A General Formulation for the Teacher Assignment Problem: Computational Analysis Over a Real-World Dataset

Moa Johannesson[1*], Lina Brink[1*], Alvin Combrink[1], Sabino Francesco Roselli[1], Martin Fabian[1]

***Abstract*— The Teacher Assignment Problem is a combinatorial optimization problem that involves assigning teachers to courses while guaranteeing that all courses are covered, teachers do not teach too few or too many hours, teachers do not switch assigned courses too often and possibly teach the courses they favor. Typically the problem is solved manually, a task that requires several hours every year. In this work we present a mathematical formulation for the problem and an experimental evaluation of the model implemented using state-of-the-art SMT, CP, and MILP solvers. The implementations are tested over a real-world dataset provided by the Division of Systems and Control at Chalmers University of Technology, and produce teacher assignments with smaller workload deviation, a more even workload distribution among the teachers, and a lower number of switched courses.**

I. Introduction

Personnel scheduling is a cornerstone of operational efficiency in many service-oriented sectors — like healthcare and education — where the primary resource to allocate are people. The challenge of assigning limited human resources to tasks while satisfying complex constraints is a classic combinatorial optimization problem. In practice, this process is frequently handled manually, which, while allowing for implicit inclusion of nuanced preferences, is often time-consuming, prone to errors, and in many cases results in suboptimal solutions. Consequently, a significant amount of research has aimed to automate these tasks.

A key challenge in implementing automated personnel scheduling is the difficulty of a priori formalizing all constraints. Practitioners often possess tacit knowledge, such as interpersonal dynamics and various personal circumstances, which are essentially unmodeled constraints. An "optimal" solution that violates these implicit rules is practically useless. Therefore, a solver should not be viewed merely as a one-shot optimizer, but instead as a decision-support tool within an iterative, human-in-the-loop workflow. To support this, the solver must allow to easily express and incorporate new constraints, and compute its result quickly so as to enable rapid re-solving during planning sessions.

Within the educational domain, the majority of research has focused on problems such as the *University Course Timetabling* or the *University Examination Timetabling* [1], which are primarily concerned with the temporal allocation of events to time-slots and rooms. A distinct and equally critical sub-problem is the *Teacher Assignment Problem* (TAP) [2, 3, 4]. Unlike rostering (e.g., Nurse Rostering [5]), which focuses on temporal shift coverage and sequencing, the TAP is a resource allocation problem. It requires matching teachers with heterogeneous skills to specific courses over a longer horizon (e.g., an academic year), often serving as a precursor to a timetabling phase. Despite its practical ubiquity, the TAP has received little attention in the literature compared to its timetabling and rostering counterparts [4].

The broader educational timetabling literature has historically favored metaheuristic solution methods due to the complexity of full university timetabling problems [6]. However, though hybrid heuristic/exact methods are often state-of-the-art for large-scale timetabling [1, 7], exact methods remain highly relevant for specific sub-problems like assignment, where optimality guarantees are valuable for fairness.

Mixed-Integer Linear Programming (MILP) can be regarded as an established "exact" baseline, however, comparative studies suggest that other modeling paradigms have much to bring to the table. For instance, Satisfiability Modulo Theories (SMT) has shown promise in adjacent domains, such as Nurse Rostering [8] and Job-Shop Scheduling [9], while the relatively younger CP-SAT — which blends SAT logic with constraint programming — performs comparatively on certain combinatorial optimization problems, such as Petri net optimization [10] and Minimum Spanning Trees [11]. This suggests that paradigms alternative to MILP, like SMT and CP-SAT, may offer distinct advantages for the combinatorial structure of the TAP. To our knowledge, however, a rigorous comparison of these paradigms specifically for the TAP remains absent from the literature.

Our contributions in this work are as follows:

1) We formulate a TAP model based on real-world data from the Division of Systems and Control, at Chalmers University of Technology, Göteborg, Sweden, over multiple years, incorporating constraints on workload deviation, competence, and course continuity.
2) We implement and benchmark this model using solvers based on three distinct paradigms: SMT via Z3 [12], MILP via Gurobi [13] and SCIP [14], and CP-SAT [15] via OR-Tools. Their computational tractability and suitability for an iterative workflow are evaluated.
3) We show that our optimization-based approach significantly outperforms historical manual assignment and thereby validate its practical utility.

