Cohort-Based Timetabling with Integer Linear Programming

Richard Hoshino and Jameson Albers

Northeastern University, Vancouver BC, Canada
Khoury College of Computer Sciences
{r.hoshino, albers.j}@northeastern.edu

Abstract. At some educational institutions, students are divided into cohorts, where they complete the same set of courses with everybody else in that cohort. In this paper, we describe an Integer Linear Programming (ILP) solution to the School Timetabling Problem (STP), for schools that are cohort-based. Our Master Timetable is generated from the input data, a user-friendly Excel document that lists all of the course/cohort/teacher/classroom constraints.

Our model, which is coded in Python and solved using the MPSolver from Google OR-Tools, generated the 2023-2024 Master Timetable for three schools in Canada: an elementary school, a middle school, and a post-secondary institute. All three timetables satisfied 100% of the school's hard constraints, and were computed in less than 60 seconds.

Keywords: Integer Programming, Linear Programming, Timetabling

1 Introduction

Timetabling is defined as "the act of scheduling something to happen or do something at a particular time" [2]. This simple definition conceals the challenge and complexity of timetabling.

For educational institutions, the Master Timetable dictates to every single teacher and student where they need to be at each hour of the school day. Given its importance, some school administrators spend weeks or months constructing the annual Master Timetable, often using post-it notes or wall magnets to assign a teacher, classroom, and timeslot to each section of a course.

When timetables are constructed by hand, the process is inefficient and the product is sub-optimal. This is why researchers have investigated the School Timetabling Problem (STP) for sixty years [6], creating timetables for schools all over the world [11].

In the most basic version of the STP, the objective is to assign courses to teachers, timeslots, and classrooms, subject to the following constraints: a teacher cannot teach two courses in the same timeslot, no classroom can be used by two courses simultaneously, and each teacher has a set of unavailable teaching timeslots. This basic version of the STP is NP-complete [4].

Although scholars have conducted research on educational timetabling since the 1960s [1], it took until the mid-1990s to standardize educational timetabling problems, along with their corresponding benchmark data sets [3].

To advance the field of educational timetabling, a group of researchers have run the International Timetabling Competition, with the 2011 edition focused on high school timetabling [12] and the 2019 edition focused on university timetabling [10]. Both competitions were modeled on real-world data sets.

The Post-Enrollment Course Timetabling Problem (PECTP) was introduced to incorporate student course preferences into the STP [9]. The PECTP involves student-related hard constraints, such as ensuring that no student is enrolled in multiple sections of the same course, and the objective function is to maximize the number of occurrences where students are enrolled in their desired courses.

The lead author has created 40 Master Timetables for various Canadian high schools over the past five years, using his published algorithms to solve large real-life PECTP instances for schools: using graph coloring [7] and large neighborhood search [8]. While each Master Timetable was custom-built to meet the school's specific requirements, several of these timetabling projects were extremely similar in that the schools were *cohort-based*, where students were divided into fixed groups and took the same set of courses.

For cohort-based timetables, the PECTP reverts back to the STP, since student preferences do not exist. Thus, the optimal timetable can be generated by solving an Integer Linear Program (ILP) where the objective function considers teacher preferences for the timeslots they would like to teach their courses.

We now present our solution to solving virtually any cohort-based STP. Our automated timetabling algorithm requires a single input file: an Excel document that contains the teacher preferences as well as all of the constraints involving courses, cohorts, teachers, and rooms. We will explain how we worked with school administrators at a Kindergarten to Grade 5 elementary school, a Grade 6 to Grade 8 middle school, and a design academy for post-secondary students, to generate each institution's provably-optimal 2023-2024 Master Timetable.

Despite the different contexts of all three of these educational institutions, we used the *exact same Python program* to create all three timetables; the only difference was that each school had its own input Excel file. At the end of our Python program, we call MPSolver, the Mathematical Programming solver from Google OR-Tools [5] that solves Mixed Integer Programs (MIPs).

2 Mathematical Model

Each course c has one or more lessons (or meetings) in a week. Thus, we define our main binary decision variable as $X_{m,c,d,p}$, which equals 1 if and only if meeting m of course c is scheduled on day d in period p. Otherwise, $X_{m,c,d,p} = 0$.

