes Scheduled	Feas.	Hard Violations	Hints Gen.	Final Penalty	Solver Time (s)
GNN	2690/2798	No	>20k	–	158,018*	–
CP-SAT	2798/2798	Yes	0	0	196,026	3182.44
GNN+CP-SAT	2798/2798	Yes	0	451	162,796	3181.27

Table 1: Performance on `pu-d9-fal19_late`. The hybrid model reduces penalty by 16.95% over CP-SAT with equivalent runtime.

Generalization to Unseen Instances

To test the robustness and generalization of the learned GNN representations, we evaluated the hybrid solver on four additional ITC 2019 instances. These instances varied in size (from 150 to 2,798 classes) and constraint configurations, and the GNN was not fine-tuned on any of them. The goal was to assess whether the model learned transferable scheduling heuristics or simply overfitted to the training instance.

Table 2 summarizes the results. In nearly all cases, the hybrid solver matched or exceeded the performance of the CP-SAT baseline. Notable improvements include:

- **tg-spr18 (676 classes):** Penalty reduced from 5,300 to 4,694 (**11.4%** improvement).
- **muni-pdfx-fal17 (R2, 150 classes):** Runtime reduced from 140.36s to 110.67s (**21.2%** speed-up).
- **pu-d9-fal19 (exp 1):** Penalty dropped from 221,616 to 172,655 (**22.1%**), and runtime from 3,204s to 2,981s (**7%**).
- **tg-fal17 (711 classes):** Minor runtime gain of **6.7%**, with equal penalty.

Crucially, no degradation was observed in any of the tested instances. The hybrid model either improved penalty, reduced runtime, or maintained baseline performance, reinforcing the model's ability to generalize to novel timetabling scenarios. The GNN's representations, though trained on a

Dataset Instance	Size	Model	Penalty	Time (s)	Improvement
tg-spr18	676	Baseline	5,300	322.20	–
tg-spr18	676	Hybrid	4,694	322.07	11.4% (Penalty)
muni-pdfx-fal17 (R1)	300	Baseline	30	51.82	–
muni-pdfx-fal17 (R1)	300	Hybrid	30	50.02	3.5% (Time)
muni-pdfx-fal17 (R2)	150	Baseline	12	140.36	–
muni-pdfx-fal17 (R2)	150	Hybrid	12	110.67	21.2% (Time)
tg-fal17	711	Baseline	80	37.79	–
tg-fal17	711	Hybrid	80	35.25	6.7% (Time)
pu-d9-fal19 (exp 1)	2798	Baseline	221,616	3204.37	–
pu-d9-fal19 (exp 1)	2798	Hybrid	172,655	2981.25	22.1% (P), 7% (T)
pu-d9-fal19 (Final)	2798	Baseline	196,026	3181.25	–
pu-d9-fal19 (Final)	2798	Hybrid	162,796	3181.27	16.95% (Penalty)

Table 2: Generalization results on unseen instances. Hybrid solver consistently improves penalty (P) or runtime (T), or both.

single instance, appear to encode domain-agnostic structural priors applicable across diverse university settings.

Our experiments provide strong empirical support for the core hypothesis of this work: combining structure-aware learned models with classical solvers leads to better performance in complex combinatorial tasks. Specifically:

- The hybrid solver reduced penalty by up to **22.1%** and improved solver time by up to **21.2%** on diverse benchmarks.
- It maintained feasibility and robustness across all tested instances.
- The GNN component generalized well despite being trained on a single dataset instance, highlighting the potential for transfer in neural-symbolic scheduling systems.

These findings position hybrid ML+CP frameworks as a promising direction for real-world, large-scale scheduling applications. By integrating learned structural priors with symbolic solvers, such systems can produce high-quality solutions with greater efficiency. Moreover, the demonstrated generalization across diverse scheduling contexts suggests strong potential for scalable deployment across educational institutions and operational domains.

