

Final Project Report

Group 1 Jasper Zhou (s2477464) Frank Fan (s2472647) Joyce Ren (s2297434)
Mia Fan (s2484172) Veronica Xu (s2564111)

1 Introduction

University course timetabling is a well-studied combinatorial optimisation problem that arises whenever courses and student cohorts must be assigned to a finite set of time periods without conflict [1]. The problem is known to be NP-hard in general, and practical instances are further complicated by programme-specific degree regulations, student preferences, and resource constraints.

A distinctive feature of the honours-level Mathematics curriculum at the University of Edinburgh is the high degree of student choice: Year 3 and Year 4 students select from gateway and optional courses, subject to complex credit and prerequisite rules that differ across single-honours and joint-programme degrees. This flexibility, while educationally desirable, dramatically expands the space of feasible timetables and makes clash-free scheduling substantially harder. Existing approaches to timetabling typically focus on feasibility or optimisation of isolated objectives, but often struggle to simultaneously accommodate diverse student pathways, programme-specific constraints, and timetable quality considerations at scale [2].

Against this background, the School of Mathematics is evaluating a proposed redesign of its honours curriculum. In this project, we optimise the timetable of the School of Mathematics, while the timetables of partner Schools for joint-degree students are taken as fixed external schedules. The problem is therefore to design a Mathematics timetable that remains feasible for students with heterogeneous external commitments. We investigate whether the new course structure can be timetabled effectively by formulating the problem as a constraint-programming model and solving it with Google OR-Tools [3].

Specifically, we investigate the feasibility and accessibility of the proposed curriculum under different programme structures, including the extent to which single-honours students can access all gateway courses, how joint-programme constraints affect course availability, and whether introducing structured pathways can improve overall timetable quality.

This project makes several contributions. First, we formulate the timetabling problem as a large-scale CP-SAT model that jointly captures course scheduling, student choices, and programme-specific requirements, while using student-type aggregation to reduce computational complexity. Second, we introduce a flexible and configurable modelling framework, in which key constraints (such as workload limits, gap control, and sequencing rules) are parameterised, allowing the timetable to be adapted to different policy settings without modifying the model structure. Third, we design a sequence of progressively refined scenarios to systematically evaluate the trade-off between feasibility, flexibility, and timetable quality. Finally, we demonstrate that high-quality, conflict-free timetables can be achieved at scale, and show that overly restrictive constraints can lead to significant degradation in solution quality.

The remainder of this report is organised as follows. Section 2 describes the timetabling scenario and defines what constitutes a good solution. Section 3 details our methodology, including student-type aggregation, the CP-SAT model, key constraints, and the objective function. Section 4 presents the experimental setup and data instances. Section 5 reports the main results and Section 6 provides a critical analysis; detailed per-scenario descriptions and results are collected in Appendix G. We conclude in Section 7.

2 Problem Description

2.1 Timetabling Scenario

The proposed curriculum offers **8 gateway courses** (4 per semester) which are the compulsory courses for single mathematics students, together with **4 optional mathematics courses** (2 per semester) for additional choices. Each course is composed of multiple teaching activities, including lectures (typically three per week) and workshops (one or two per week, possibly on a fortnightly basis). For courses with high enrolment, workshops may be offered in multiple parallel sections, which introduces additional combinatorial complexity in the allocation of students.

Programme requirements differ across student groups:

Single-honours Mathematics students are required to complete all 8 gateway courses across

Years 3 and 4, typically taking 4–6 courses per year (2–4 per semester), and must select optional courses to reach a total of 120 credits per year, with a balanced distribution across semesters. In contrast, joint-programme students (e.g. Mathematics combined with Computer Science, Physics, Economics, Business, or Philosophy) share the same mathematics curriculum but must allocate a minimum 40 credits to their partner discipline (referred to as *outside credits*). This requirement reduces the number of mathematics courses they can take and restricts their feasible course combinations.

To ensure conflict-free timetabling under different student requirements, we introduce the concept of **pathways**, which represent structured bundles of courses corresponding to typical student choices within each programme. Each pathway encodes a feasible combination of required and optional courses, allowing the model to capture diverse student demands while maintaining consistency in scheduling decisions.

2.2 What Defines a Good Timetable

We evaluate candidate timetables along three key dimensions, which will be analysed in detail in Section 5:

1. **Feasibility.** All degree regulations (e.g. credit bounds, gateway requirements, and outside-credit constraints) must be satisfied, and timetable conflicts should be avoided.
2. **Student perspective.** The timetable should provide a balanced and manageable schedule for students, including high levels of preference fulfilment, limited daily workload, controlled gaps between classes, and minimal consecutive sessions.
3. **Planner perspective.** From an operational viewpoint, the timetable should make efficient use of teaching resources, avoid excessive clustering of classes, and maintain a coherent structure (e.g. appropriate sequencing of lectures and workshops).

2.3 Scope and Simplifications

Included: course scheduling across two years and two semesters per year; weekly and fortnightly components with odd/even week patterns; multi-section workshops; aggregate student-type modelling; programme-specific degree rules.

Excluded / simplified: physical room assignment; individual student preferences beyond type-level aggregation; prerequisite chains between courses; instructor availability.

3 Methodology

3.1 Problem Formulation

We formulate the problem as a constraint-programming model with two coupled layers of decisions. The first layer determines the placement of Mathematics teaching activities in the Year 3 timetable. The second layer evaluates student-level feasibility: for each student, or for each aggregated student type, the model determines whether the required set of Mathematics courses can be attended without clashes, taking account of fixed outside-Mathematics commitments and programme rules.

Hard constraints enforce the structure of each course, the use of valid teaching slots, and the degree regulations governing gateway and optional-course choices. Soft constraints are used to express timetable-quality considerations such as workload balance, gap control, and undesirable sequencing patterns when these are included in a scenario.

To reduce model size, students with identical external timetables and the same admissible Mathematics choice structure are aggregated into student types. This preserves the relevant feasibility structure of the problem while substantially reducing the number of repeated constraints.

The resulting model is solved using Google OR-Tools CP-SAT solver [3]. Compared with mixed-integer programming (MIP) solvers, CP-SAT natively supports Boolean reification (e.g. `OnlyEnforceIf`), avoiding the need for big- M linearisations and leading to more compact and expressive formulations [4]. Furthermore, CP-SAT combines SAT techniques with linear programming relaxations and cutting planes, allowing it to provide optimality gap bounds, in contrast to purely heuristic approaches that cannot certify solution quality [5].

3.2 Model Assumptions

To ensure model tractability, we make the following implicit but necessary assumptions:

- **Unforeseen external disruptions are not explicitly considered**, such as severe weather conditions and industrial actions.

- **Inter-campus travel time is ignored.** In particular, Economics & Maths and Mathematics & Business may have classes in both GS and KB campuses on the same day.
- **Detailed teaching resource allocation is excluded from the model**, such as room assignment and instructor availability.
- **All student and course information is assumed to be complete and up to date.**

3.3 Key Modelling Innovations

This subsection highlights the key modelling innovations that improve both the computational efficiency and practical adaptability of the proposed timetabling framework.

Student Aggregation. Rather than modelling each student individually, we group students into *types* with identical programme structures and course selections. All students within the same type share a single set of decision variables, enabling a substantial reduction in model dimensionality while preserving the combinatorial structure of the timetabling problem [1, 6]. The underlying implementation is described later in Section 3.4.

Configurable Rules. To preserve the model's long-term usability, given that school policies may change from year to year, we do not hard-code most rules but instead parameterise them via an external configuration file [7]. This includes both switchable constraints (e.g. whether a lunch break is enforced) and numerical parameters (e.g. maximum daily gap between classes, credit requirements). The school can therefore adapt the model to the latest policy settings without modifying its core structure, improving scalability and reusability.

3.4 Student Aggregation Detailed Strategy

Motivation. In the individual-level formulation, the decision variables $t_{s,c,y,\sigma}$ and $a_{s,c,p,k}$ are both indexed by student, requiring the model to instantiate them for every individual. This dramatically increases the number of variables and constraints, enlarging the search space and degrading computational tractability. We therefore reformulate the model by aggregating students with the same programme and course choices into a common student type. The reformulation greatly reduces model size—a critical factor for solvability [5]—and shortens runtime while preserving the programme-level structure of the problem.

Example: If there are 200 single-honours students belonging to the same parent type, and $G = 30$, then they are partitioned into subgroups of size 30 until the remaining number of students is at most 30. In this example, the parent type is split into 7 subgroups: 6 subgroups of size 30 and 1 subgroup of size 20. All subsequent timetabling decisions are then made at the subgroup level.

Parent-Type Construction and Subgroup Splitting. As described in Section 3.3, we group students with identical programme structures (e.g. same curricula or cohorts) into a parent student type, with N_s denoting the number of students in each parent type.

However, if a parent type is large and mapped to a single decision variable $t_{s,c,y,\sigma}$, two problems arise. First, the model loses flexibility: for a parent type of 50 students, the model must either assign all 50 to a course or none at all, which is unrealistic. Second, assigning a large group to sectioned workshops becomes unnatural, since both course and workshop capacities may not accommodate such a block.

Hence, we further split each parent type into sibling subgroups whenever N_s exceeds the parameter `max_group_size = G`, so that each subgroup contains at most G students.

Moreover, aggregation will not change the logical structure of programme rules, as constraints such as credit bounds and gateway/optional-course requirements are still enforced exactly for each subgroup. While the main modelling effect occurs in those enrolment-related quantities, such as course and sectioned workshop capacities, which are hence evaluated in N_s -weighted subgroup blocks.

Symmetry Breaking on Aggregated Student Types. Splitting a parent type into sibling subgroups naturally introduces symmetry: subgroups share programme rules and course choices, differing only in index. Exchanging the course assignments of two siblings does not affect the resulting timetable, yet the solver treats the two solutions as distinct and explores both. To eliminate this redundant search, we impose an ordering constraint among sibling subgroups of the same parent type, retaining only one canonical representation [8]. This constraint does not alter the feasible region or exclude distinct timetables; its sole effect is to prune equivalent solutions and thereby improve solving

efficiency.

3.5 Model Formulation

The full mathematical formulation is provided in the Appendix (S1–S19 for student constraints, C1–C13 for course constraints). Here we focus on four constraints whose modelling logic is non-trivial or requires further justification.

S8. Per-Year Optional Course Count (Conditional Minimum). A natural approach to bounding the number of optional courses per year would be a simple two-sided inequality $R^{\text{opt_min_yr}} \leq \text{opt}_y \leq R^{\text{opt_max_yr}}$. However, this would force every student to take at least $R^{\text{opt_min_yr}}$ optional courses each year, even though some programmes legitimately allow a student to take zero optional courses in a given year (e.g. when all credits are filled by gateway or outside courses).

To capture this “all-or-nothing” behaviour we introduce an auxiliary Boolean $h_{s,y} \in \{0, 1\}$:

$$h_{s,y} = 1 \iff \underbrace{\sum_{\sigma} \sum_{c \in C^{\text{OPT}}} t_{s,c,y,\sigma}}_{\text{opt}_y} \geq 1 \quad (1)$$

The optional count is then bounded by:

$$h_{s,y} = 1 \implies \text{opt}_y \geq R_s^{\text{opt_min_yr}}, \quad \text{opt}_y \leq R_s^{\text{opt_max_yr}} \quad (2)$$

The effective domain is therefore $\text{opt}_y \in \{0\} \cup [R_s^{\text{opt_min_yr}}, R_s^{\text{opt_max_yr}}]$, which avoids the infeasibility that a simple lower bound would cause while still enforcing a meaningful minimum whenever an optional course is chosen. In CP-SAT this is implemented via the `OnlyEnforceIf` half-reification mechanism.

S17. Maximum Daily Consecutive Gap. To prevent fragmented timetables (e.g. a 9 am lecture followed by nothing until 3 pm), we impose an upper bound gap_{max} on the gap between any two *consecutively attended* periods of the same day.

Step 1 (Attendance indicators). For each student type s , year y , semester σ , week parity ω (odd/even), day d , and period π , we define $\text{att}_{s,y,\sigma,\omega,d,\pi} \in \{0, 1\}$. For a non-sectioned component p at timeslot key $k = (y, \sigma, \omega, d, \pi)$:

$$\text{att}_{s,y,\sigma,\omega,d,\pi} \geq t_{s,c,y,\sigma} + o_{c,p,k} - 1 \quad (3)$$

For a sectioned component:

$$\text{att}_{s,y,\sigma,\omega,d,\pi} \geq a_{s,c,p,k} \quad (4)$$

Since setting $\text{att} = 1$ unnecessarily would tighten the gap constraint, the solver keeps $\text{att} = 0$ whenever no activity is scheduled, without requiring an explicit upper bound.