We can view each (d, p) pair as a *timeslot*, and each (m, c) pair as a single *event* that is attended by one or more student cohorts, is taught by one or more teachers, and is offered in one or more rooms. Our ILP will generate the Master Timetable by assigning exactly one timeslot to each event.

Let D be the set of days, P be the set of periods, and C be the set of courses. For each course c, if there are $L(c)$ lessons (i.e., events) that must be scheduled during the $|D|$-day timetable, then we have the following constraint.

$$\sum_{d \in D} \sum_{p \in P} X_{m,c,d,p} = 1 \qquad \forall\, c \in C, m \in [1, L(c)] \tag{1}$$

Each course $c \in C$ is unique, where the teacher(s) and room(s) for course c are pre-assigned by the school.

For example, one of our school clients has 6A-Science, 6B-Science, 6C-Science in its set of courses C since each of the three Grade 6 cohorts has its own Science course that meets three times each week. Additionally, this school has all of its Grade 6 students taking Physical Education at the same time. This single course, 6-PhysEd, is offered to all three cohorts (6A, 6B, 6C), is co-taught by three teachers, and takes place in two rooms (Gym1, Gym2).

By specifying the cohorts/teachers/rooms for each event, once our ILP solver determines all four-tuples (m, c, d, p) for which $X_{m,c,d,p} = 1$, we can rapidly generate the various "cross-sections" of our Master Timetable to determine the timetable from the perspective of each course, each cohort, each room, and each teacher.

For each cohort, each teacher, and each room, we can determine E, the set of events (m, c) involving that entity, and ensure that there are no scheduling conflicts. Thus, we have our next set of hard constraints.

$$\sum_{(m,c) \in E(h)} X_{m,c,d,p} \leq 1 \;\forall\, d \in D, p \in P, h \in \text{Cohorts} \tag{2}$$

$$\sum_{(m,c) \in E(t)} X_{m,c,d,p} \leq 1 \;\forall\, d \in D, p \in P, t \in \text{Teachers} \tag{3}$$

$$\sum_{(m,c) \in E(r)} X_{m,c,d,p} \leq 1 \;\forall\, d \in D, p \in P, r \in \text{Rooms} \tag{4}$$

Finally, we define $PR_{m,c,d,p}$ to be the *preference* of having meeting m of course c scheduled on day d and period p. This preference coefficient may be influenced by a teacher's desire to teach on certain days and periods, or pedagogical reasons of having certain courses assigned to particular timeslots. Thus, the **objective function** of our ILP is

$$\sum_{m \in M} \sum_{c \in C} \sum_{d \in D} \sum_{p \in P} PR_{m,c,d,p} X_{m,c,d,p}.$$

There are two major families of constraints in our model, and we now explain each one in detail.

2.1 Family I: Restrictions on Sets of Events

Let E be a set of events and let T be a set of timeslots. Then we can connect E and T via the following linear constraint.

$$\sum_{(m,c)\in E} \sum_{(d,p)\in T} X_{m,c,d,p} \quad \{=, \leq, \geq\} \quad n \tag{5}$$

For each constraint, we choose the appropriate sign from $\{=, \leq, \geq\}$, and set n to be a specific non-negative integer. Let us provide several examples on the versatility of this family of constraints.

(i) "Teacher X is unavailable to teach on Tuesdays": E is the set of events taught by Teacher X, T is the set of timeslots with $d = 2$, our sign is $=$, and $n = 0$.
(ii) "The majority of the three 6A-French lessons must occur before lunch": E is the set of 6A-French events, T is the set of timeslots that occur before lunch, our sign is $\geq$, and $n = 2$.
(iii) "There is at most one Grade 8 Art class scheduled in Period 4": E is the set of events whose course is Grade 8 Art, T is the set of timeslots with $p = 4$, our sign is $\leq$, and $n = 1$.
(iv) "Ensure that Grade 7s do not have Physical Education more than once on any day": E is the set of events whose course is Grade 7 Physical Education, T is the set of timeslots with $d = k$, our sign is $\leq$, and $n = 1$. We repeat this constraint for each $k \in [1, |D|]$.
(v) "Ensure that Grade 7s do not have Physical Education on three consecutive days": E is the set of events whose course is Grade 7 Physical Education, T is the set of timeslots with $d \in [k, k+1, k+2]$, our sign is $\leq$, and $n = 2$. We repeat this constraint for each $k \in [1, |D| - 2]$.