As these systems influence resource allocation and institutional planning, it is essential to consider their impact on stakeholders. While our approach optimizes for constraints defined by human experts, fairness, transparency, and adaptability to evolving institutional priorities remain important. Future work may include incorporating human-in-the-loop mechanisms or fairness-aware constraints to ensure equitable scheduling outcomes.

Ablation Studies

To better understand the contributions of key components in our hybrid architecture, we conducted a series of ablation studies on the pu-d9-fal19 instance. These experiments investigate the impact of the (1) hint coverage, (2) model width, and (3) training data variations on overall performance.

Hint Coverage. The default hybrid model uses the top 20% of class assignments (based on GNN confidence scores) as hints. We evaluated alternative thresholds of 10%, 30%, and 50%. Results showed that 20% provided the best trade-off: lower thresholds resulted in insufficient guidance, while higher thresholds introduced noisy or incorrect hints, occasionally harming solver performance. This confirms the importance of selecting a calibrated confidence cutoff.

Model Width. We varied the number of hidden channels in the GNN from 128 to 256 across three configurations (WIDE1: 128, WIDE2: 192, WIDE3: 256). Increasing the width did not significantly improve solver speed, and in one case (WIDE3) the hybrid solver slightly underperformed the baseline. This suggests that excessively wide architectures can introduce redundant features or overfitting, providing no additional benefit to the CP-SAT solver.

GNN Architecture	Final Penalty (Hybrid / Baseline)	Solver Time (s) (Hybrid / Baseline)	Speedup (%)	Outcome
WIDE1 (128 hidden)	39 / 39	77.30 / 77.76	0.59	Success
WIDE2 (192 hidden)	1 / 1	65.02 / 65.87	1.30	Success
WIDE3 (256 hidden)	1 / 1	66.52 / 66.48	-0.06	Failure to improve

Table 3: Ablation study on GNN model width for 400-class instances. The hybrid solver is compared with the cold-start CP-SAT baseline, and speedup is the relative runtime improvement.

Training Data Variations. To evaluate how the amount of training data influences hybrid solver performance, we generated supervised training datasets of 50, 100, and 150 subinstances for the 1,000-class problem. Results indicate that increasing the training set size improves the GNN's ability to generate high-quality hints, yielding better penalty reductions and modest runtime gains. With only 50 training instances, the hybrid solver achieved a 63.5% penalty reduction but required 10.6% more runtime than the baseline. With 100 and 150 training instances, the hybrid solver became both faster (up to 2.44% speedup) and more effective (up to 24.6% penalty reduction), demonstrating that richer training data leads to more confident and useful hint generation for the CP-SAT solver.

Training Instances	Hints Gen.	Final Penalty (Hybrid / Baseline)	Solver Time (s) (Hybrid / Baseline)	Speedup (%)	Penalty Reduction (%)
50	4	3,199 / 8,774	364.35 / 329.57	-10.56	63.5
100	6	7,363 / 8,222	328.91 / 337.15	2.44	10.45
150	21	7,327 / 9,721	321.80 / 327.33	1.69	24.63

Table 4: Ablation study on varying GNN training data sizes for the 1,000-class instance. Penalty reduction is computed relative to the cold-start CP-SAT baseline, and speedup is the relative runtime improvement.

Discussion

Our hybrid GNN+CP-SAT framework proposed in this paper offers promising results for large-scale university

timetabling, achieving improvements in both solution quality and runtime efficiency. Beyond computational performance, the broader significance of this work lies in its potential to improve the quality and equity of education delivery, an objective squarely aligned with the goals of AI for Social Good.

In real-world university settings, poor timetabling can impose hidden costs on students and staff. For example, schedules that cluster all of a student's classes into long, exhausting days, or scatter them across an inefficient week, can disadvantage those with caregiving responsibilities, part-time jobs, disabilities, or limited access to transportation. Similarly, faculty schedules that disproportionately favor senior instructors or particular departments may reflect legacy bias rather than pedagogical priorities. Automating timetabling without explicit fairness considerations risks reinforcing such inequities.