Step 2 (Gap elimination). For every pair of periods $\pi_i < \pi_j$ on the same day with $\pi_j - \pi_i > \text{gap}_{\text{max}}$:

$$\text{att}_{\pi_i} + \text{att}_{\pi_j} \leq 1 + \sum_{m: \pi_i < \pi_m < \pi_j} \text{att}_{\pi_m} \quad (5)$$

If both periods are attended (LHS = 2), at least one intermediate period must also be attended; by induction this ensures no two consecutively attended periods are separated by more than gap_{max} slots.

S18. Sibling-Subgroup Symmetry Breaking. When detailing the timetabling problem, we further divide each student type, s , into $\lfloor N_s/G \rfloor$ smaller sibling subgroups, $s_0, s_1, \dots, s_{m-1}$, ordered by index. Subgroups sharing the same `parent_type` represent the same underlying student type, which means they only differ by labels, and swapping the course assignments of two such sibling subgroups does not change the actual timetable. However, the solver will still treat them as distinct solutions and

explore them separately, introducing unnecessary symmetry and enlarging the search space. To eliminate these redundant solutions, we fix orderings on sibling subgroups such that only one canonical representation is retained, thereby improving computational efficiency.

As described above, for adjacent siblings, s_i, s_{i+1} for each course c , year y , semester σ , we impose:

$$t_{s_i, c, y, \sigma} \geq t_{s_{i+1}, c, y, \sigma}, \quad (6)$$

which means that if a later subgroup takes a course, then all the earlier subgroups must also take it. It is worth demonstrating that if N_s cannot be evenly divisible, then the subgroups may have slight differences in size, leading to an over-restrictive constraint. Fortunately, such an influence can be neglected under our balanced partitioning mechanism.

O3. Conflict Penalty. To reduce class clashes, we introduce a soft constraint that captures the cases where the same student type is required to attend multiple teaching activities at the same time. For each student type s , course c , teaching component p (e.g. a lecture or workshop), and timeslot k , we define an attendance indicator $z_{s, c, p, k}^{\text{att}} \in \{0, 1\}$, which equals 1 if the student type s is required to attend component (c, p) at timeslot k . Generally speaking, this attendance indicator depends on the component type:

$$z_{s, c, p, k}^{\text{att}} = \begin{cases} a_{s, c, p, k}, & \text{for sectioned component, e.g. workshops} \\ t_{s, c, y, \sigma} \wedge o_{c, p, k}, & \text{otherwise} \end{cases} \quad (7)$$

A conflict indicator is also defined for any pair of activities i and j , for student type s , at timeslot b :

$$z_{s, b, i, j}^{\text{cf}} = z_i^{\text{att}} \wedge z_j^{\text{att}}, \quad (8)$$

which equals 1 when student type s needs to simultaneously attend both activities i and j in the same timeslot b . In the objective, only pairs with overlapping week patterns are counted. Hence, the corresponding penalty is computed as follows:

$$\text{Penalty}_{s, b, i, j} = \text{sev}(i, j) \cdot N_s \cdot z_{s, b, i, j}^{\text{cf}}, \quad (9)$$

where N_s is the size of student type s , and $\text{sev}(i, j)$ is the severity weight of the clash types, the detailed settings are listed in Table 1.

To sum up, the total conflict penalty is therefore:

$$\text{Penalty}_{\text{conflict}} = \sum_s \sum_b \sum_{\substack{(i, j): \\ \text{overlap}(w_i, w_j)}} \text{sev}(i, j) \cdot N_s \cdot z_{s, b, i, j}^{\text{cf}} \quad (10)$$

Here, $\text{overlap}(w_1, w_2) = \text{TRUE}$ indicates that activities i and j occur in overlapping weeks under their week patterns. In particular, a weekly activity overlaps with any pattern, while an odd-week activity overlaps only with odd weeks, and similarly for even weeks.

3.6 Objective Function

The model minimises a weighted combination of penalties minus bonuses (the full expanded formulation with all index sets is given in Appendix 7):

$$\min \quad \underbrace{w_1 \sum_{s, c} N_s \cdot t_{s, c, y, \sigma}}_{\text{O1: desire bonus (negative)}} + \underbrace{w_2 \sum_c \alpha_{c, y, \sigma}}_{\text{O2: activation penalty}} + \underbrace{\sum_{\ell} w_{\ell} \cdot N_s \cdot z^{\text{cf}}}_{\text{O3: conflict penalties}} + \underbrace{w_4 \sum_k \text{overflow}_k}_{\text{O4: room overflow}} \quad (11)$$

Component	Description	Weight
O1	Desired-course bonus	−8 per student
O2	Course activation penalty	+2 per course
O3a	Lecture–Lecture conflict	+18 per student
O3b	Lecture–Workshop conflict	+6 per student
O3c	Workshop–Workshop conflict	+2 per student
O4	Concurrent open classes above soft limit	+1 per slot

Table 1: Objective weights (default configuration).

The severity hierarchy ($O3a \gg O3b \gg O3c$) reflects that lecture–lecture clashes are the most disruptive, as lectures are typically non-repeatable, whereas workshops often have alternative sections available.

4 Experimental Setup

4.1 Data and Instance Construction

To assess the performance of the proposed model, we construct a baseline dataset, `report_v0`, based on the current DPT curriculum data. It contains **56 courses**, comprising **8 gateway courses**, **4 optional courses**, and **40 outside courses**, each worth **20 credits**.

Aligning with the scenario setting, all teaching components of outside courses are fixed in advance and treated as exogenous inputs rather than optimisation variables. The model therefore optimises the School of Mathematics timetable subject to these pre-existing external course commitments. Figure 1 below is an example of the merged Semester 1 timetable for all outside courses (Semester 2 timetable can be found in Appendix).

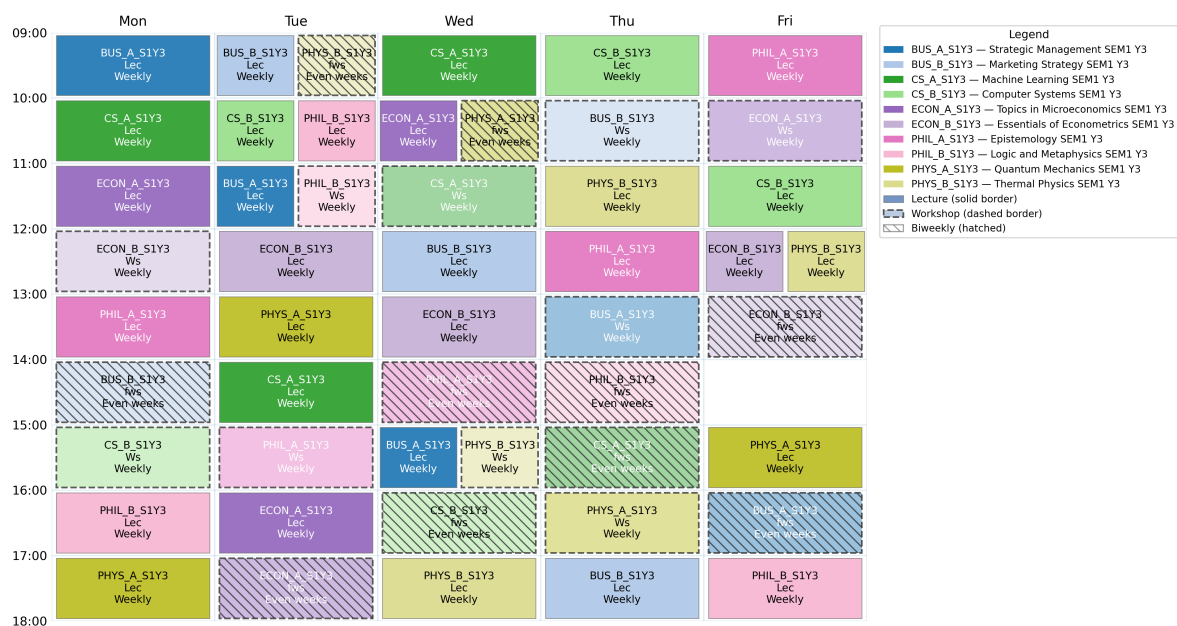


Figure 1: Merged Semester 1 timetable of non-Mathematics courses for Year 3 joint-programme students.

Although external timetables are fixed and not optimised, they are evenly distributed across time slots, with minimal clustering (Figure 1). This ensures that the remaining feasible slots for Mathematics courses are balanced and not systematically biased.

The dataset also includes 152 teaching components, consisting of lectures, weekly workshops, and fortnightly workshops. In addition, students are aggregated into **56 parent student types** from a total of **475 students** across 6 programmes (1 single-honours and 5 joint programmes). After subgroup splitting (at most 30 students per subgroup), the single-honours type expands into 7 subtypes, yielding **62 subtypes** in total. These student types are defined either manually, based on historical

course-choice patterns, or generated randomly to represent additional demand variation. Time is discretised into hourly timeslots, with each timeslot representing one hour of teaching time. A brief breakdown is listed in Table 2, while more detailed data can be found here.

Programme	Types	Students	Notes
Single-honours Maths	1 (7 [†])	200	Full 8-gateway requirement
Maths–Physics	11	55	40 outside credits/year
Maths–Economics	11	55	40 outside credits/year
Maths–Business	11	55	40 outside credits/year
Maths–Computer Sci.	11	55	40 outside credits/year
Maths–Philosophy	11	55	40 outside credits/year
Total	56 (62[†])	475	

Table 2: Student distribution by programme (report_v0).

[†] After subgroup splitting (max 30 students per subgroup); the single-honours parent type expands into 7 subtypes.

Building on this, each scenario below generates and optimises an 11-week semester timetable to evaluate how well the model can produce a timetable that meets the desired quality criteria. Note that we only schedule courses for the School of Mathematics; outside-course schedules are fixed in advance and are not subject to optimisation.

4.2 Progressive Scenario Refinement

Building on the baseline scenario, V0, the optimisation is developed in three main stages, with each scenario inheriting the settings of the previous one. In particular, the first two stages (V1–V3 and V4–V6) mainly focused on refining timetable through compactness and additional rules, respectively, while the last stage (V7 onward) further explores the boundary of model feasibility.

Baseline Scenario: V0. To establish a feasibility benchmark, the baseline scenario adopts a relatively permissive set of scheduling rules. In particular, each course may occupy at most two teaching slots per day (`course_max_slots_per_day` = 2), and the maximum daily gap between two consecutive attended classes is limited to four hours (`student_max_daily_gap` = 4) for all students. The latter aligns with the current timetabling policy, while the former prevents individual courses from being overly concentrated within a single day.

Compactness and Sequencing Refinements: V1-V3.

- **V1:** Two additional rules are added: 1) for each student, the maximum number of class periods they need to attend per day is 4, which aims to reduce students' excessive daily workload; 2) workshops are enforced to be scheduled after lectures (`workshop_after_lecture` = TRUE), in order to give students the opportunity to review the delivered content, thereby improving pedagogical coherence and efficiency.
- **V2:** Timetable compactness is further strengthened by reducing the maximum daily gap between two consecutive attended classes from 4 to 3. Moreover, the number of concurrently scheduled courses in any timeslot is capped at 4 from the perspective of the school (`max_concurrent_courses_per_timeslot` = 4), which helps to avoid excessive clustering of teaching activities and improve the balance of course spread.
- **V3:** Workshops are allowed to be scheduled in parallel in multiple sessions, which is achieved by resetting the parameter `sections_max` in `components.csv`. This change will not only provide students with greater scheduling flexibility, but will also reduce the likelihood of class clashes and support a more compact timetable.

Additional timetable-quality rules: V4-V6. In addition to the base scenarios with 56 parent types, we also create **more-choice (MC) variants** for each of V4–V6 (denoted V4-MC through V6-MC). These variants retain the same rule set as their base version but increase the number of parent types

from 56 to 76 (15 types per joint programme instead of 11), providing a richer set of student demands to test whether the corresponding rule set remains feasible under greater demand diversity.

- **V4:** Given the compactness of the existing timetable, a considerable proportion of students would have to sacrifice their lunch break if they want to attend all courses. We therefore introduce a new timetabling rule that no class is scheduled between 12:00 and 13:00 by setting `lunch_break_no_class = TRUE`.
- **V5:** One more hard constraint is added by setting `student_max_consecutive_slots = 2`, which forbids model from scheduling students for more than two consecutive hours of classes and requires at least a one-hour break afterwards. This new rule intends to avoid fatigue caused by excessively long continuous study periods. Also, more penalty (with `weight = 2`, `threshold = 7`) is imposed on classes scheduled after 16:00 through a soft constraint, `late_slot_penalty`, which encourages teaching activities to be concentrated within the daytime.
- **V6:** Since feasibility has been maintained throughout the previous scenarios, in this version, we add several quality-oriented rules from both the school and student perspectives, as listed below:
 - **School Perspective:** 1) Given Math gateway courses have both weekly and fortnightly workshops, we now prohibit the model to schedule these two workshop types in the same timeslot (`ws_weekly_fortnightly_no_overlap = TRUE`), so as to further reduce potential clashes and improve timetable flexibility. 2) No matter the kinds of workshop, the number of parallel sessions of the same workshop type in a single timeslot is capped at two (`max_same_type_ws_per_timeslot = 2`) to prevent an excessive concentration of teaching activities in one slot.
 - **Student Perspective:** For single-honours Mathematics students, no Mathematics course is scheduled at 9:00 (`no_first_period_prefixes = ["MATH"]`) or on Friday afternoon (`no_Friday_afternoon_prefixes = ["MATH"]`); in order to give students more flexibility for independent study and extracurricular activities.