*The authors contributed equally to this work. {moajohan, linabri}@student.chalmers.se

[1]Division of Systems and Control, Department of Electrical Engineering, Chalmers University of Technology, Göteborg, Sweden {combrink, rsabino, fabian}@chalmers.se

This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation

II. Teacher Assignment Problem Model

The teacher assignment problem presented in this work concerns assigning PhD students, henceforth referred to as *Teaching Assistants* or TAs, to courses taught by the Division of Systems and Control, in the Department of Electrical Engineering at Chalmers University of Technology. The input data presented in this work is based on real course and staff data provided by the department. The following paragraphs introduce the structure and requirements of the courses, and the TAs' goals, skills, and preferences.

a) Teaching Assistant Set: We have a finite set $\mathcal{S}$ of TAs. Each TA can teach for up to 5 years, depending on their contract; $\upsilon_s \in \{1,2,3,4,5\}$ is the academic year for TA $s \in \mathcal{S}$. Moreover, each TA has a workload $\phi_s \in [0,1], \forall s \in \mathcal{S}$ that ranges from 0% to 100%, where 100% corresponds to $\vartheta = 350$ hours per year. It typically happens for a given year that TAs do not teach exactly the amount of hours that they are supposed to according to their contract; hence, for each TA $s \in \mathcal{S},\ \theta_s \in \mathbb{Z}$ is the amount of teaching hours left from previous years, which can be either positive or negative. From these parameters, the target total workload for this year for each TA is computed as $\Theta_s = \phi_s \cdot \vartheta + \theta_s$.

b) Course and Task Sets: We have a set of courses $\mathcal{C}$. Each course $c \in \mathcal{C}$ is characterized by tasks from the set $\mathcal{T}$, containing *course administration*, *exercise session*, *problem solving session*, *lab session*, *computer session*, *assignment supervision*, *assignment evaluation*, *project supervision*, *exam evaluation*, and *other*. Except for *course administration* (t^* in the model), which is always present, not all courses involve all tasks, and for tasks not present in a course their allotted time is set to 0.

Based on the number of attending students, each task t in course c has a required number of TAs $\rho_{c,t} \in \mathbb{N}$ and a total time $\tau_{c,t} \in \mathbb{N}$, which is the sum of hours allocated to the TAs assigned to the task. Finally, in order to avoid TAs being assigned to tasks for only a very limited time, it is possible to specify the minimum number of hours $\epsilon \in \mathbb{N}$ a TA has to teach a task if they are assigned to it.

c) Additional Parameters: In addition to the previously defined sets and parameters, some features of the problem regard both courses and TAs. To begin with, TA s is able to express their preference $\pi_{s,c} \in \{-1,0,1\}$ on course c to say that they are not in favor, indifferent, or in favor of teaching the course. $\xi_{s,c} \in \{0,1\}$ is equal to 1 if TA s is forbidden to teach course c, 0 otherwise. This feature is typically used when TAs are either not qualified or are not available for a specific period of time due to business or personal reasons. Moreover, one of the optimization criteria is the consistency of courses assigned to a TA throughout the years, as teaching the same course for multiple years reduces the preparation overhead incurred when a TA must learn a new subject. We denote a *new course* as a course that an assigned TA did not teach in the previous year. Therefore $\kappa_{s,c} \in \{0,1\}$ is equal to 1 if TA s taught course c the year before, 0 otherwise.

Bounds are given on workload deviation, new courses, number of TAs per course, and number of courses per TA. Since optimization in the SMT formulation is performed by maximizing the number of satisfied soft constraints, there are both hard and soft bounds.

- λ^H, λ^S: hard and soft bounds on workload deviation.
- σ^H, σ^S: hard and soft bounds for the number of new courses.
- μ^H, μ^S: hard and soft bounds for the number of courses per TA.
- ν^H: hard bound on the number of TAs per course.
- ν^S: soft bound for additional TAs beyond the required number of TAs for a task $\rho_{c,t}$.

A. Decision Variables

The following is the set of variables defined to model the problem:

- $w_{s,c} \in \{0,1\}$: 1 if TA s teaches course c, 0 otherwise.
- $y_{s,c,t} \in \{0,1\}$: 1 if TA s teaches task t of course c, 0 otherwise.
- $x_{s,c,t} \in \mathbb{N}$: the number of hours that TA s is assigned to task t in course c.
- $h_s \in \mathbb{N}$: total assigned workload for TA s.
- $n_{c,t} \in \mathbb{N}$: number of TAs assigned to task t in course c.
- $z_s \in \mathbb{N}$: number of new courses c assigned to TA s.