Our approach offers several mechanisms for mitigation. First, by using optimally solved synthetic subinstances for GNN training, the system avoids direct learning from potentially biased historical schedules. Second, the hybrid design allows fairness-aware objectives to be injected into the CP-SAT solver as soft constraints—enabling institutions to prioritize, for example, schedule compactness for commuting students or equitable time-slot allocation across departments. Third, the framework supports human-in-the-loop workflows, giving administrators oversight and the ability to adjust schedules in response to local needs or equity concerns.

To assess the model's ability to handle specific fairness-aware objectives, we conducted an experiment focusing on the "SameRoom" soft constraint from the ITC-2019 dataset. This constraint penalizes schedules where related classes are not assigned to the same room. We ran two tests: one where the constraint was omitted from the solver's objective function, and one where it was actively included.

The results demonstrated the significant impact of explicitly defining the constraint. When the "SameRoom" objective was added, the penalty score from its violations was reduced by over 94%—from 1530 to just 80 in the baseline CP-SAT run. More importantly, guiding the solver with this constraint led to a much more efficient overall schedule. The total solution penalty for both the baseline and hybrid models dropped from over 2200 to 862 when the constraint was enabled. This shows that adding targeted soft constraints can help the solver find substantially better global solutions, rather than simply causing a trade-off between competing objectives. In the final test, both the hybrid and baseline models found a solution with the same optimal penalty of 862, demonstrating the framework's effectiveness in incorporating and optimizing for specific, complex scheduling requirements.

From a fairness perspective, this result highlights how encoding domain-specific soft constraints can lead to more equitable outcomes. For example, enforcing the "SameClass" constraint—where all students enrolled in a course section are assigned the same lecture-recitation combination—ensures parity in instructional delivery. Without such constraints, students in the same section might be scattered across recitations occurring hours or even days apart, introducing disparities in access to instruction, feedback, and assessment conditions. This demonstrates how fairness-aware objectives can be operationalized as soft constraints and effectively optimized within our framework.

From a deployment perspective, institutional adoption of AI-generated schedules raises questions of transparency and trust. Stakeholders, including students, faculty, and planners, must understand how decisions are made and have the means to audit or contest outcomes. To this end, our method is designed with interpretability in mind: the GNN-generated hints are human-readable, and the final schedules remain fully auditable through CP-SAT's constraint logs. Moreover, the modularity of the system makes it compatible with interfaces that allow users to test "what-if" scenarios or explore trade-offs between efficiency and fairness.

Finally, this work opens up opportunities for deeper collaboration between AI researchers, education policy makers, and campus planning offices. By co-designing constraints, feedback loops, and user interfaces, future versions of the system could enable participatory scheduling, where equity is not only an outcome, but a principle embedded into the system's design. In this way, the work moves beyond technical optimization to serve broader social goals, helping institutions create timetables that are not just efficient, but inclusive and responsive to the diverse realities of their communities.

Conclusion and Future Work

In this paper, we presented a hybrid approach to university course timetabling that combines Graph Neural Networks (GNNs) with the CP-SAT constraint solver. The GNN is trained on optimally solved subinstances and used to generate high-confidence scheduling hints that warm-start the CP-SAT solver. This integration improves both solution quality and runtime efficiency. Experiments on ITC 2019 benchmarks show that our method consistently outperforms standalone baselines, reducing penalty scores by up to 22.1% while maintaining feasibility and scalability. The approach generalizes well across unseen instances, and ablation studies confirm the value of model architecture, hint coverage, and training data size.

Beyond performance, we addressed the ethical considerations of automated scheduling, including risks of bias propagation and the need for fairness-aware objectives. Our framework supports human-in-the-loop intervention and can be extended to accommodate institutional diversity and inclusion goals.

Future work includes enhancing the GNN with more expressive architectures such as Graph Transformers, developing adaptive or instance-specific hinting strategies, and formally analyzing the generalization behavior of learned scheduling heuristics. We also envision building foundation GNN models trained on a broad set of institutional data, enabling rapid deployment across universities. Finally, we aim to extend this hybrid framework to other combinatorial problems and integrate fairness constraints more deeply into the optimization process to support equitable, real-world decision-making.