This framework enables us to model constraints that relate almost *any* set of events to *any* set of timeslots, including the five examples provided above. We can place constraints on teacher and room availability, guarantee that certain events are scheduled (or not scheduled) in certain timeslots, spread out the multi-lesson courses taken by each cohort, and ensure that each teacher has a reasonable schedule each day without too many consecutive lessons or large gaps of non-teaching periods.

2.2 Family II: Relationships between Sets of Events

Let $E_i = (m_i, c_i)$ for each $i \geq 1$, and let $E_1, E_2, \ldots, E_v$ be a set of v events. Using a linear equation or linear inequality, we can model five additional timetabling constraints that relate these v events.

(i) All v events must occur in the same timeslot.

$$\begin{gathered} X_{m_i,c_i,d,p} = X_{m_{i+1},c_{i+1},d,p} \\ \forall d \in D, p \in P, i \in [1, v-1] \end{gathered} \tag{6}$$

(ii) All v events must occur on the same day.

$$\sum_{p\in P} X_{m_i,c_i,d,p} = \sum_{p\in P} X_{m_{i+1},c_{i+1},d,p} \tag{7}$$

$$\forall d \in D, i \in [1, v-1]$$

(iii) The v events must occur on v different days.

$$\sum_{i \in [1,v]} \sum_{p \in P} X_{m_i,c_i,d,p} \leq 1 \quad \forall d \in D \tag{8}$$

(iv) The v events must occur on v (different) consecutive days, with E_i occurring before E_j for all $i < j$.

$$\sum_{p \in P} X_{m_i,c_i,d_1,p} + \sum_{p \in P} X_{m_{i+1},c_{i+1},d_2,p} \leq 1 \tag{9}$$
$$\forall i \in [1, v-1] \text{ and } d_1, d_2 \in D \text{ with } d_2 - d_1 \neq 1.$$

A similar set of inequalities also allows us to ensure that the v events must occur in v consecutive periods of the same day, with E_i occurring before E_j for all $i < j$.

(v) There is a minimum gap of g days between events E_i and E_{i+1}, for all $1 \leq i \leq v-1$.

$$\sum_{p \in P} X_{m_i,c_i,d_1,p} + \sum_{p \in P} X_{m_{i+1},c_{i+1},d_2,p} \leq 1 \tag{10}$$
$$\forall i \in [1, v-1], \text{ and } d_1, d_2 \in D \text{ with } |d_2 - d_1| < g. \tag{11}$$

To get a maximum gap of g days, we simply replace $<$ with $>$ in the above inequality.

This versatile and flexible framework enables us to model constraints that relate almost any set of events to each other. For example, we can ensure that certain courses are not scheduled on the same day, that two cohorts have their French courses at the exact same time each week, and that a part-time teacher's work times are limited to two consecutive days in the timetable.

Real-life School Timetabling Problems (STPs) can be modeled effectively using the two families of constraints provided in this section. In fact, for all three of our cohort-based schools, we were able to model 100% of their constraints using these two families of constraints, and generate each school's Master Timetable in less than 60 seconds. We now explain how we accomplished these results.

3 Solving Three Different STPs

Victoria is the capital city of the Canadian province of British Columbia, and is the home of two leading independent co-educational K-12 preparatory schools named St. Michaels University School (SMUS) and Glenlyon Norfolk School (GNS). Victoria is also the location of Pacific Design Academy (PDA), an innovative post-secondary institute that offers eight full-time diploma programs including Fashion Design, Interior Design, and Graphic Media Design.

We were hired to create the Master Timetable for the SMUS Junior School (Kindergarten to Grade 5), the GNS Middle School (Grade 6 to Grade 8), and the entire PDA academic timetable with its eight different diploma programs.

