A Preference-Based Faculty-Assignment Tool for Course Scheduling Optimization

Dr. Sami Khorbotly, Valparaiso University

Received the Bachelor of Engineering degree in Electrical Engineering from Beirut Arab University, Beirut, Lebanon in 2001. Received the M.S. and Ph. D. degrees both in Electrical and Computer Engineering from the University of Akron, Akron, OH in 2003 and 2007, respectively. Currently serves as a Chair and Professor of Electrical and Computer Engineering at Valparaiso University.

Daniel White, Valparaiso University

Daniel J. White is an Associate Professor in Electrical and Computer Engineering at Valparaiso University's College of Engineering, joining as an Instructor in 2013. He received the B.S. EE and M.S. EE in 2005 and 2006, respectively, and the Ph.D. in Ele

A Preference-Based Faculty-Assignment Tool for Course Scheduling Optimization

1 Introduction

Course scheduling is one of the most time-consuming tasks that department chairs must perform every academic semester. The course scheduling problem includes assigning a faculty member, the course time/day(s), and a classroom for each offered course. Course scheduling is an NP-hard problem that has been extensively studied over the years.

In this work, rather than addressing the NP-hard course scheduling problem in its entirety, we tackle only the faculty assignment side of the problem. This "divide and conquer" approach reduces the task from an NP-complete problem to a problem for which an optimal solution exists. The goal here is to prioritize assigning the best possible faculty member for each course. Once that is accomplished, class times and locations can be later assigned using other tools. Faculty assignment is prioritized because we believe that no factor is nearly as important as having the most suitable professor teach each course. The time of a qualified professor is by far the most valuable academic resource, above the limited time slots and the limited spatial resources available on campus.

Our faculty-assignment optimization tool uses Linear Programming (LP) with the objective function being the maximization of the overlap between the courses to be offered in a semester and the faculty members' preferences and skills. This maximizes the chances of every faculty member teaching courses they are interested in. A set of constraints is created to ensure the full coverage of all courses/sections to be offered and also to ensure that no faculty member is assigned to teach more than a pre-determined teaching load limit. The tool is embedded in a web-based application and is available for the public to use.

One of the greatest features of the tool is its objectivity. It generates the faculty-course assignments based on the faculty preferences. It does not favor one faculty member over the other. Disgruntled faculty members who are not pleased with the outcome can no longer be upset with the chair of the department. Additionally, the tool also helps identify structural holes in the department's depth of coverage across topics, prompting strategic staffing discussions and guiding future faculty searches. The paper explains how to use the tool and includes some scheduling results for the sake of demonstration. The paper also includes a link for interested future users to access the free, web-based version of the tool to find optimized solutions to their own scheduling problems. The source code is also made available for anyone who may be interested in a modified version of the tool.

2 Literature Review

Course scheduling (also known as timetabling) is a multi-parameter combinatorial optimization problem with multiple constraints. The created schedule must take into consideration many parameters and constraints including teacher expertise and preferences, student need for classes, best (non-conflicting) times for both teachers and students, and (for in-person courses) the availability of a suitable classroom with adequate teaching equipment. The overall problem is an NP-hard problem [1] that has been extensively studied over the years. A wide variety of solutions have been suggested. For example, [2] used genetic algorithms to solve the problem while [3] used simulated annealing. Similarly, [4] used the Particle Swarm Optimization (PSO) method while [5] solved the problem using a Tabu search. All of these efforts attempted to find an acceptable solution while optimizing all the parameters of the problem. A different approach was taken in [6], where the focus is on maximizing the faculty preference when it comes to the courses to teach and the assigned time-blocks. Similarly, [7] focused on the optimization of the faculty assignment. However, their approach was to use the Depth-First Search algorithm which assigns one course at a time before moving to the next one. Our approach, using Linear Optimization takes a wholistic approach to optimize the assignment of all the courses.