References

2019. ITC 2019 Timetabling Dataset. https://www.itc2019.org/instances/a. Accessed: 2025-07-25.

Abdullah, S.; and Turabieh, H. 2012. On the use of multi neighbourhood structures within a Tabu-based memetic approach to university timetabling problems. *information sciences*, 191: 146–168.

Abdullah, S.; Turabieh, H.; McCollum, B.; and McMullan, P. 2012. A hybrid metaheuristic approach to the university course timetabling problem. *Journal of Heuristics*, 18(1): 1–23.

Błazewicz, J.; Ecker, K. H.; Schmidt, G.; and Weglarz, J. 1994. *Scheduling in computer and manufacturing systems*. Springer.

Bonutti, A.; De Cesco, F.; Di Gaspero, L.; and Schaerf, A. 2012. Benchmarking Curriculum-Based Course Timetabling: Formulations, Data Formats, Instances, Validation, Visualization, and Results. *Annals of Operations Research*, 194: 59–70.

Burke, E.; Jackson, K.; Kingston, J. H.; and Weare, R. 1997. Automated University Timetabling: The State of the Art. *The Computer Journal*, 40(9): 565–571.

Cappart, Q.; Chételat, D.; Khalil, E. B.; Lodi, A.; Morris, C.; and Velickovic, P. 2023. Combinatorial optimization and reasoning with graph neural networks. *Journal of Machine Learning Research*.

Ceschia, S.; Di Gaspero, L.; and Schaerf, A. 2023. Educational timetabling: Problems, benchmarks, and state-of-the-art results. *European Journal of Operational Research*, 308(1): 1–18.

Fey, M.; and Lenssen, J. E. 2019. Fast Graph Representation Learning with PyTorch Geometric. arXiv:1903.02428.

Gasse, M.; Chételat, D.; Ferroni, N.; Charlin, L.; and Lodi, A. 2019. Exact combinatorial optimization with graph convolutional neural networks. In *Advances in Neural Information Processing Systems*.

Khalil, E.; Le Bodic, P.; Song, L.; Nemhauser, G.; and Dilkina, B. 2016. Learning to branch in mixed integer programming. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30.

Khalil, E. B.; Dai, H.; Zhang, Y.; Dilkina, B.; and Song, L. 2017. Learning Combinatorial Optimization Algorithms over Graphs. In *Advances in Neural Information Processing Systems*. https://papers.nips.cc/paper_files/paper/2017/hash/d9896106ca98d3d05b8cbdf4fd8b13a1-Abstract.html.

Lewis, R. 2008. A survey of metaheuristic-based techniques for university timetabling problems. *OR spectrum*, 30(1): 167–190.

Lewis, R.; Paechter, B.; and McCollum, B. 2007. Post Enrolment-based Course Timetabling: A Description of the Problem Model Used for Track Two of the Second International Timetabling Competition. Technical report, Cardiff University, Cardiff Business School. Cardiff Accounting and Finance Working Paper.

Lewis, R.; and Thompson, J. 2015. Analysing the effects of solution space connectivity with an effective metaheuristic for the course timetabling problem. *European Journal of Operational Research*, 240(3): 637–648.

Müller, T.; Rudová, H.; Müllerová, Z.; et al. 2018. University course timetabling and international timetabling competition 2019. In *Proceedings of the 12th international conference on the practice and theory of automated timetabling (PATAT-2018)*, volume 1, 5–31.

Müller, T.; Rudová, H.; and Müllerová, Z. 2025. Real-world university course timetabling at the International Timetabling Competition 2019. *Journal of Scheduling*, 28(2): 247–267.

Perron, L.; and Furnon, V. 2019. OR-Tools. https://developers.google.com/optimization/introduction. Accessed: 2025-07-25.