Model Robustness Examination. After simulating all previous scenarios, we find that they all remain conflict-free throughout. Hence, from V7 onwards, the main objective shifts to identify the limit of timetable flexibility. To do this, we increase the number of student types while making minor adjustments to several timetable-quality rules.

- **V7 (Stress test):** The number of student types per joint programme is increased from 11 to 15, resulting in 76 parent types in total. All V6 rules are retained (including the lunch-break restriction and the full-week 9:00 ban). This scenario tests whether the V6 rule set can handle the increased demand diversity.
- **V7.5 (Extended population):** To disentangle the effects of rule strength from those of instance size, V7.5 uses 86 parent types and 525 students—more than V7—but adopts the same relaxed rule set as V7-Final (no lunch break, 9:00 ban restricted to Mon/Wed/Fri). This scenario isolates the role of student-type composition versus constraint tightness.
- **V7-Final (Selective relaxation):** To recover feasibility, three rules are relaxed: 1) the lunch-break restriction is removed; 2) the penalty on late classes after 16:00 is dropped; and 3) the previous prohibition on 9:00 Mathematics classes throughout the whole week is replaced by a restriction applying only on Monday, Wednesday, and Friday (`no_first_period_days = [0, 2, 4]`).
- **V8 (Maximum expansion):** Student types are further increased to 100 parent types (approximately 20 per joint programme). In addition, if two lectures from the same course are scheduled on the same day, then they must be consecutive, forming a two-hour lecture block.
- **V9:** Same rules as V8 but with 76 parent types (same as V7-Final). This isolates the effect of the consecutive-lecture constraint from the effect of type expansion.

A detailed summary of the differences across scenarios is listed in Table 3.

Version	Instance	Rules												
	Student Types	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13
V0	56	4												
V1	56	4	4	✓										
V2	56	3	4	✓	4									
V3	56	3	4	✓	4	✓								
V4	56	3	4	✓	4	✓	✓							
V5	56	3	4	✓	4	✓	✓	2	✓					
V6	56	5	4	✓	4	✓	✓	2	✓	✓	✓	✓	2	
V7	76	5	4	✓	4	✓	✓	2	✓	✓	✓	✓	2	
V7.5	86	5	4	✓	4	✓	×	2	×	✓	M,W,F	✓	2	
V7-Final	76	5	4	✓	4	✓	×	2	×	✓	M,W,F	✓	2	
V8	100	5	4	✓	4	✓	×	2	×	✓	M,W,F	✓	2	✓
V9	76	5	4	✓	4	✓	×	2	×	✓	M,W,F	✓	2	✓

Table 3: Summary of scenario evolution. Blank cells indicate that the corresponding rule has not been enforced in that scenario; “M, W, F” denotes Monday, Wednesday, and Friday; × indicates a rule that was previously active but is now disabled. More-choice variants (V4-MC through V6-MC) use the same rules as their base version but with 76 parent types.

Rule codes:

- R1: Maximum daily gap (in hours) between two consecutive classes.
- R2: Maximum daily class load.
- R3: Workshops must be scheduled after lectures.
- R4: Maximum number of concurrent courses per timeslot for school.
- R5: Allowed parallel session for workshops.
- R6: Lunch break.
- R7: Maximum consecutive teaching periods (in hours).
- R8: Penalty on late classes after 16 : 00.
- R9: No overlap between weekly and fortnightly workshops of the same course.
- R10: No Mathematics classes at 9 : 00.
- R11: No Friday Mathematics classes from 12 : 00 onwards.
- R12: Maximum number of same-type workshop sessions per timeslot.
- R13: Consecutive two-hour lecture if two lectures are scheduled on the same day.

5 Results

5.1 Overall Performance Comparison

We compare three representative scenarios introduced in Section 4: V0 (baseline), V7-Final (final selected model), and V9 (constrained variant). The baseline model V0 confirms that the proposed curriculum can be feasibly scheduled, as it produces a fully feasible and conflict-free solution in which all degree requirements are satisfied. Building on this, V7-Final retains the same conflict-free feasibility while incorporating substantially stronger timetable-quality constraints, including compactness, workload limits, and sequencing rules, and is therefore adopted as the final timetable used for evaluation and analysis in this study. This demonstrates that feasibility is robust and can be preserved under more realistic and operationally desirable conditions. In contrast, V9 introduces an additional same-day lecture adjacency constraint, which significantly reduces scheduling flexibility. As a result, the number of conflicts increases sharply to 211, affecting 28.8% of students, indicating a clear deterioration in solution quality.

A summary comparison is given in Table 4.

Metric	V0	V7-Final	V9
Feasible	Yes	Yes	Yes (incumbent)
Objective	128	128	1 438
Bound	128	128	206
Optimality gap	0%	0%	85.7%
Conflict count	0	0	211
% students affected	0%	0%	28.8%
Timetable quality	Basic	Best	Poor
Runtime	~5 min	~1.5 h	2 h (limit)

Table 4: Comparison of representative scenarios.

These results highlight a fundamental distinction between course-level feasibility and student-level choice flexibility. While all scenarios satisfy course requirements, only V7-Final preserves both feasibility and a high level of accessibility for students, particularly for joint-programme students constrained by external timetables.

From a computational perspective, clear differences also emerge across scenarios. The baseline model V0 is solved within approximately 5 minutes, reflecting its relatively low structural complexity. V7-Final requires around 1.5 hours to reach optimality due to the inclusion of additional timetable-quality constraints, yet remains computationally tractable. In contrast, V9 reaches the 2-hour time limit without closing the optimality gap (85.7%), indicating a substantial increase in computational burden without corresponding improvements in solution quality.

Overall, the results demonstrate that achieving a high-quality timetable requires a careful balance between flexibility, constraint strength, and computational tractability, with V7-Final representing the most effective and practically viable configuration.

5.2 Trade-off Analysis: Choice Flexibility, External Constraints, and Timetable Quality

The comparison between V7-Final and V9 provides a clear and quantitative illustration of the trade-off between timetable flexibility and structural regularity. Under V7-Final, the model achieves a fully conflict-free solution, indicating that sufficient temporal flexibility is retained to accommodate both Mathematics courses and externally fixed partner-department courses without overlap. In contrast, introducing the same-day lecture adjacency constraint in V9 significantly reduces the number of feasible lecture placements. In particular, enforcing consecutive lecture blocks reduces the availability of isolated time slots, which are often needed to accommodate workshops.

As a result, V9 produces a substantial number of timetable conflicts: In total, 211 conflicts are observed. Among these, 192 are lecture–workshop conflicts and 19 are workshop–workshop conflicts. These conflicts affect 137 students, corresponding to approximately 28.8% of the total population. Notably, all affected students belong to joint-programme cohorts, while single-honours students remain unaffected, as shown in the programme-level breakdown (Table 5).

Programme	Conflicts	% Affected
Single-honours	0	0.0%
Joint CS	84	83.6%
Joint Economics	49	49.1%
Joint Physics	44	65.5%
Joint Business	31	45.5%
Joint Philosophy	3	5.5%

Table 5: V9 per-programme conflict breakdown.

This asymmetric impact reflects the role of external constraints, as joint-programme students must accommodate a large number of externally fixed courses, leaving limited flexibility for Mathematics timetabling. From an optimisation perspective, this deterioration is also evident in the objective values: while V7-Final is solved to optimality with an objective value of 128, the best solution found for V9

has a significantly worse objective value of 1438 with an optimality gap of 85.7% under the same computational budget.

Taken together, these results demonstrate that enforcing additional structural regularity at the course level can unintentionally reduce global timetable flexibility, leading to disproportionately large increases in conflicts, particularly for students subject to external constraints.

5.3 Quality of the Final Selected Timetable

Given that V7-Final preserves conflict-free feasibility while avoiding the deterioration observed in V9, we now examine the qualitative strengths of the final timetable in terms of feasibility, timetable quality, and practical usability. In particular, we evaluate its performance from both the student and planner perspectives, focusing on key operationally desirable properties.

Student Perspective: From a student perspective, the timetable produced under V7-Final offers a well-balanced and manageable schedule at both the daily and weekly levels.

At the daily level, workload is effectively controlled, with students attending at most four sessions per day, which ensures that study demands remain manageable. In addition, teaching is organised into short blocks, typically no more than two consecutive sessions, with breaks introduced between them. This structure reduces fatigue and supports sustained engagement throughout the day.

Furthermore, gaps between consecutive classes are limited, typically not exceeding two hours. This avoids prolonged idle periods on campus and improves overall time efficiency, making the timetable more practical for students. Compared to less structured scenarios, this represents a clear improvement in usability.

At the weekly level, the timetable is designed to be student-friendly and well-distributed. Early morning classes are largely avoided, with very few sessions scheduled at 9:00a.m. In addition, Friday afternoons are kept free, allowing students to finish earlier at the end of the week. This creates a more flexible weekly structure, supporting both academic and extracurricular activities.

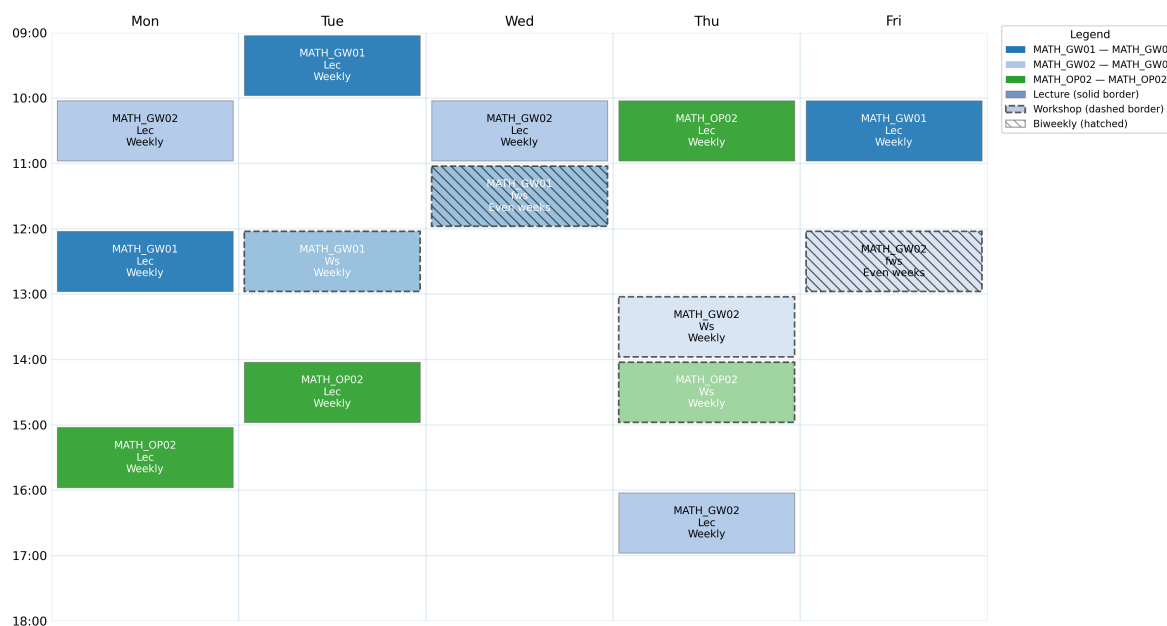


Figure 2: Individual timetable for a single-honours Mathematics student (type 0, subgroup 3) in Year 3 Semester 1 under V7-Final. Daily workload does not exceed 4 sessions, consecutive blocks are limited to 2 hours, and no class is scheduled at 9:00 on Mon/Wed/Fri.

Overall, these features show that the timetable is not only conflict-free, but also aligned with realistic student needs in terms of workload balance, time efficiency, and weekly flexibility (see Appendix F for individual student timetable examples, Figures 2–7).

Planner Perspective: From a planning perspective, the timetable demonstrates efficient use of teaching resources while maintaining a high-quality structure. Most teaching activities are scheduled within core hours (approximately 9:00–16:00), with only a small number of sessions occurring later in

the day: 16 out of 18 lectures fall within this window, and only 2 are scheduled after 16:00. This ensures that teaching is concentrated within standard working hours and improves resource utilisation.