As a student-first, undergraduate program, the College of Engineering at Valparaiso University prioritizes the learning experience of our students and the achievement of their learning outcomes. To this end, while the time of a course and the location of a classroom are important parameters, we believe that nothing impacts a learning experience more than the quality of the instructor. This has been extensively studied and documented in the teaching and learning literature. For example, you can find the following statements in [8] - “In fact, without effective teacher guidance and instruction in the classroom, learning cannot be achieved” and [9] – “since the instructor is responsible for most aspects of the college course, including the tone of the classroom, the types of activities assigned, and interactions within the classroom, that instructor plays a major part on the students’ learning.” As a result, our tool focuses exclusively on assigning the best faculty member to every offered course.

3 The Tool

The preference-based faculty-assignment tool uses the linear programming optimization method and is coded in Python. The linear optimization portion is implemented using PuLP, which is a linear programming modeler written in Python [10]. The tool is published as a web application and can be found at the address: https://apps.dgnd.net/scheduling-solver/. The source code for the tool is available at https://github.com/wiredlab/scheduling-solver. A snapshot of the user interface of the tool is shown in Figure 1.

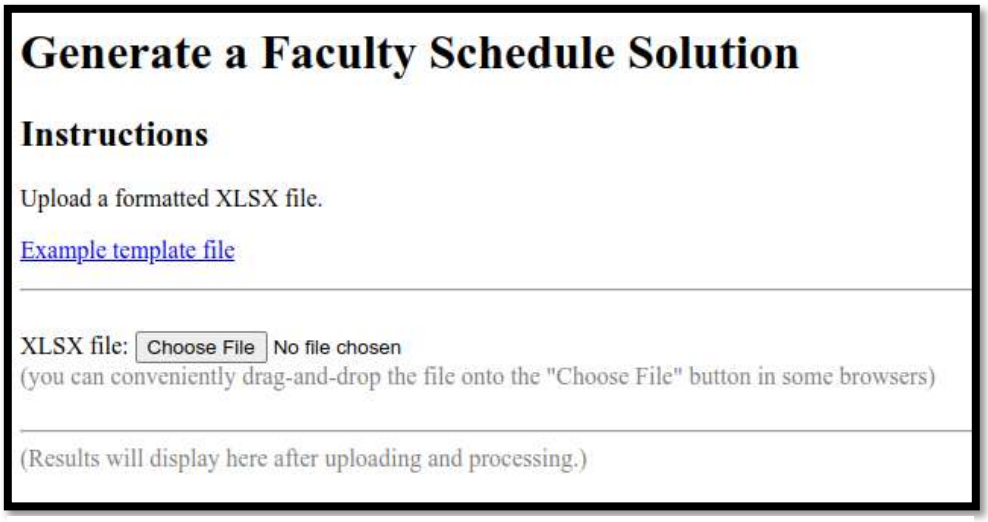

Figure 1. The user interface of the scheduling tool.

3.1 Configuration input

The only input a user needs to provide is an Excel spreadsheet that includes a list of the courses/sections to be covered in the semester, the teaching load associated with each course/section, a list of the faculty members to be assigned to those courses with the available load credits for each of them, and the preferences of each faculty member to teach every course on the list. An empty template of the spreadsheet is available to download from the webpage of the app and is shown in Figure 2, truncated to fit the space.

In the template, one can see that row 1 lists all the courses that are scheduled for offering in the current term. Row 2 (TLC/Section) is the number of Teaching Load Credits (TLC) associated with each course section. For example, a faculty member gets 1 TLC for teaching a section of the 1-Credit ECE 211 course while a faculty member who is teaching a section of the 3-Credit ECE 450