B. SMT Formulation of the Teacher Assignment Problem

The model includes both hard (1)-(13) and soft (14)-(18) constraints. Hard constraints must always be satisfied lest the model be infeasible, while soft constraints may be violated without causing the model to be infeasible.

$$\xi_{s,c} = 1 \rightarrow \sum_{t \in \mathcal{T}} x_{s,c,t} = 0 \qquad \forall s \in \mathcal{S}, c \in \mathcal{C} \quad (1)$$

$$h_s = \sum_{c \in \mathcal{C}, t \in \mathcal{T}} x_{s,c,t} \qquad \forall s \in \mathcal{S} \quad (2)$$

$$\sum_{t \in \mathcal{T}} x_{s,c,t} > 0 \rightarrow w_{s,c} = 1 \qquad \forall s \in \mathcal{S}, c \in \mathcal{C} \quad (3)$$

$$z_s = \sum_{c \in \mathcal{C}} w_{s,c}(1 - \kappa_{s,c}) \qquad \forall s \in \mathcal{S} \quad (4)$$

$$\tau_{c,t} = 0 \rightarrow \sum_{s \in \mathcal{S}} x_{s,c,t} = 0 \qquad \forall c \in \mathcal{C}, t \in \mathcal{T} \quad (5)$$

$$\sum_{s \in \mathcal{S}} x_{s,c,t} = \tau_{c,t} \qquad \forall c \in \mathcal{C}, t \in \mathcal{T} \quad (6)$$

$$\min(\tau_{c,t}, \epsilon) \leq x_{s,c,t} \leq \tau_{c,t} \qquad \forall s \in \mathcal{S}, c \in \mathcal{C}, t \in \mathcal{T} \quad (7)$$

$$n_{c,t} \geq \rho_{c,t} \qquad \forall c \in \mathcal{C}, t \in \mathcal{T} \quad (8)$$

$$|h_s - \Theta_s| \leq \lambda^H \qquad \forall s \in \mathcal{S} \quad (9)$$

$$\sum_{c \in \mathcal{C}} w_{s,c} \leq \mu^H \qquad \forall s \in \mathcal{S} \quad (10)$$

$$\sum_{s \in \mathcal{S}} w_{s,c} \leq \nu^H \qquad \forall c \in \mathcal{C} \quad (11)$$

$$z_s \leq \sigma^H \qquad \forall s \in \mathcal{S} \quad (12)$$

$$n_{c,t^*} \leq 1 \qquad \forall c \in \mathcal{C} \quad (13)$$

$$\upsilon_s \geq 5 \rightarrow h_s = \Theta_s \qquad \forall s \in \mathcal{S} \quad (14)$$

$$h_s \leq \lambda^S \qquad \forall s \in \mathcal{S} \quad (15)$$

$$z_s \leq \sigma^S \qquad \forall s \in \mathcal{S} \quad (16)$$

$$n_{c,t} \leq \rho_{c,t} + \nu^S \qquad \forall c \in \mathcal{C}, t \in \mathcal{T} \quad (17)$$

$$\sum_{c \in \mathcal{C}} w_{s,c} \leq \mu^S \qquad \forall s \in \mathcal{S} \quad (18)$$

$$\pi_{s,c} = 1 \rightarrow x_{s,c} = 1 \qquad \forall s \in \mathcal{S}, c \in \mathcal{C} \quad (19)$$

$$\pi_{s,c} = -1 \rightarrow x_{s,c} = 0 \qquad \forall s \in \mathcal{S}, c \in \mathcal{C} \quad (20)$$

Constraint (1) forbids TA-course assignments if the TA is not available or qualified. Constraint (2) connects the number of hours a TA teaches in each task of each course to their cumulative number of taught hours. Constraint (9) defines the workload deviation as the difference between theoretical and actual total workload. Constraint (3) states that if a TA teaches any amount of hours in a task of a specific course, that TA is assigned to the course. This constraint, together with (4), is used to keep track of new courses. Constraint (5) models that if a course does not include a task, no TA is assigned that task in that course. Constraint (6) states that the cumulative amount of hours TAs are teaching a task must be equal to the task required time. Constraint (7) bounds the number of hours that a TA is assigned to a task to be larger than ϵ and at most equal to the total time for that task. Constraint (8) ensures that a task is assigned at least the required number of TAs for that task. Constraint (10) ensures that each TA must not teach more than the allowed amount of courses. Constraint (11) ensures that each course must not have more than the allowed amount of TAs assigned to it. Constraint (12) limits the number of new courses a TA can be assigned. The task *course administration* may only be assigned to a single TA, a feature expressed by (13).