In addition, the number of concurrent sessions per timeslot is controlled, avoiding excessive clustering of courses. In most timeslots, only one or two sessions are scheduled simultaneously, which reduces pressure on shared resources and improves timetable clarity.

Finally, workshop allocation is carefully managed through controlled parallel sessions, balancing accessibility and capacity constraints. This demonstrates that resources are utilised efficiently without compromising timetable quality, in contrast to more permissive settings such as V0.

Timetable Illustration: Figure 3 provides a visual illustration of the timetable structure for Year 3 Mathematics in Semester 1. The figure highlights the even distribution of teaching activities across the week, the absence of significant clustering, and the presence of regular breaks between sessions.

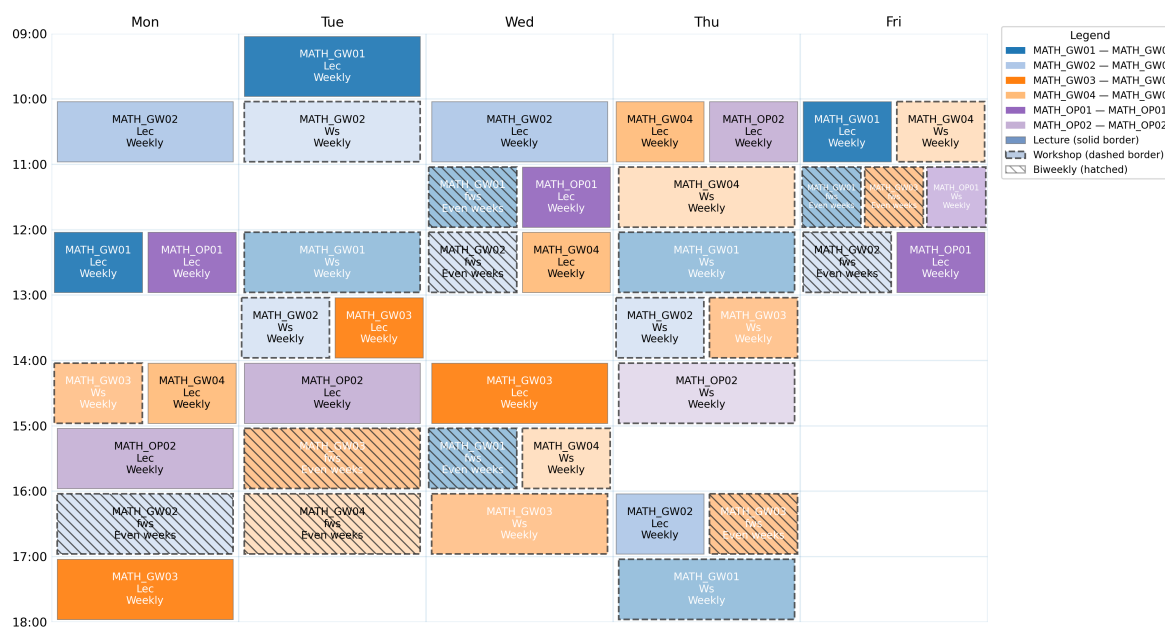


Figure 3: Year 3 Mathematics timetable in Semester 1 (V7-Final).

As shown in the figure, teaching sessions are distributed across the week with a clear concentration in core hours and visible breaks between teaching blocks. This structured pattern avoids both excessive clustering and prolonged idle periods, providing visual evidence of a well-balanced timetable.

Taken together, these results show that high-quality timetable structures can be achieved without sacrificing feasibility, provided that sufficient temporal flexibility is preserved.

5.4 Key Insights for Curriculum Design

The experimental results provide several important insights into the design of flexible and feasible timetables.

1. **Conflict-free timetabling is achievable at scale.** The model is able to generate fully conflict-free timetables (0 conflicts) under realistic constraints, even for a large instance with 475 students and multiple programme structures. This demonstrates that the proposed curriculum is operationally feasible.
2. **Feasibility does not imply full flexibility.** Although all courses can be scheduled without conflicts, not all course combinations are equally accessible, particularly for joint-programme students who must satisfy additional external requirements. This highlights a gap between theoretical feasibility and practical accessibility.
3. **External constraints are the main limiting factor.** The results indicate that fixed outside-course timetables significantly restrict scheduling flexibility for joint students. In many cases, these external constraints (not the Mathematics timetable itself) are the primary source of reduced choice.
4. **Over-constraining the model can severely degrade performance.** The comparison between V7-Final and V9 shows that introducing additional constraints (e.g. enforcing consecutive lec-

tures) leads to a sharp increase in conflicts (from 0 to 211) and affects nearly 30% of students. This demonstrates that even well-motivated constraints can have unintended negative effects if they excessively restrict flexibility.

5. **Structured pathways offer a practical compromise.** Given that unrestricted choice is difficult to achieve for all students, particularly under external constraints, grouping courses into structured pathways can provide a more balanced solution. Such pathways reduce conflicts while still preserving meaningful flexibility.

Overall, the results demonstrate that achieving high-quality, conflict-free timetables requires a careful balance between flexibility and constraint strength, with V7-Final representing a practical and robust solution.

6 Critical Analysis

6.1 Operational Implications

The zero-conflict optimum achieved on the 475-student baseline (V0) demonstrates that the proposed Y3/Y4 Mathematics redesign *is* timetable-feasible under realistic conditions. The scenario experiments further show that many practically desirable constraints, such as daily load caps, lecture spreading, and workshop sequencing, can be adopted at no cost to solution quality, while certain restrictions, in particular the lunch-break ban and the universal 9:00 prohibition, must be applied cautiously at higher levels of student-type diversity. The model can be deployed as a planning tool: administrators input updated course/student data, and the solver produces an optimised timetable within minutes to hours depending on constraint complexity.

6.2 Key Trade-Offs

The progressive scenario experiment reveals three principal trade-offs:

- **Quality rules vs. temporal freedom.** At moderate type counts (56 types), all quality-of-life rules (V1–V6) are non-binding and produce zero conflicts. However, at 76 types, the lunch-break restriction and universal 9:00 ban become the primary drivers of infeasibility (V7: 621 conflicts), requiring selective relaxation to restore the optimum (V7-Final: 0 conflicts).
- **Student-type diversity vs. constraint tightness.** Increasing the number of student types does not uniformly degrade solutions. V7.5 (86 types, 525 students) adopts the same relaxed rule set as V7-Final (no lunch-break restriction, 9:00 ban on Mon/Wed/Fri only), achieving zero conflicts despite a larger population than V7. In contrast, V7 (76 types, 475 students) retains the stricter V6 rules and incurs 621 conflicts. Problem difficulty is therefore governed by the interaction between demand diversity and rule tightness, not by instance size alone.
- **Structural regularity vs. scheduling flexibility.** The consecutive two-hour lecture rule (V9) reduces lecture placement options, indirectly triggering 211 workshop conflicts for joint-programme students. This effect is data-sensitive: V8 (100 types, same rule) achieves zero conflicts, demonstrating that the impact depends on the specific student-type composition.

6.3 Strengths

1. **Optimality Guarantees.** Unlike heuristic methods, CP-SAT is able to provide an optimal solution, or at least a solution with a quantified optimality gap [3].
2. **High Computational Efficiency and Scalability.** Aggregating students into student types and introducing symmetry breaking significantly improve solving speed and scalability [9].
3. **Configurable and Reusable Rule Design.** Specific programme and scheduling rules can be adjusted to better fit different scenarios.
4. **Tunable Aggregation Granularity.** Although we choose $G = 30$ in our experiments, which is relatively large, it can be adjusted to other values to better fit instances of different scales.
5. **Compatible Soft Constraints.** By modelling students' preferences and conflict penalties as soft constraints, the model is able to produce a more student-friendly timetable, not only reducing the likelihood of class clashes, but also trying to satisfy students' preferences simultaneously.

6.4 Limitations

1. **Capacity constraints are not independent.** For both course capacities and sectioned-workshop capacities, feasibility depends not only on the capacity values themselves, but also on their interaction with students' credit requirements and other programme constraints. If a capacity limit is set below the demand implied by these constraints, the model may be unable to satisfy both the

opening requirement and the upper-capacity bound simultaneously, which can lead to infeasibility.

2. **Limited Coverage of Student Types.** The student types used in our experiments are constructed either from historical course-choice patterns or generated randomly. Although we test instances with up to 100 types, real course choices may still not be fully covered.
3. **Idealised Student-Type Sizes.** The assumed number of students in each type may not fully match the real distribution in practice. In this study, these sizes are simplified in order to keep the computation manageable.
4. **Ideal Workshop Capacities.** At present, we assume that parallel workshop sessions of the same course share the same capacity limit. In reality, however, workshop capacities may depend on the sizes of the rooms in which they are scheduled. Therefore, even when several parallel workshop sessions are opened for the same course, their actual capacities may differ.

6.5 Recommendations for the School

1. **Proceed with the 8-gateway, 4-optional structure:** it is feasible and clash-free for all programmes under the V7-Final rule set with up to 76 student types.
2. **Adopt the V7-Final rule set as the default configuration:** this includes daily load caps, workshop sequencing, compactness constraints, and the selective 9:00 restriction (Mon/Wed/Fri only), all of which are satisfied with zero conflicts.
3. **Be cautious with the lunch-break ban and universal 9:00 prohibition:** these are the primary feasibility bottlenecks at higher student-type diversity (V7: 621 conflicts) and should only be imposed if the type count is sufficiently low.
4. **Avoid the consecutive two-hour lecture requirement:** this constraint is data-sensitive and can cause substantial conflicts for joint-programme students (V9: 211 conflicts), while providing limited pedagogical benefit.
5. **Integrate the CP-SAT optimizer** into the annual timetabling workflow as a decision-support tool.

6.6 Further Improvements

1. **More Granular Modelling.** Although aggregating students into sibling subgroups with at most $G = 30$ students in each subgroup improves solving efficiency, some loss of individual-level heterogeneity is inevitable. Therefore, future work could consider using smaller subgroup sizes to provide a more fine-grained representation of student demand.
2. **Further Robustness Analysis.** Current results are obtained under a deterministic setting, and therefore, a natural direction for future work is to examine the robustness of the model under uncertainty. For instance, future work could consider fluctuations in student demand and preferences, changes in the numbers of students and teaching staff, or the closure of some courses due to insufficient enrolment.

7 Conclusion

This project developed and validated a CP-SAT constraint-programming model for the proposed Y3/Y4 honours Mathematics curriculum redesign at the University of Edinburgh, and systematically evaluated a series of progressively tightened scenario variants against a 475-student, 62-type baseline.

Q1 (Single-honours flexibility): The model confirms that single-honours students can access all 8 gateway courses with genuine optional-course choice and zero timetable conflicts across all tested scenarios, including the most constrained variants (V7-Final, V9).

Q2 (Joint-programme accessibility): Joint-programme students achieve zero conflicts under all scenarios up to and including V7-Final (76 student types with quality rules). However, they are disproportionately affected when constraints become overly restrictive: in V9 (consecutive-lecture rule), all 211 conflicts fall on joint-programme cohorts, reflecting the limited scheduling flexibility imposed by externally fixed partner-department timetables.

Q3 (Pathway/cohort improvement): Structured student-type aggregation with predefined desired-course bundles enables conflict-free scheduling at scale. The progressive scenario analysis shows that quality-of-life rules (daily load caps, workshop sequencing, compactness) are costless at moderate type counts, but that certain restrictions—notably the lunch-break ban and universal 9:00 prohibition—become the primary drivers of infeasibility at higher diversity (V7: 621 conflicts), requiring selective

relaxation (V7-Final: 0 conflicts).

Main recommendation: The proposed redesign is timetable-feasible and should proceed. We recommend adopting the V7-Final rule set as the default configuration, which preserves all key quality-of-life constraints while maintaining zero conflicts, and caution against imposing the consecutive-lecture requirement (V9), which is data-sensitive and can severely degrade joint-programme accessibility. Full per-scenario details are provided in Appendix G.