We worked closely with the administrators at the three institutions to create an Excel document that encoded all of the school's constraints and requirements, as well as teacher preferences, and would serve as the input file to our Python program. As mentioned earlier, we used the same Python program for all three timetables, which called Google's MPsolver to solve our Integer Linear Program. Each school's input Excel file consists of the following five worksheets.

1. Timetable Structure
2. Timetable Content
3. Event Set Constraints
4. Event Relationship Constraints
5. Teacher Preferences

The **Timetable Structure** worksheet contains $|P|$ rows and $|D|$ columns, indicating the names of each day (1-Monday, 2-Tuesday, ...) and each period (Period 1, Period 2, ...). Each of the $|D||P|$ timeslots is labeled d-p, for each day and period. For example, 2-4 is Tuesday Period 4.

Each of our three schools had a different number of timeslots, with SMUS having 5 days and 7 periods, GNS having 10 days and 5 periods, and PDA having 5 days and 3 periods.

The **Timetable Content** worksheet provides the complete set of events, containing the ID of each unique course and the number of total meetings for that course. For each course $c \in C$, this worksheet also lists the name of the course, and all of the affected cohorts, teachers, and classrooms. Figure 1 provides an excerpt of this worksheet for PDA.

Course ID	Course Type	Course Name	Cohorts	Teachers	Classrooms	Meetings
GD 238	Class	Adavanced Publishing	GD-2	Chapman	Lab 2	1
GD 131	Class	Adobe Illustrator 2	GD-1	Chesson	Lab 1	1
GD 132	Class	Adobe InDesign 2	GD-1	Forbes	Lab 1	1
GD 230	Class	Audio/Visual 2 Design	GD-2	Lussenburg	Lab 2	1
BT 137 / ID 234	Class	BC Building Code 2	BT-1, ID-2	Morhart	Lecture	1
FD 136	Class	Business of Fashion	FD	Ferreira	Lecture	1
GD 232	Class	Business of Graphic Design	GD-2	Chesson	Art	2

Fig. 1: Excel Worksheet Listing the Set of Events.

While most rows in this Excel worksheet have "Class" as their Course Type, we can also include "Day Off" and "Prep Time" to denote events such as teachers needing a day off, or one or more teachers requiring a common period to plan together. Since no teacher can be scheduled for two events in the same timeslot, our automated timetabling program guarantees off-days for certain teachers as well as ensuring that a set of teachers can have overlapping non-teaching timeslots in which to prepare for future lessons.

As a concrete illustration, SMUS required one part-time teacher (Teacher M) to have exactly one non-teaching day (i.e., no teaching for all seven periods on any one of the five days) in addition to at most three periods of teaching on each of the remaining four days. The administrator at SMUS informed us that Teacher M's non-teaching day could be any day between Monday and Friday, but that a non-teaching day was required for this teacher.

Course ID	Course Type	Course Name	Cohort	Teacher	Classroom	Set of Timeslots	Sign	Value
	Class			Teacher M		1-1, 1-2, 1-3, 1-4, 1-5, 1-6, 1-7	at most	3
	Class			Teacher M		2-1, 2-2, 2-3, 2-4, 2-5, 2-6, 2-7	at most	3
	Class			Teacher M		3-1, 3-2, 3-3, 3-4, 3-5, 3-6, 3-7	at most	3
	Class			Teacher M		4-1, 4-2, 4-3, 4-4, 4-5, 4-6, 4-7	at most	3
	Class			Teacher M		5-1, 5-2, 5-3, 5-4, 5-5, 5-6, 5-7	at most	3

Fig. 2: Excel Worksheet of the Set of Event Constraints.

For each row of the worksheet provided in Figure 2, our Python program generates E, the set of events that are consistent with the leftmost six columns. In Figure 2, the set E refers to the set of events taught by Teacher M that have Course Type = "Class". In other words, by labeling her off-day with a different Course Type, we do not violate the constraint that she can be assigned at most three events (i.e., classes) on each day between Monday and Friday.

We then created a 7-meeting course named NoTeacherM, with Course Type set to "Day Off". In the **Event Relationships** worksheet, we added a row to indicate that these 7 events (i.e., meeting i for Teacher M, for each $1 \leq i \leq 7$) must occur in 7 consecutive periods of the same day. This worksheet contains all the constraint options provided in the Family II subsection of our model.