Constraints (14)–(20) are *soft*, and while the goal is to satisfy as many of them as possible, violating any number of them shall not result in an infeasible solution. Each soft constraint is associated with a weight, allowing the solver to balance competing objectives. Constraint (14) ensures that if a TA is in their fifth year, their assigned hours should equal their target hours, so possibly the deviation will be zero. Constraint (15) sets the assigned hours for a TA to be below a soft bound. Assigning new courses is penalized by (16) by softly constraining it to remain below a threshold. Constraint (17) softly limits the number of TAs assigned to a task. Constraint, (18) sets the courses taught by each TA below a given soft bound. Finally, TAs' preferences are accounted for in (19) and (20), rewarding assignments with $\pi_{s,c} = 1$ and penalizing assignments with $\pi_{s,c} = -1$.

In order to run the model using the MILP solvers and the constraint programming solver, all the constraints had to be linearized. This involved the definition of additional sets of decision variables, as well as additional constraints to connect the new variables to the existing ones. Moreover, the new model only involves hard constraints and the soft constraints are instead accounted for in the objective function. For the full formulation we refer the reader to the implementation provided on our Github repository[1].

TABLE I: Parameters and weights for hard and soft constraints respectively for each year.

Constraint	Type	2022	2023	2024	2025	2026
(9)	hard	100	120	150	150	160
(10)	hard	2	3	3	2	3
(11)	hard	8	8	8	8	8
(12)	hard	1	1	1	1	1
(15)	soft	30	30	30	30	30
(17)	soft	1	1	1	1	1
(18)	soft	5	5	5	5	5
(16)	soft	0	0	0	0	0

III. Experiments

In this section, we present the experimental set up used to evaluate the goodness of our model formulation, as well as the performance of the different solvers. When evaluating the SMT solver Z3, the optimization mode is enhanced, and soft constraints are added with associated weights. The solver searches for solutions that satisfy all hard constraints while minimizing the total penalty induced by soft constraint violations, according to their weights. When evaluating the MILP solvers and the CP solver, the optimization is performed by minimizing an objective function while satisfying the hard constraints. Each term of the objective function represents one of the soft constraints, with a coefficient corresponding to its weight.

The five problem instances used in the evaluation originate from the real-world instances of assigning PhD students as TAs to courses, with schedules available for the years 2022 to 2026. Each instance counts 50 TAs and 45 courses, each with up to 10 tasks to be assigned. No data exists for the TAs' preferences to courses since this was not taken into consideration. Therefore, all TA preferences are set to 0. The data is anonymized and made available[1], together with the implementation of the SMT, CP, and MILP models, as well as visualization support functions.

The solvers are tested on the same data as the manual solution each year, to fairly evaluate the goodness of the solution produced. Alternatively, the solver output from year x could be used as input for year $x+1$ to evaluate the benefit of long-ter use of the solver; However, this study is not included in this work. The parameter values for hard and soft constraints, shown in Table I, are fine-tuned for each year and are based on expert advise (the current person responsible for the schedule) and comparisons with the manual scheduling. The different solvers are compared with manual scheduling on workload deviation, number of assigned courses, and number of new courses per TA.

We begin by comparing the computation times shown in Table II. For each solver, a time limit of 3600 seconds is set; the solver will return the optimal solution as soon as it is computed and proven to be the optimal, or the best solution found within the time limit. The SMT solver Z3 timed out

[1]https://github.com/Chalmers-Control-Automation-Mechatronics/TASP

TABLE II: Time in seconds required by each solver and corresponding Root mean squared error (RMSE) between scheduled and target workloads for solver-based and manual TA assignments, computed for the five problem instances. When the time is equal to 3600 it means the solver has reached the time limit.