References

- [1] Andrea Bettinelli et al. “An overview of curriculum-based course timetabling”. en. In: *TOP* 23.2 (July 2015), pp. 313–349. ISSN: 1863-8279. DOI: 10.1007/s11750-015-0366-z. URL: <https://doi.org/10.1007/s11750-015-0366-z> (visited on 03/01/2026).
- [2] Rachelle A. Oude Vrielink et al. “Practices in timetabling in higher education institutions: a systematic review”. en. In: *Annals of Operations Research* 271.2 (2019), pp. 547–582. ISSN: 0254-5330. DOI: 10.1007/s10479-017-2688-8. URL: <https://doi.org/10.1007/s10479-017-2688-8>.
- [3] Google LLC. *OR-Tools – Google Optimization Tools*. Version 9.10. Available at <https://developers.google.com/optimization>. 2024. URL: <https://developers.google.com/optimization>.
- [4] Éber Júnio Borges Moreira and Sergio Antonio Andrade De Freitas. “A CP-SAT Approach for Academic Resource Timetabling in Higher Education Institutions: A Case Study at a Major Public University”. In: *2024 21st International Conference on Information Technology Based Higher Education and Training (ITHET)*. ISSN: 2473-2060. Nov. 2024, pp. 1–8. DOI: 10.1109/ITHET61869.2024.10837617. URL: <https://ieeexplore.ieee.org/document/10837617> (visited on 03/31/2026).
- [5] Tobias Achterberg. “SCIP: solving constraint integer programs”. en. In: *Mathematical Programming Computation* 1.1 (July 2009), pp. 1–41. ISSN: 1867-2957. DOI: 10.1007/s12532-008-0001-1. URL: <https://doi.org/10.1007/s12532-008-0001-1> (visited on 03/01/2026).
- [6] Michele Conforti, Gérard Cornuéjols, and Giacomo Zambelli. *Integer Programming*. en. Vol. 271. Graduate Texts in Mathematics. Cham: Springer International Publishing, 2014. ISBN: 978-3-319-11007-3 978-3-319-11008-0. DOI: 10.1007/978-3-319-11008-0. URL: <https://link.springer.com/10.1007/978-3-319-11008-0> (visited on 03/01/2026).
- [7] Hilary P. Williams. *Model building in mathematical programming*. eng. 5. ed. Chichester, West Sussex: Wiley, 2013. ISBN: 978-1-118-44333-0.
- [8] Krzysztof R. Apt, ed. *Principles of constraint programming*. eng. Cambridge New York: Cambridge University Press, 2003. ISBN: 978-0-511-61532-0 978-0-511-06249-0 978-0-511-05616-1 978-0-511-07095-2.
- [9] Tobias Achterberg et al. “Presolve Reductions in Mixed Integer Programming”. In: *INFORMS Journal on Computing* 32.2 (2020), pp. 473–506. ISSN: 1091-9856. DOI: 10.1287/ijoc.2018.0857. URL: <https://doi.org/10.1287/ijoc.2018.0857>.

Appendix

A. Common Notation — Sets and Parameters

A.1 Sets and Indices

Symbol	Description
$\mathcal{S}$	Set of student types (post-aggregation). Each type groups students sharing the same programme, year, compulsory courses, and desired courses.
$\mathcal{C}$	Set of all courses; partitioned into $\mathcal{C}^{\text{GW}}$ (gateway), $\mathcal{C}^{\text{OPT}}$ (optional), $\mathcal{C}^{\text{OUT}}$ (outside).
$\mathcal{P}_c$	Set of teaching components for course c (e.g. lectures, workshops).
$\mathcal{P}_c^{\text{sec}} \subseteq \mathcal{P}_c$	Sectioned components of course c (workshop-type or <code>sections_max</code> > 1).
$\mathcal{K}$	Set of timeslot keys. Each $k \in \mathcal{K}$ is a tuple $(y_k, \sigma_k, \omega_k, d_k, \pi_k)$: year, semester, week-pattern, day, period.
$\mathcal{K}_{c,p}$	Allowed timeslot keys for component p of course c (filtered by allowed days, periods, years, semesters, and forbidden keys).
$\mathcal{Y}_s$	Set of academic years in student type s 's planning horizon.
Σ_s	Set of semesters for student type s .
$\mathcal{Y}, \Sigma$	Global sets of years / semesters derived from timeslot keys.
$\mathcal{D}$	Set of weekdays.
Π	Set of periods (hourly timeslots within a day).
$\mathcal{W} = \{A, O, E\}$	Week patterns: ALL (every week), ODD (odd weeks only), EVEN (even weeks only).
$\text{base}(k)$	The base slot of key k , obtained by collapsing the week-pattern dimension: $(y_k, \sigma_k, d_k, \pi_k)$.

Table 6: Sets and indices used throughout the formulation.

A.2 Parameters

Student-level degree rules (per student type s ; sourced from programme-specific rule sets):

Symbol	Description
N_s	Number of students in type s
prog_s	Programme of student type s (e.g. SINGLE, JOINT_PHYS, ...)
comp_s	Set of compulsory course IDs for type s
des_s	Set of desired course IDs for type s
<i>Per-semester bounds</i>	
$R_s^{\text{gw_min_sem}}$	Min number of gateway courses per semester
$R_s^{\text{gw_max_sem}}$	Max number of gateway courses per semester
$R_s^{\text{opt_min_sem}}$	Min number of optional courses per semester
$R_s^{\text{opt_max_sem}}$	Max number of optional courses per semester
$R_s^{\text{crs_min_sem}}$	Min total courses (all types) per semester
$R_s^{\text{crs_max_sem}}$	Max total courses (all types) per semester
$R_s^{\text{cr_min_sem}}$	Min total credits per semester
$R_s^{\text{cr_max_sem}}$	Max total credits per semester
<i>Per-year bounds</i>	
$R_s^{\text{gw_min_yr}}$	Min number of gateway courses per year
$R_s^{\text{gw_max_yr}}$	Max number of gateway courses per year
$R_s^{\text{opt_min_yr}}$	Min optional courses per year (conditional; see S8)
$R_s^{\text{opt_max_yr}}$	Max optional courses per year
$R_s^{\text{crs_min_yr}}$	Min total courses per year
$R_s^{\text{crs_max_yr}}$	Max total courses per year
$R_s^{\text{cr_year}}$	Total credits required per year (equality constraint)
$R_s^{\text{out_cr_year}}$	Outside (non-maths) credits required per year (joint only)
<i>Cross-year and balance rules</i>	
$R_s^{\text{gw_total}}$	Total gateway courses required across all years
R_s^{balanced}	Whether balanced credits across semesters is enforced

Table 7: Student-level degree-rule parameters.

Course-level parameters:

Symbol	Description
cr_c	Credits awarded for course c
$\overline{E}_c$	Max enrollment capacity of course c
$\underline{E}_c$	Min enrollment required for course c to run
n_p^w	Number of meetings per week for component p
$\underline{S}_p$	Min number of parallel sections for component p
$\overline{S}_p$	Max number of parallel sections for component p
$\underline{Q}_p$	Min student capacity per section for component p
$\overline{Q}_p$	Max student capacity per section for component p

Table 8: Course and component parameters.

Global / timetable-structure parameters:

Symbol	Description
G_{σ}^{GW}	Number of gateway courses to offer per semester σ
G_{σ}^{OPT}	Number of optional courses to offer per semester σ
G	Max students per aggregated subgroup (groups larger than G are split into $\lceil N/G \rceil$ subgroups)
$\text{gap}_{\max}$	Student max daily consecutive gap (in periods)
$\text{slots}_{\max}$	Student max attended periods per day
$\text{day}_{\max}$	Course max total teaching slots per day
$\text{lec_day}_{\max}$	Course max lecture periods per day
ws_after_lec	Whether workshops must be scheduled on a day $\geq$ the earliest lecture day
$\text{conc_slot}_{\max}$	Max concurrent open course sections at the same timeslot across all courses

Table 9: Global timetable and scheduling parameters.

Notation convention. All sums below are taken only over indices for which the corresponding variable was created. Domain restrictions imposed during variable creation (banned courses, allowed years/semesters, programme restrictions) are implicitly applied.

B. Decision Variables

All decision variables are binary (BoolVar). Within an aggregated type, all N_s students share an identical schedule.

Variable	Domain	Meaning
$t_{s,c,y,\sigma}$	$\{0, 1\}$	Student type s takes course c in year y , semester σ
$o_{c,p,k}$	$\{0, 1\}$	Component p of course c is open (scheduled) at timeslot key k
$a_{s,c,p,k}$	$\{0, 1\}$	Type s is assigned to section (c, p, k) ; only for sectioned $p \in \mathcal{P}_c^{\text{sec}}$
$\alpha_{c,y,\sigma}$	$\{0, 1\}$	Course c is offered (active) in year y , semester σ

Table 10: Decision variables.

B'. Auxiliary Variables

The following intermediate variables are created inline by specific constraints or the objective function. They encode complex logical conditions within the linear/Boolean framework.

Variable	Domain	Meaning	Created by
$\beta_{c,\sigma}$	$\{0, 1\}$	Course c offered in semester σ (any year)	C1
$h_{s,y}$	$\{0, 1\}$	Type s takes ≥ 1 optional course in year y	S8
$\text{att}_{s,y,\sigma,\omega,d,\pi}$	$\{0, 1\}$	Type s has any scheduled activity at (d, π) on physical week ω	S17
$z_{s,c,p,k}^{\text{att}}$	$\{0, 1\}$	Attendance: type s attends component (c, p) at slot k	O3
$z_{s,b,i,j}^{\text{cf}}$	$\{0, 1\}$	Conflict: activities i, j both attended by type s at base slot b	O3
$\delta_{c,y,\sigma,\omega,d,\pi}$	$\{0, 1\}$	Course c uses period π on day d (physical week ω)	C11
$\delta_{c,y,\sigma,\omega,d,\pi}^{\text{lec}}$	$\{0, 1\}$	Course c has a <i>lecture</i> in period π on day d	C12
overflow_k	$\mathbb{Z}_{\geq 0}$	Number of open components at slot k exceeding the soft limit	O4

Table 11: Auxiliary variables created by constraints and objective.

C. Full Mathematical Formulation

C.1 Student Constraints (S1–S19)

S1. Per-Semester Gateway Count. Each semester, each student type must take a number of gateway courses within the required range.

$$R_s^{\text{gw_min_sem}} \leq \sum_{c \in \mathcal{C}^{\text{GW}}} t_{s,c,y,\sigma} \leq R_s^{\text{gw_max_sem}}$$

for each $s \in \mathcal{S}, y \in \mathcal{Y}_s, \sigma \in \Sigma_s$. Bounds are enforced independently only when non-trivial ($R_s^{\text{gw_min_sem}} > 0$ or $R_s^{\text{gw_max_sem}} < 999$); the constraint is skipped when no gateway course choices exist for the student in that semester.

S2. Per-Semester Optional Count. Each semester, each student type must take a number of optional courses within the required range.

$$R_s^{\text{opt_min_sem}} \leq \sum_{c \in \mathcal{C}^{\text{OPT}}} t_{s,c,y,\sigma} \leq R_s^{\text{opt_max_sem}}$$

for each s, y, σ . Same guard conditions as S1.

S3. Per-Semester Course Count. Each semester, each student type's total course count (gateway + optional + outside) must lie within the allowed range.

$$R_s^{\text{crs_min_sem}} \leq \sum_c t_{s,c,y,\sigma} \leq R_s^{\text{crs_max_sem}}$$

for each s, y, σ . Same guard conditions as S1.

S4. Per-Semester Credit Bounds. Each semester, each student type's total credit load must stay within the minimum and maximum range.

$$R_s^{\text{cr_min_sem}} \leq \sum_c \text{cr}_c \cdot t_{s,c,y,\sigma} \leq R_s^{\text{cr_max_sem}}$$

for each s, y, σ . Same guard conditions as S1.

S5. Total Credits Per Year (Equality). Each year, each student type's total credits across both semesters must exactly equal the required annual credit load.

$$\sum_{\sigma \in \Sigma_s} \sum_c \text{cr}_c \cdot t_{s,c,y,\sigma} = R_s^{\text{cr_year}}$$

for each s, y .

S6. Outside Credits Per Year. Joint-programme students must accumulate enough credits from their partner department each year.

$$\sum_{\sigma} \sum_{c \in C^{\text{OUT}}} \text{cr}_c \cdot t_{s,c,y,\sigma} \geq R_s^{\text{out_cr_year}}$$

for each s with $\text{prog}_s \neq \text{SINGLE}$ and each y . Only enforced when $R_s^{\text{out_cr_year}} > 0$.

S7. Per-Year Gateway Count. Each year, each student type must take a number of gateway courses within the required range.

$$R_s^{\text{gw_min_yr}} \leq \sum_{\sigma} \sum_{c \in C^{\text{GW}}} t_{s,c,y,\sigma} \leq R_s^{\text{gw_max_yr}}$$

for each s, y . Same guard conditions as S1.

S8. Per-Year Optional Count (Conditional Minimum). See Section 3.5 for the full derivation.

S9. Per-Year Course Count. Each year, each student type's total course count must lie within the allowed range.

$$R_s^{\text{crs_min_yr}} \leq \sum_{\sigma} \sum_c t_{s,c,y,\sigma} \leq R_s^{\text{crs_max_yr}}$$

for each s, y .