Finally, the **Teacher Preferences** worksheet indicates information on when teachers would prefer to teach their classes during all the timeslots they are available. At PDA, each preference coefficient $PR_{m,c,d,p}$ was marked as 2 points whenever the teacher of course c *wanted* to teach on day d period p, and was marked as 1 point whenever that teacher *could* teach in that timeslot. (For GNS and SMUS, each PR coefficient was 1 as teachers could not indicate preferences.)

This five-worksheet Excel file serves as the input to our Python program. We now provide the key statistics for each of our schools, listing the number of cohorts, the total number of events (m, c) in the timetable, the number of rows in our Event Constraints worksheet, the number of rows in our Event Relationships worksheet, and the average total running time of our Python program on this input file over ten iterations. This information is provided in Table 1.

All calculations were made on a stand-alone laptop, specifically a 8GB Lenovo running Windows 10 with a 2.1 Ghz processor.

From the table above, we see that it took less than one minute to generate the 2023-2024 Master Timetables for these three educational institutions. All three institutions accepted and implemented our timetable, and have hired us to build their Master Timetable again in 2024-2025.

School Name	PDA	GNS	SMUS
Days in Timetable	5	10	5
Periods in Day	3	5	7
Total Timeslots	15	50	35
Cohorts at the School	8	9	12
Total Events Scheduled	91	348	402
Event Constraint Rows	22	74	134
Event Relationship Rows	16	20	38
Running Time (in seconds)	0.54	13.48	45.51

Table 1: Statistics for our Automated Timetabling Program.

Acknowledgments The authors thank the administrators of the three educational institutions cited in this paper. Specifically we thank Jamie Kemp and Isabel Yu (Pacific Design Academy), Denise Lamarche, Becky Anderson, and Richard Brambley (St. Michaels University School), and Samantha Goddard and Erin Dallin (Glenlyon Norfolk School).

References

1. Appleby, J., Blake, D., Newman, E.: Techniques for producing school timetables on a computer and their application to other scheduling problems. The Computer Journal **3**(4), 237–245 (1961)
2. Collins Dictionary: Definition of timetabling. https://www.collinsdictionary.com/dictionary/english/timetabling (2024), accessed: 2024-03-07
3. Cooper, T.B., Kingston, J.H.: The complexity of timetable construction problems. In: Burke, E., Ross, P. (eds.) Practice and Theory of Automated Timetabling. pp. 281–295. Springer Berlin Heidelberg, Berlin, Heidelberg (1996)
4. Even, A.S.S., Itai, A., Shamir, A.: On the complexity of timetable and multicommodity flow problems. SIAM Journal on Computing **5(4)**, 691–703 (1976)
5. Google OR-Tools: Fast and portable software for combinatorial optimization. https://developers.google.com/optimization (2024), accessed: 2024-03-07
6. Gotlieb, C.: The construction of class-teacher timetables. IFIP congress, Amsterdam **62**, 73–77 (1963)
7. Hoshino, R., Fabris, I.: Optimizing student course preferences in school timetabling. Proceedings of the 17th International Conference on the Integration of Constraint Programming, Artificial Intelligence, and Operations Research (CPAIOR 2020) pp. 283–299 (2020)
8. Hoshino, R., Fabris, I.: Partitioning students into cohorts during COVID-19. Proceedings of the 18th International Conference on the Integration of Constraint Programming, Artificial Intelligence, and Operations Research (CPAIOR 2021) pp. 89–105 (2021)
9. Lewis, R., Paechter, R., McCollum, B.: Post enrolment based course timetabling: a description of the problem model used for track two of the second international timetabling competition. Cardiff Working Papers in Accounting and Finance **A2007-3** (2007)
10. Müller, T., Rudová, H., Müllerová, Z.: Real-world university course timetabling at the international timetabling competition 2019. Journal of Scheduling pp. 1–21 (2024)
11. Pillay, N.: A survey of school timetabling research. Annals of Operations Research **218(1)**, 261–293 (2014)

12. Post, G., Di Gaspero, L., Kingston, J., Mccollum, B., Schaerf, A.: The third international timetabling competition. Annals of Operations Research **239** (2013)