	2022		2023		2024		2025		2026	
	RMSE	Time	RMSE	Time	RMSE	Time	RMSE	Time	RMSE	Time
Z3	68.70	3600	100.95	3600	128.78	3600	116.51	3600	119.94	3600
CP-SAT	**2.84**	1349	**0.22**	1934	**24.82**	605	**38.2**	3600	**47.76**	49
SCIP	25.06	3600	24.81	3600	35.72	3600	45.17	3127	52.84	133.53
Gurobi	22.98	**33.21**	22.57	**95.72**	34.61	**16.42**	44.95	**17.29**	52.84	**14.01**
Manual	92.97	-	95.72	-	125.59	-	132.8	-	144.94	-

on all instances. The open source MILP solver SCIP timed out on 3 instances out of 5, while the CP solver CP-SAT timed out on the 2025 instance. Finally, the commercial solver Gurobi solved all instances in orders of magnitude less than all other solvers, remaining below 100 seconds on all instances.

We now compare the solvers and manual solution on the deviation between assigned hours and target hours. Table II shows the root mean squared error (RMSE) of the deviation, calculated for each solver and for the manual solution for each of the years 2022–2026. The SMT solver Z3 shows the worst performance, sometimes under-performing the manual solution. On the other hand, both the MILP solvers Gurobi and SCIP show a significantly better performance than the manual schedule, with Gurobi being slightly better than SCIP on each problem instance. Finally, CP-SAT outperforms both MILP solvers, showing the best performance on each of the five problem instances. Interestingly, the RMSE for year 2022 and 2023 is ten and one hundred times smaller, respectively, compared to the MILP solvers. When it comes to SCIP, this may be due to the fact that the solver timed out and was unable to find the optimal solution. However, when compared to Gurobi, both solvers found the optimal solution (except in 2025), and yet, of the many solutions with the same objective value, CP-SAT consistently chose one that led to a smaller RMSE. This shows how parameter tuning may be needed, depending on not only the problem instance but also on the solver.

The results of the solvers and the manual schedule on the workload deviation for year 2022 is shown in Figure 1. Each TA is represented as a dot with their target hours along the x-axis and assigned hours along the y-axis. The diagonal line shows the aim of aligned target and assigned hours. Dots above and below the reference line represent TAs that are over- and under-scheduled, respectively. It is immediately noticeable how the solutions from CP-SAT almost exactly coincide with the reference line; SCIP and Gurobi have almost identical solutions — close to the reference line but consistently slightly above target; Z3 performs significantly worse, with most TAs teaching below target; even so, the result is still visibly better than the manual schedule. Similar plots are produced for the remaining years 2023 to 2026 and available in the appendix in Figure 3.

The solutions produced manually and by the solvers are also compared on the number of assigned courses per TA and the number of new courses per TA. Generally, assigning new courses is undesirable as it involves preparing for the sessions to teach and, possibly, learning new topics; both tasks are time-consuming and typically not accounted for when assigning a new course. Therefore we try to keep the number of new courses taught by each TA as low as possible. The upper part of Figure 2 shows the number of TAs that are assigned 0 to 5 courses, respectively (no student is assigned 4 courses), while the lower part shows the number of TAs that are assigned 0 to 3 new courses, respectively. The requirements for 2022 were maximum 2 courses per TA and maximum 1 new course. Clearly, the manual schedule violates these constraints, likely due the difficulty of solving such problems manually; Z3 shows the highest number of TAs being assigned 2 courses and also the highest number of TAs being assigned a new course, the vast majority of them. All the other solvers perform identically in terms of number of assigned courses, while performing very similarly in terms of new courses, with Gurobi being slightly better than SCIP, being slightly better than CP-SAT. The difference between Gurobi and CP-SAT confirms our previous statements about parameter tuning needed to achieve identical solutions with different solvers. Similar results are achieved on the instances corresponding to 2023–2026; the reader is referred to figures 4 and 5 in the Appendix.

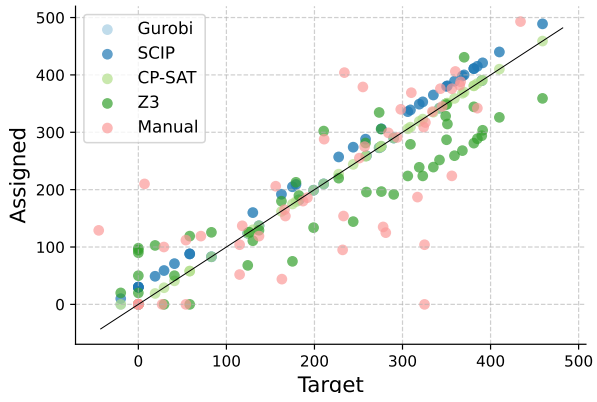

Fig. 1: Comparison of assigned versus target hours for each TA in year 2022 among solvers Gurobi, SCIP, CP-SAT, and Z3, as well as the manual teacher assignment.