S10. Balanced Credits Across Semesters. When $R_s^{\text{balanced}} = \text{true}$ and $|\Sigma_s| = 2$ (semesters σ_1, σ_2):

$$\sum_c \text{cr}_c \cdot t_{s,c,y,\sigma_1} = \sum_c \text{cr}_c \cdot t_{s,c,y,\sigma_2}$$

for each s, y . This prevents an unbalanced workload (e.g. enforces $60 + 60 = 120$).

S11. Total Gateway Requirement (Across All Years). Across all years in the planning horizon, each student type must take at least the required total number of gateway courses.

$$\sum_{y \in \mathcal{Y}_s} \sum_{\sigma} \sum_{c \in C^{\text{GW}}} t_{s,c,y,\sigma} \geq R_s^{\text{gw_total}}$$

for each s .

S12. Compulsory Course Enrolment. Each student type must take each of their compulsory courses at least once across all years and semesters.

$$\sum_{y \in \mathcal{Y}_s} \sum_{\sigma \in \Sigma_s} t_{s,c,y,\sigma} \geq 1$$

for each s and each compulsory course $c \in \text{comp}_s$.

S13. Course Uniqueness. No student type may take the same course more than once.

$$\sum_{y,\sigma} t_{s,c,y,\sigma} \leq 1$$

for each s and each course c .

S14. Sectioned Component Attendance. If a student type takes a course, the number of section-slots assigned must equal the required weekly meetings; otherwise no assignments are made.

$$\sum_{k:y_k=y,\sigma_k=\sigma} a_{s,c,p,k} = n_p^w \cdot t_{s,c,y,\sigma}$$

for each s , each course c , and each sectioned component $p \in \mathcal{P}_c^{\text{sec}}$.

S15. Take–Active Consistency. If any student type takes a course, that course must be active (offered).

$$t_{s,c,y,\sigma} = 1 \Rightarrow \alpha_{c,y,\sigma} = 1$$

for each (s, c, y, σ) .

S16. Course Enrolment Capacity. If a course is active, total weighted enrolment must be within its capacity limits.

$$\alpha_{c,y,\sigma} = 1 \Rightarrow \underline{E}_c \leq \sum_s N_s \cdot t_{s,c,y,\sigma} \leq \overline{E}_c$$

for each course c (with specified capacity range) and each (y, σ) .

S17. Maximum Daily Consecutive Gap. See Section 3.5 for the full two-step derivation (attendance indicators and gap elimination via Equations 3–5).

S18. Sibling-Subgroup Symmetry Breaking. See Section 3.5 for the full derivation.

S19. Student Max Slots Per Day. Each student type cannot attend more than $\text{slots}_{\max}$ class periods in a single day. Reuses the same att indicators defined in S17.

$$\sum_{\pi} \text{att}_{s,y,\sigma,\omega,d,\pi} \leq \text{slots}_{\max}$$

for each s, y, σ, d, ω . Only enforced when $\text{slots}_{\max} < 999$.

C.2 Course Constraints (C1–C13)

C1. Course Offered in At Most One Semester. A course must be offered in at most one semester (but may run in multiple academic years within that semester). Introduce $\beta_{c,\sigma} \in \{0, 1\}$:

$$\beta_{c,\sigma} \Leftrightarrow \bigvee_{y \in \mathcal{Y}} \alpha_{c,y,\sigma}$$

$$\sum_{\sigma \in \Sigma} \beta_{c,\sigma} \leq 1$$

for each course c .

C2. Active Consistency Across Years. Within the chosen semester, a course must be active in *all* allowed years or in none:

$$\alpha_{c,y_1,\sigma} = \alpha_{c,y_2,\sigma} \quad \forall y_1, y_2 \in \mathcal{Y}$$

for each course c , semester σ .

C3. Global Gateway Offering Count Per Semester. The department must offer exactly the required number of gateway courses per semester:

$$\sum_{c \in \mathcal{C}^{\text{GW}}} \beta_{c,\sigma} = G_{\sigma}^{\text{GW}}$$

for each semester σ .

C4. Global Optional Offering Count Per Semester. Similarly, the required number of optional courses must be offered:

$$\sum_{c \in \mathcal{C}^{\text{OPT}}} \beta_{c,\sigma} = G_{\sigma}^{\text{OPT}}$$

for each semester σ .

C5. Active $\Rightarrow$ Section Count Bounds. When a course is active, each component must open the correct number of timeslots; when inactive, all open variables are zero.

$$\underline{S}_p \cdot n_p^w \cdot \alpha_{c,y,\sigma} \leq \sum_{k: y_k=y, \sigma_k=\sigma} o_{c,p,k} \leq \bar{S}_p \cdot n_p^w \cdot \alpha_{c,y,\sigma}$$

for each course c , component p , year y , semester σ . Additionally, $o_{c,p,k} = 1 \Rightarrow \alpha_{c,y_k,\sigma_k} = 1$ (Open $\Rightarrow$ Active implication).

C6. Active $\Rightarrow$ Minimum Attendance Threshold. A course can only be active if enough students are enrolled to fill the minimum required sections.

$$\alpha_{c,y,\sigma} = 1 \Rightarrow \sum_s N_s \cdot t_{s,c,y,\sigma} \geq \underline{Q}_p \cdot \underline{S}_p$$

for each course c , component p with $\underline{Q}_p \cdot \underline{S}_p > 0$, and each (y, σ) .

C7. Force Inactive When No Students Can Enrol. If no student type has a take-variable for a course in a given (y, σ) , the course is forced inactive:

$$\alpha_{c,y,\sigma} = 0 \quad \text{if } \nexists s : t_{s,c,y,\sigma} \text{ exists}$$

C8. Sectioned Section Capacity. Each workshop section has a minimum and maximum capacity; if closed, no students can be assigned.

$$o_{c,p,k} = 0 \Rightarrow \sum_s N_s \cdot a_{s,c,p,k} = 0$$

$$o_{c,p,k} = 1 \Rightarrow \underline{Q}_p \leq \sum_s N_s \cdot a_{s,c,p,k} \leq \bar{Q}_p$$

for each sectioned component $p \in \mathcal{P}_c^{\text{sec}}$ and each timeslot key k .

C9. Non-Sectioned Component Capacity. For lecture-type components (all enrolled students attend the same slot), total enrolment must be within capacity when the slot is open.

$$o_{c,p,k} = 1 \Rightarrow \underline{Q}_p \leq \sum_s N_s \cdot t_{s,c,y_k,\sigma_k} \leq \bar{Q}_p$$

for each non-sectioned component $p \notin \mathcal{P}_c^{\text{sec}}$ and each timeslot key k .

C10. Internal Course Conflict Avoidance. Within the same course, at most one lecture can occupy the same physical timeslot per week pattern. For each course c and each base slot $b = (y, \sigma, d, \pi)$, define lecture open-var sums per week pattern w :

$$L_w = \sum_{p \in \text{Lec}, k: \text{base}(k)=b, \omega_k=w} o_{c,p,k}$$

Per week-pattern: $L_w \leq 1$ for each $w \in \{A, O, E\}$.

Cross week-pattern exclusions (ALL conflicts with ODD/EVEN):

$$L_A + L_O \leq 1, \quad L_A + L_E \leq 1$$

Global timeslot cap across all courses and components (ODD and EVEN may co-exist since they run on different physical weeks):

$$\sum_{(c,p,k): \text{base}(k)=b, \omega_k=A} o_{c,p,k} + \sum_{(c,p,k): \text{base}(k)=b, \omega_k \in \{O, E\}} o_{c,p,k} \leq \text{conc_slot}_{\max}$$

C11. Course Max Slots Per Day. A non-outside course cannot use more than $\text{day}_{\max}$ periods on the same day. For each non-outside course c , year y , semester σ , day d , physical week ω :

Create indicator $\delta_{c,y,\sigma,\omega,d,\pi} \Leftrightarrow \bigvee_{p,k} o_{c,p,k}$ for each period π (over k with $(d_k, \pi_k) = (d, \pi)$ and ω_k compatible with ω).

$$\sum_{\pi} \delta_{c,y,\sigma,\omega,d,\pi} \leq \text{day}_{\max}$$

Only created when the number of candidate periods exceeds $\text{day}_{\max}$.

C12. Lecture Max Per Day. On any physical day, a non-outside course can have at most $\text{lec_day}_{\max}$ distinct lecture periods. For each non-outside course c (with lecture components), year y , semester σ , day d , physical week ω :

Create indicator $\delta_{c,y,\sigma,\omega,d,\pi}^{\text{lec}} \Leftrightarrow \bigvee_{p \in \text{Lec}, k} o_{c,p,k}$ for each period π .

$$\sum_{\pi} \delta_{c,y,\sigma,\omega,d,\pi}^{\text{lec}} \leq \text{lec_day}_{\max}$$

Only created when the number of candidate lecture periods exceeds $\text{lec_day}_{\max}$.

C13. Workshop After Lecture. When $\text{ws_after_lec} = \text{true}$, every open workshop slot must be scheduled on a day that is the same as or later than a day on which the course has at least one open lecture. For each non-outside course c (with workshop component p_w), each allowed timeslot key k_w :

$$o_{c,p_w,k_w} = 1 \Rightarrow \sum_{p_l \in \text{Lec}, k_l: y_{k_l}=y, \sigma_{k_l}=\sigma, d_{k_l} \leq d_{k_w}} o_{c,p_l,k_l} \geq 1$$

If no lecture key exists on any day $\leq d_{k_w}$, the workshop slot is forbidden: $o_{c,p_w,k_w} = 0$.

C.3 Objective Function (O1–O4)

The model minimises a weighted combination of penalties minus bonuses (see also Section 3.6 for a summary and default weight table):

$$\min \quad \underbrace{w_{\text{act}} \sum_{c,y,\sigma} \alpha_{c,y,\sigma}}_{\text{O2: activation penalty}} + \underbrace{\sum_{s,b,(i,j)} \text{sev}(i,j) \cdot N_s \cdot z_{s,b,i,j}^{\text{cf}}}_{\text{O3: conflict penalties}} + \underbrace{w_{\text{room}} \sum_k \text{overflow}_k}_{\text{O4: room overflow}} - \underbrace{w_{\text{desire}} \sum_{s,c \in \text{des}_s, y,\sigma} N_s \cdot t_{s,c,y,\sigma}}_{\text{O1: desire bonus}}$$

O1. Desired-Course Bonus. The model grants a bonus when students take their desired courses:

$$\text{Bonus}_{\text{total}} = w_{\text{desire}} \cdot \sum_s \sum_{c \in \text{des}_s} \sum_{y, \sigma} N_s \cdot t_{s,c,y,\sigma}$$

O2. Activation Penalty. A small penalty for each active course–semester pair, discouraging unnecessary activations:

$$\text{Penalty}_{\text{act}} = w_{\text{act}} \cdot \sum_{c,y,\sigma} \alpha_{c,y,\sigma}$$

O3. Conflict Penalty. Penalises timetable clashes with severity depending on the component types involved (Lecture–Lecture > Lecture–Workshop > Workshop–Workshop; see Table 1 for default weights). See Section 3.5 for the full derivation of the attendance indicator z^{att} and conflict indicator z^{cf} .

O4. Concurrent Open Courses Penalty. A soft penalty for each course-component exceeding the concurrent-open soft limit at any timeslot:

$$\text{overflow}_k \geq \sum_{c,p} o_{c,p,k} - \text{soft_limit}, \quad \text{overflow}_k \geq 0$$

$$\text{Penalty}_{\text{room}} = w_{\text{room}} \cdot \sum_k \text{overflow}_k$$

D. Non-Obvious Constraint Details

The derivations for constraints S8 (conditional optional minimum), S17 (maximum daily consecutive gap), S18 (symmetry breaking), and objective term O3 (conflict penalty) involve non-trivial modelling logic and are discussed in full detail in Section 3.5 (main text).

E. Solver Configuration and Reproducibility

Parameter	Debug	Production
Workers	4	13
Time limit	120 min	120 min
Random seed	42	random
Linearisation level	1	2
Probing level	2	2
Symmetry level	3	3
Cut level	default	2
Max cuts	default	10 000
SAT inprocessing	default	enabled
Memory limit	default	90% RAM

Table 12: Solver parameters.

F. Additional Figures

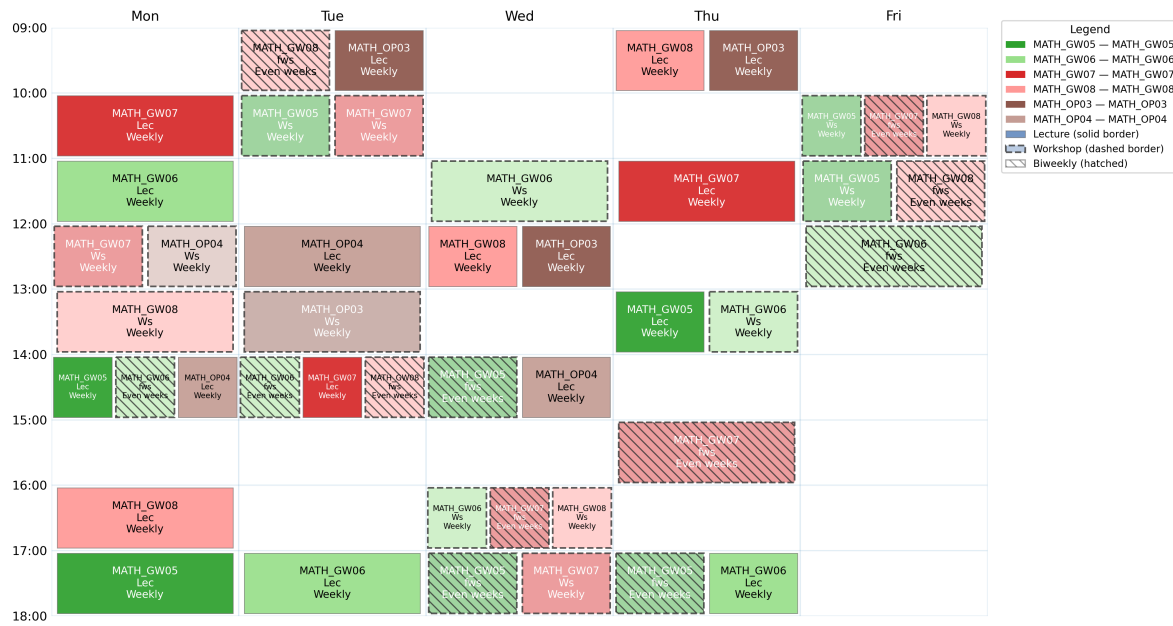


Figure 4: Year 3 Mathematics timetable in Semester 2 (V7-Final).

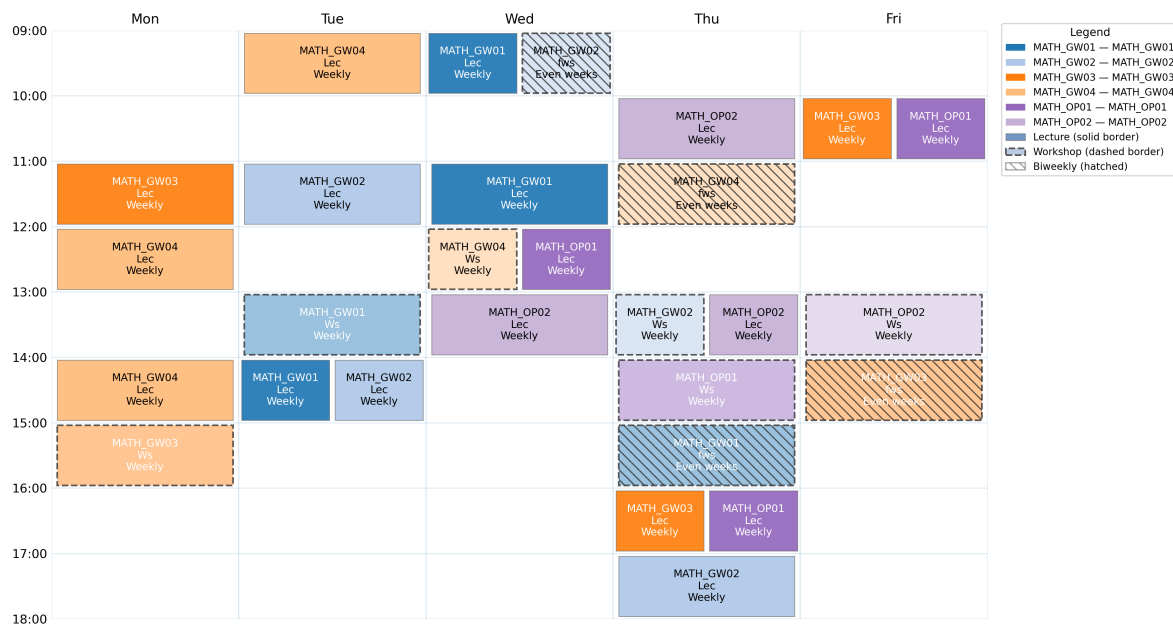


Figure 5: Year 3 Mathematics timetable in Semester 1 (V0 baseline). Compared with V7-Final (Figure 3), teaching sessions are more dispersed across the day and extend into late afternoon (17:00–18:00), reflecting the absence of quality-oriented scheduling rules.

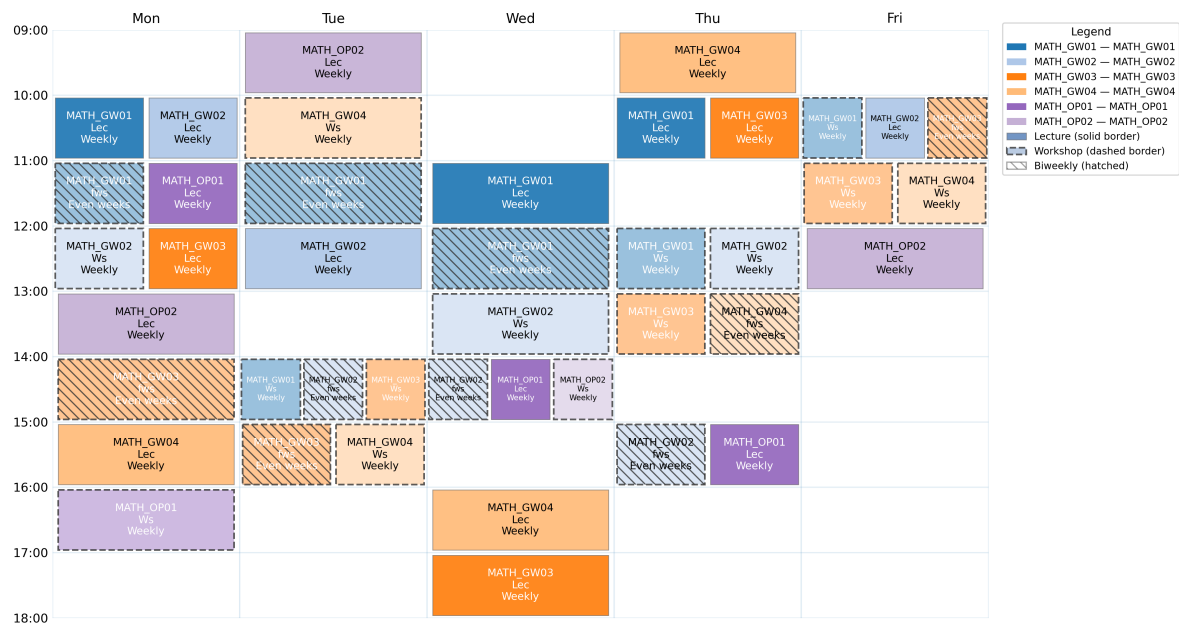


Figure 6: Year 3 Mathematics timetable in Semester 1 (V9). Consecutive-lecture blocks are visible, constraining workshop placement and contributing to the 211 conflicts observed in this scenario.

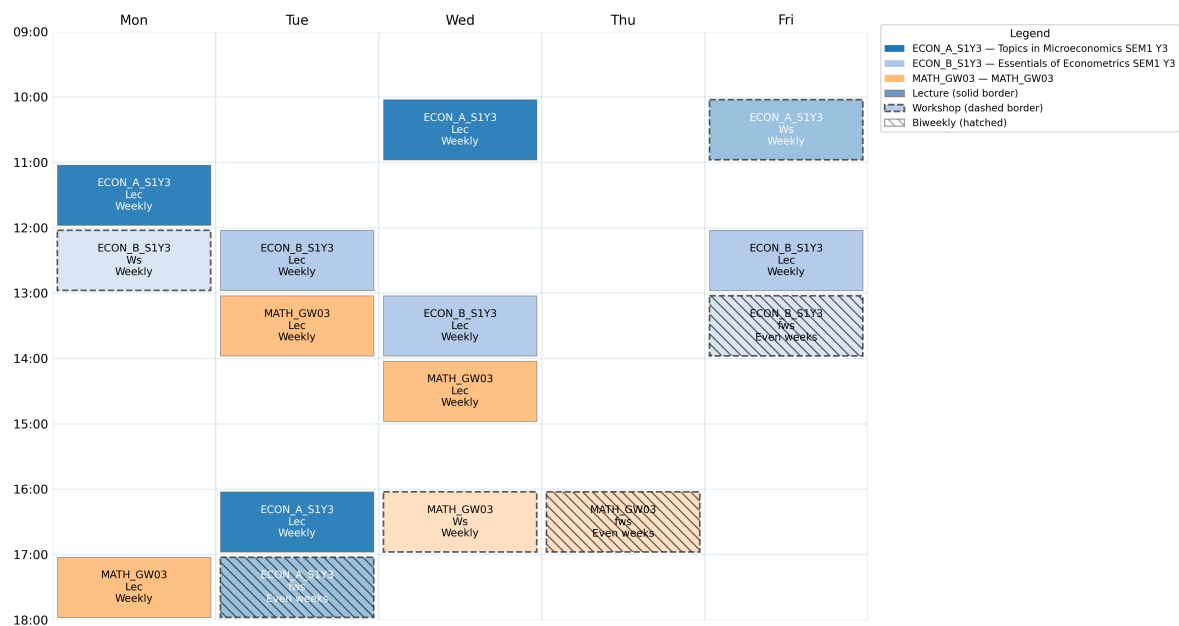


Figure 7: Individual timetable for a Maths–Economics joint-programme student (type 17) in Year 3 Semester 1 under V7-Final. External Economics courses occupy substantial time windows, leaving constrained but conflict-free slots for Mathematics classes.

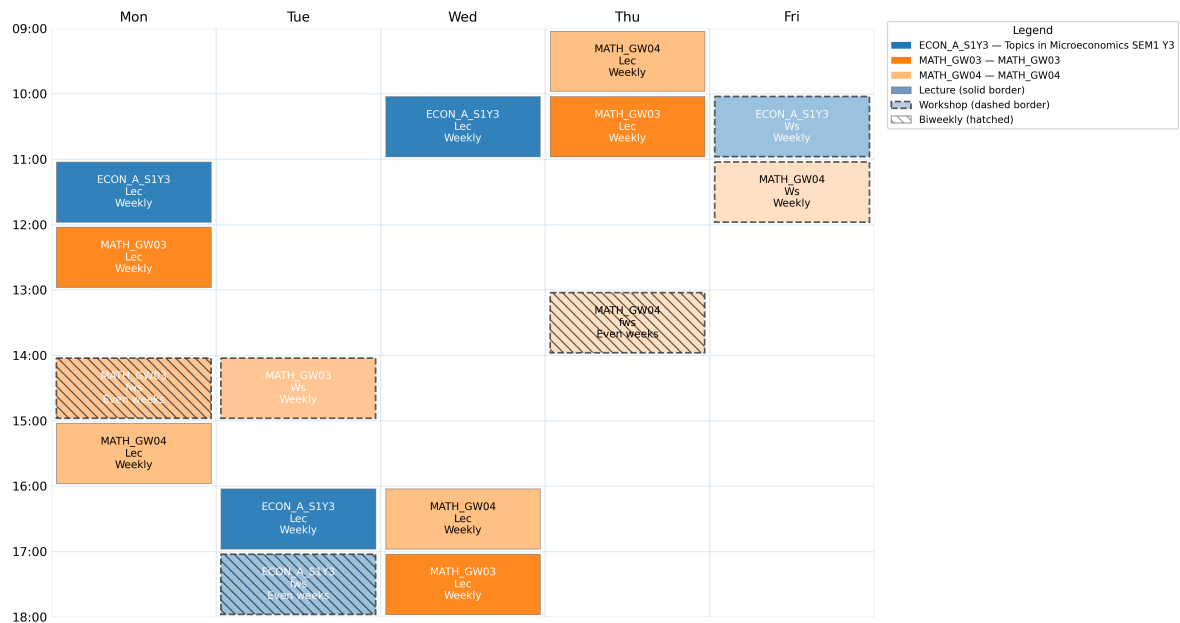


Figure 8: Individual timetable for the same Maths–Economics student type (type 17) under V9. Compared with V7-Final (Figure 7), the consecutive-lecture constraint shifts Mathematics blocks into time windows occupied by Economics courses, generating workshop conflicts.