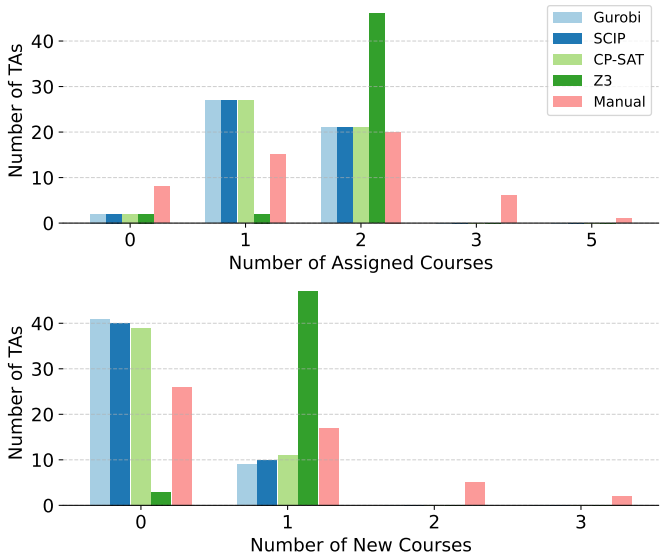

Fig. 2: Comparison of the number of assigned courses (upper) and number of new courses (lower) between solvers output and manual teacher assignment for year 2022.

IV. Conclusion

In this work we presented a mathematical formulation to model the teacher assignment problem. We implemented the model using state-of-the-art solvers from different communities, both commercial and open source; we tested such solvers on a dataset of real-world instances from the Division of System and Control at Chalmers University of Technology, Göteborg, Sweden. We optimized the problem in terms of workload deviation from the requirement, workload bounds for the individual TAs and number of new courses, and we obtained solutions that were superior compared to the manual schedule. Depending on the solver used, we were able to generate a solution within a few seconds up to one hour, while the manual schedule can take several hours over the entire year.

The obvious next step in this area of research would be to generate synthetic data that mimics the real one and test the scalability of this formulation with the different solvers; it would also be interesting to enrich the data with TA preferences on courses to evaluate how the corresponding term in the model's objective function affect the solution quality and solving time. Model decomposition could then be applied to the current formulation to increase scalability; finally, an interesting feature that could increase the model's usefulness is to produce and interpret the minimal infeasible constraint set for infeasible instances to provide insight on the cause of infeasibility.

References

[1] Sara Ceschia, Luca Di Gaspero, and Andrea Schaerf. "Educational timetabling: Problems, benchmarks, and state-of-the-art results". In: *European Journal of Operational Research* 308.1 (2023), pp. 1–18. ISSN: 0377-2217. DOI: https://doi.org/10.1016/j.ejor.2022.07.011.

[2] Michael W. Carter and Gilbert Laporte. "Recent developments in practical course timetabling". In: *Practice and Theory of Automated Timetabling II*. Ed. by Edmund Burke and Michael Carter. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 3–19. ISBN: 978-3-540-49803-2.

[3] Bruno Domenech and Amaia Lusa. "A MILP model for the teacher assignment problem considering teachers' preferences". In: *European Journal of Operational Research* 249.3 (2016), pp. 1153–1160.

[4] Cristian D. Palma, Pablo González-Brevis, Pamela A. Riffo, and Nicolás S. Montenegro. "A Proposed Mathematical Optimization Approach for Undergraduate Teaching Assistant Selection and Tutorial Scheduling". In: *INFORMS Transactions on Education* (2025). DOI: 10.1287/ited.2024.0095.

[5] Edmund K Burke, Patrick De Causmaecker, Greet Vanden Berghe, and Hendrik Van Landeghem. "The state of the art of nurse rostering". In: *Journal of scheduling* 7.6 (2004), pp. 441–499.

[6] Rudy A Oude Vrielink, EA Jansen, Erwin W Hans, and Jos van Hillegersberg. "Practices in timetabling in higher education institutions: a systematic review". In: *Annals of operations research* 275.1 (2019), pp. 145–160.

[7] Simon Kristiansen and Thomas Riis Stidsen. *A Comprehensive Study of Educational Timetabling - a Survey*. DTU Management Engineering Report 8.2013. DTU Management Engineering, 2013. ISBN: 978-87-93130-02-9.

[8] Alvin Combrink, Stephie Do, Kristofer Bengtsson, Sabino Francesco Roselli, and Martin Fabian. "A Comparative Study of SMT and MILP for the Nurse Rostering Problem". In: *arXiv preprint arXiv:2505.10328* (2025).

[9] Sabino Francesco Roselli, Kristofer Bengtsson, and Knut Åkesson. "SMT solvers for job-shop scheduling problems: Models comparison and performance evaluation". In: *2018 IEEE 14th international conference on automation science and engineering (CASE)*. IEEE. 2018, pp. 547–552.

[10] Bengt Lennartson. "Optimization of Timed Petri Nets using CP-SAT". In: *IFAC-PapersOnLine* 58.1 (2024), pp. 90–95.

[11] Roberto Montemanni and Derek H Smith. "On Solving the Minimum Spanning Tree Problem with Conflicting Edge Pairs". In: *Algorithms* 18.8 (2025), p. 526.

[12] Leonardo de Moura and Nikolaj Bjørner. "Z3: An Efficient SMT Solver". In: *Tools and Algorithms for the Construction and Analysis of Systems*. Vol. 4963. Lecture Notes in Computer Science. Springer, 2008, pp. 337–340.

[13] Gurobi Optimization, LLC. *Gurobi Optimizer Reference Manual*. 2024. URL: https://www.gurobi.com.

[14] Tobias Achterberg. "SCIP: solving constraint integer programs". In: *Mathematical Programming Computation* 1.1 (2009), pp. 1–41.

[15] Laurent Perron and Frédéric Didier. *CP-SAT*. Version v9.12. Google, Feb. 17, 2025. URL: https://developers.google.com/optimization/cp/cp_solver/.