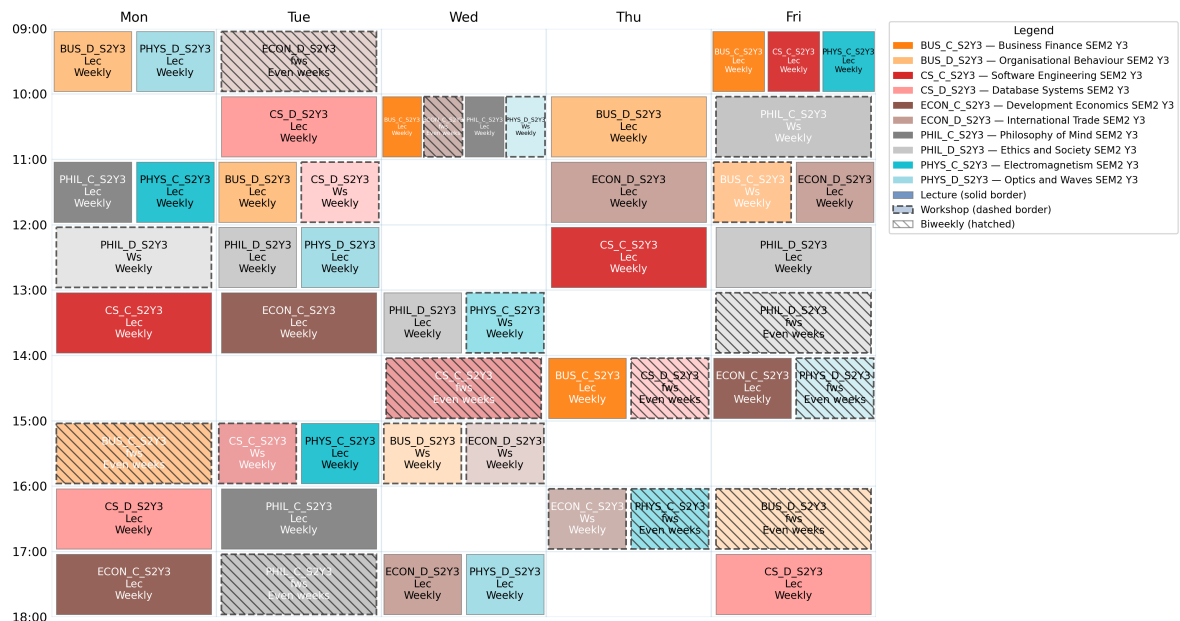


Figure 9: Merged Semester 2 timetable of non-Mathematics courses for Year 3 joint-programme students. Together with Semester 1 (Figure 1), this illustrates the extensive external scheduling constraints faced by joint-programme students across the full academic year.

G. Additional Scenario and Sensitivity Results

This appendix provides the complete scenario definitions, solver results, and analysis for all model instances evaluated in this project. Table 13 summarises the key metrics; the subsections below give per-scenario details grouped by experimental stage.

Scenario	Types	Students	Objective	Bound	Conflicts	LL / LW / WW	% Affected	Status
<i>Stage 1 — Compactness & Sequencing (V0–V3)</i>								
V0	56	475	128	128	0	0/0/0	0.0	Optimal
V1	56	475	128	128	0	0/0/0	0.0	Optimal
V2	56	475	128	128	0	0/0/0	0.0	Optimal
V3	56	475	128	128	0	0/0/0	0.0	Optimal
<i>Stage 2 — Quality Rules (V4–V6 & MC variants)</i>								
V4	56	475	128	128	0	0/0/0	0.0	Optimal
V4-MC	76	475	206	206	0	0/0/0	0.0	Optimal
V5	56	475	209	206	0	0/0/0	0.0	Feasible (1.4%)
V5-MC	76	475	206	206	0	0/0/0	0.0	Optimal
V6	56	475	206	206	0	0/0/0	0.0	Optimal
V6-MC	76	475	208	206	0	0/0/0	0.0	Feasible (1.0%)
<i>Stage 3 — Robustness & Boundary (V7–V9)</i>								
V7	76	475	5 958	206	621	178/397/46	78.9	Feasible
V7.5	86	525	206	206	0	0/0/0	0.0	Optimal
V7-Final	76	475	128	128	0	0/0/0	0.0	Optimal
V7-Final-NP	76	475	128	128	0	0/0/0	0.0	Optimal
V8	100	475	281	206	0	0/0/0	0.0	Feasible (3.6%)
V9	76	475	1 438	206	211	0/192/19	28.8	Feasible

Table 13: Solver results across all scenarios. MC = more-choice variant (76 parent types); LL = Lecture–Lecture, LW = Lecture–Workshop, WW = Workshop–Workshop. Time limit: 120 min for all runs.

G.1 Stage 1 — Compactness and Sequencing (V0–V3)

Version	Rule changes (cumulative)	Key result
V0	max_daily_gap = 4; course_max_slots/day = 2	Objective = 128 (optimal, 0 conflicts). Solve time ≈ 299 s; ~ 1.59 M variables. Confirms baseline feasibility of the proposed curriculum.
V1	+ student_max_slots/day = 4 + workshop_after_lecture	Objective = 128 (optimal, 0 conflicts). Solve time ≈ 268 s. Workshop sequencing is satisfiable on this dataset (unlike earlier smaller-scale tests).
V2	max_daily_gap $\downarrow$ 3 + max_concurrent_courses = 4	Objective = 128 (optimal, 0 conflicts). Solve time $\approx 1\,804$ s; model grows to ~ 1.72 M variables. Tighter gap and concurrency cap remain non-binding.
V3	+ parallel workshop sections (sections_max $\uparrow$ in data)	Objective = 128 (optimal, 0 conflicts). Solve time $\approx 5\,995$ s; adds ~ 208 k BoolOr constraints. Parallel workshops: costless structural improvement.

Stage 1 summary. All four scenarios achieve the same optimal objective (128) with zero conflicts. The model comfortably absorbs each successive constraint at the 56-type level. Solve time increases reflect model complexity growth, not solution-quality loss.

G.2 Stage 2 — Quality Rules (V4–V6 and MC Variants)

Version	Rule changes (cumulative)	Key result
V4	+ lunch_break_no_class	Objective = 128 (optimal, 0 conflicts). Solve ≈ 1 676 s. Removing the 12–13 slot costs one period/day but does not bind.
V4-MC	Same rules; types $\uparrow$ 76	Objective = 206 (optimal, 0 conflicts). Objective rises (more activation penalties from richer type set); feasibility preserved with 76 types.
V5	+ max_consecutive_slots = 2 + late-slot soft penalty	Objective = 209; Bound = 206 (gap 1.4%, 0 conflicts). Consecutive-slot cap increases solver difficulty; small gap remains.
V5-MC	Same rules; types $\uparrow$ 76	Objective = 206 (optimal, 0 conflicts). MC variant reaches a <i>better</i> objective than the 56-type base, suggesting richer routing flexibility.
V6	max_daily_gap $\uparrow$ 5 + no_9 am MATH classes + no_Fri afternoon MATH + ws_no_wk/ftn overlap + max_same_ws/slot = 2	Objective = 206 (optimal, 0 conflicts). Five new rules added; gap relaxation (3 $\rightarrow$ 5) compensates for reduced available timeslots.
V6-MC	Same rules; types $\uparrow$ 76	Objective = 208; Bound = 206 (gap 1.0%, 0 conflicts). Solve hits 120 min limit; ~ 1.66 M variables. Practical boundary for near-optimal solutions under the full V6 rule set.

Stage 2 summary. All scenarios remain **zero-conflict**. The objective shifts from 128 to ~ 206 –209 as the type set expands and activation penalties grow. The MC variants confirm that 76 types are feasible under the quality-rule set, although solve times approach the 120-minute limit.

G.3 Stage 3 — Robustness and Boundary Exploration (V7–V9)

This stage probes the limits of the model by increasing student-type diversity and imposing the consecutive-lecture constraint.

V7 — Stress Test (76 types, full V6 rules).

Types	76	Objective	5 958	Conflicts	621	% Affected	78.9%
Students	475	Bound	206	LL / LW / WW	178 / 397 / 46		

Analysis. The *worst-performing scenario*. Retaining all V6 quality rules (lunch break, universal 9:00 ban, late-slot penalty) while increasing to 76 types creates an over-constrained space that the solver cannot resolve within 120 min. This establishes the hard feasibility boundary for the full V6 rule set.

V7.5 — Extended Population (86 types, 525 students).

Types	86	Objective	206	Conflicts	0	% Affected	0.0%
Students	525	Bound	206				

Analysis. Despite having *more* types and students than V7, V7.5 achieves optimality with zero conflicts. The key difference is in the rule set: V7.5 adopts the same relaxed rules as V7-Final—no lunch-break restriction, and the 9:00 ban applies only on Monday, Wednesday, and Friday—rather than the strict V6 rules used in V7. This confirms that problem difficulty is driven by the interaction between demand diversity and constraint tightness, not by instance size alone.

V7-Final — Selective Relaxation (76 types).

Types	76	Objective	128	Conflicts	0	% Affected	0.0%
Students	475	Bound	128				

Three relaxations vs. V7:

1. Lunch-break restriction **removed**.
2. 9 am ban restricted to Mon/Wed/Fri only (Tue/Thu allowed).
3. Late-slot penalty **dropped**.

All other V6 rules retained. Solve time $\approx 4\,297$ s (~ 72 min); ~ 2.05 M vars.

Analysis. Best result for 76 types: same optimal objective as V0 baseline. The three relaxations completely eliminate V7's 621 conflicts, confirming that the lunch-break and universal 9 am restrictions are the primary drivers of infeasibility at this scale.

V7-Final-NP — Non-Parallel Workshop Variant.

Types	76	Objective	128	Conflicts	0	% Affected	0.0%
Students	475	Bound	128				

Same rules as V7-Final; `sections_max = 1` (no parallel workshops). Solve time $\approx 1\,520$ s (~ 25 min)— $3\times$ faster due to fewer section-assignment variables. Confirms zero-conflict solutions exist without parallelism, though parallel sections remain beneficial for real-world flexibility.

V8 — Maximum Type Expansion + Consecutive Lectures (100 types).

Types	100	Objective	281	Conflicts	0	% Affected	0.0%
Students	475	Bound	206				

Changes from V7-Final: + `same_day_lectures_consecutive`; types expanded to 100 (~ 20 per joint programme)—the largest instance tested.

Analysis. Zero conflicts despite 100 types and the consecutive-lecture constraint, though the 3.6% optimality gap indicates the solver cannot close the bound within 120 min. The practical implication is positive: even under the tightest rule set and maximum type diversity, the model produces a clash-free timetable.

V9 — Consecutive Lectures with 76 Types.

Types	76	Objective	1 438	Conflicts	211	% Affected	28.8%
Students	475	Bound	206	LL / LW / WW	0 / 192 / 19		

Same rules as V8, types reduced to 76. Programme-level breakdown:

Programme	Conflicts	% Affected
Single-honours	0	0.0
Joint CS	84	83.6
Joint Economics	49	49.1
Joint Physics	44	65.5
Joint Business	31	45.5
Joint Philosophy	3	5.5

Analysis. No lecture–lecture conflicts: the consecutive-lecture rule itself does not cause direct clashes, but it constrains lecture-block positions, indirectly triggering workshop conflicts for joint students with incompatible outside schedules. Performs *worse* than V8 despite fewer types, reinforcing that student-type composition matters more than type count.

Stage 3 summary. The V7→V7-Final comparison identifies the lunch-break and universal 9 am restrictions as the sole feasibility bottlenecks at 76 types. Selective relaxation fully restores the V0 optimal. The consecutive-lecture rule (V8/V9) is data-sensitive: zero conflicts at 100 types (V8) but 211 conflicts at 76 (V9).

G.4 Cross-Scenario Sensitivity Summary

1. Constraint sensitivity ranking.

1. **Lunch break + universal 9 am ban:** primary infeasibility drivers at ≥ 76 types (V7 vs. V7-Final).
2. **Consecutive-lecture rule:** data-dependent; conflicts in V9 but not V8.
3. **Workshop sequencing, concurrent cap, daily load cap:** non-binding across all tested configurations.
4. **Gap relaxation** ($3 \rightarrow 5$): necessary compensating measure when multiple slot restrictions are active (V6).

2. Scalability. Up to 100 parent types with zero conflicts under the V7-Final/V8 rule set. Solve times range from ~ 5 min (V0) to the 120 min limit (V6-MC, V8), correlating with both type count and constraint tightness.

3. More-choice variants. MC variants (V4-MC through V6-MC) consistently achieve zero or near-zero conflicts. In some cases (V5-MC vs. V5), the MC variant achieves a better objective, suggesting that richer feasible regions can aid solver routing.

4. Data sensitivity. V7 (621 conflicts) vs. V7.5 (0 conflicts with *more* types/students but a relaxed rule set) shows that difficulty is not monotonic in instance size; the interaction between demand diversity and constraint tightness is the dominant factor.