V. Appendix

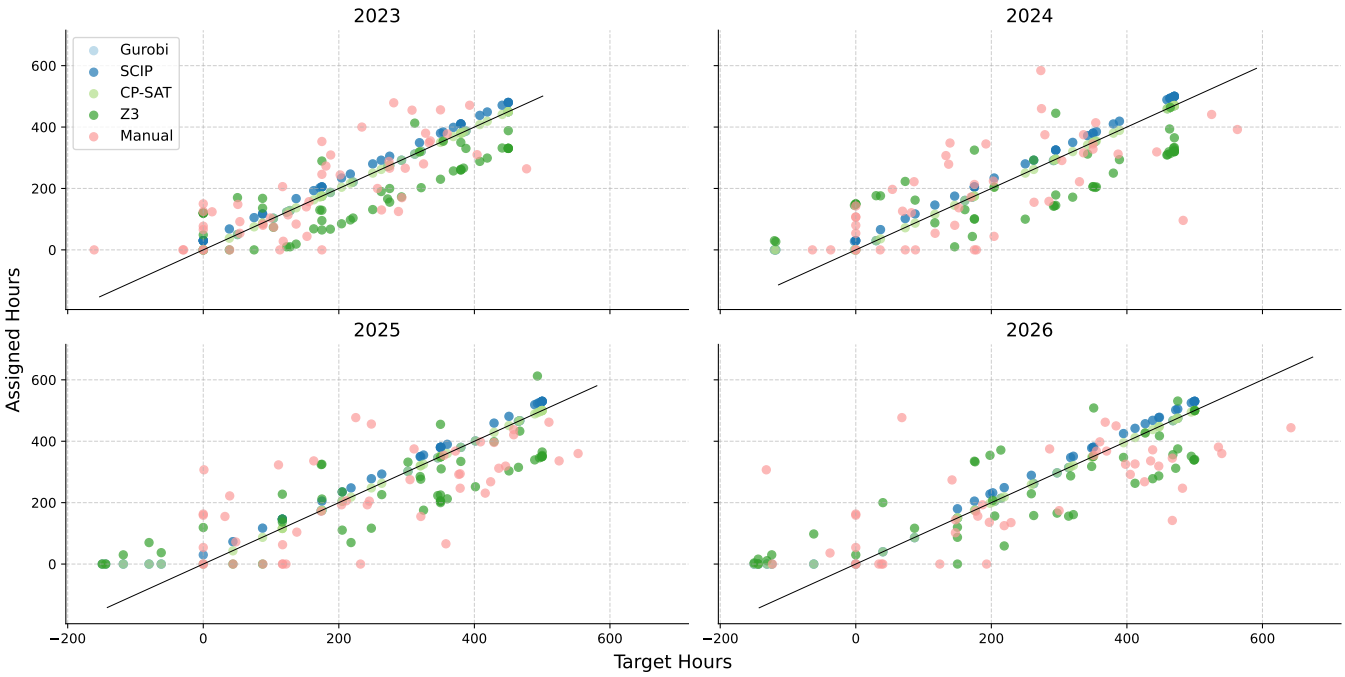

Fig. 3: Comparison of the TAs' assigned versus target hours between solvers and manual assignment for 2023–2026.

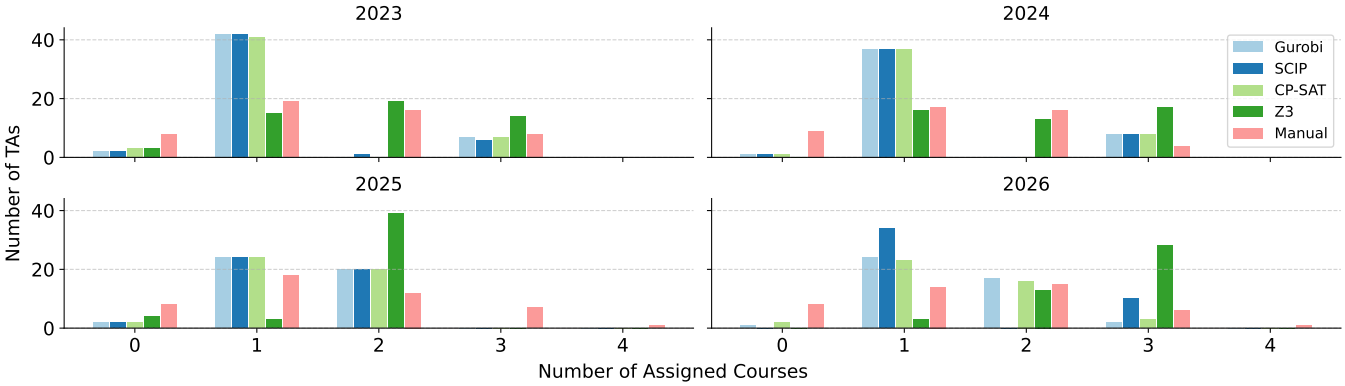

Fig. 4: Comparison of the number of assigned courses between solvers and manual assignment for 2023–2026.

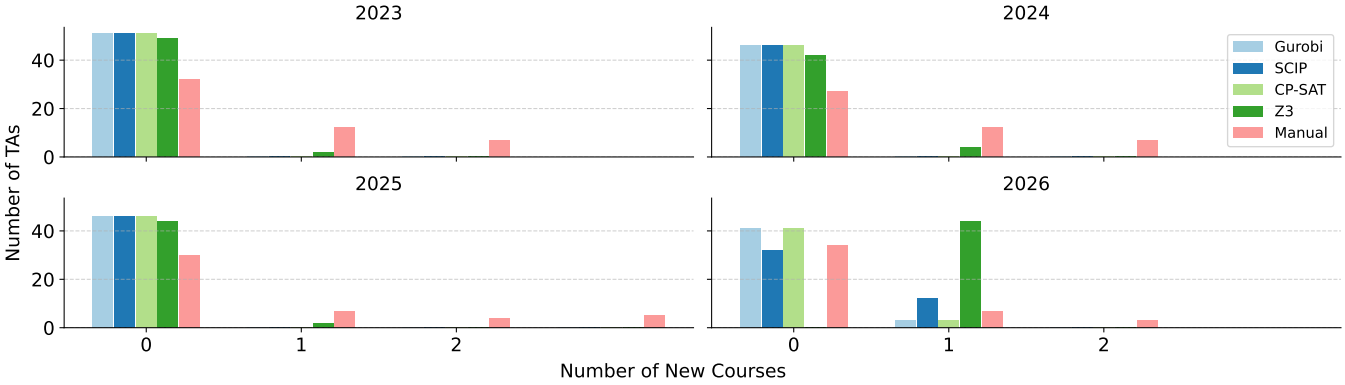

Fig. 5: Comparison of the number of assigned new courses between solvers and manual assignment for 2023–2026.