	A	B	C	D	E	F	P	Q	R	S
1		TLC_capacity	GE100	ECE211	ECE221	ECE221L	ECE450	ECE471	GE497	
2	TLC/Section		4	1	2.5	1.5	3	3	3	
3	# of Sects		2	2	1	2	1	1	3	Total TLCs
4	Total TLC		8	2	2.5	3	3	3	9	63
5	ProfA	12	9	7	5	5	5	1	8	
6	ProfB	6	5	7	10	10	6	3	8	
7	ProfC	9	7	8	5	8	8	1	9	
8	ProfD	12	10	5	5	8	10	1	9	
9	ProfE	12	3	8	3	1	5	1	2	
10	ProfG	12	8	7	5	8	1	8	5	
11	ProfH	3	10	10	8	8	1	10	1	
12										
13		66								

Figure 2. A sample of the user-input spreadsheet.

course gets 3 TLCs. Row 3 of the spreadsheet shows the number of sections to be scheduled for each offered course. For example, 2 sections of GE100 will be offered but only 1 section of ECE 221. Row 4 shows the total number of TLCs needed for a course. This is calculated by multiplying the number of sections by the number of TLCs per section for each course. At the end of the row, the user can see the total number of TLCs to be covered by the department in that academic term. Content in the white background cells is ignored by the tool.

Column A lists the names of the professors who are scheduled to teach in the term. Column B lists the TLC capacity for each professor. For example, Prof A is teaching a full load (12 TLCs) while Prof B, also serving as the department chair, is only available for 6 TLCs. Similarly, Prof C has a lower TLC capacity of 9. This is due to an endowed professorship that comes with 25% teaching load release. Finally, Prof H is an adjunct instructor with a TLC capacity of only 3. Any user can easily add or remove courses or professors by adding or removing columns / rows in the shaded regions.

3.2 Teaching preferences matrix

The remaining columns (C - R) correspond each to a course that is scheduled for offering. Figure 2 does not show all the columns to fit the width of the page. The numerical values in these columns (Cells C5 to R11) represent an instructor's interest to teach a course. These values are obtained by asking every instructor to indicate their interest in teaching each of the listed courses using a number $t_{ij} \in [1 - 10]$, where 10 indicates extreme interest in teaching the course and 1 indicates no interest in teaching it — a value of zero does not interact well with the tool and would effectively mean never assign this course. The preference numbers are arranged in the two-dimensional matrix T as follows:

$$T = \begin{bmatrix} t_{11} & \cdots & t_{1L} \\ \vdots & \ddots & \vdots \\ t_{N1} & \cdots & t_{NL} \end{bmatrix} \tag{1}$$

where t_{ij} represents the interest of instructor i to teach course j.

3.3 The Linear Programming Optimization

The first step in the optimization process is to investigate the feasibility of a solution. This is done by checking that the number of total TLCs to be covered (cell S4) is less than the TLC capacity of the available instructors (cell B13). No solution exists if this condition is not satisfied. If a solution is feasible, the problem is modeled by defining the variable X as an $(N \times L)$ matrix where N is the number of available instructors and L is the number of courses to be covered.

$$X = \begin{bmatrix} x_{11} & \cdots & x_{1L} \\ \vdots & \ddots & \vdots \\ x_{N1} & \cdots & x_{NL} \end{bmatrix} \tag{2}$$

In this variable matrix, x_{ij} is the number of sections of course j that are assigned to instructor i. For example, $x_{ij} = 1$ means that instructor i is assigned to teach one section of course j while $x_{ij} = 3$ means that instructor i is assigned to teach three sections of course j. Naturally, X is expected to be a sparse matrix. The LP optimal solution is achieved by maximizing the objective function F defined as the dot product of the two matrices T and X

$$F = T \cdot X = \sum_{j=1}^{N} \sum_{i=1}^{L} t_{ij} \cdot x_{ij}. \quad (3)$$

One can see that F is the weighted sum of the values denoting the interest of instructor i to teach course j, only if instructor i is assigned to teach one or more sections of course j. In other words, maximizing F leads to assigning courses to the instructor who have greater interest in teaching them.

When it comes to the constraints, two sets of constraints must be considered. The first set ensures that every section is assigned to an instructor. This set can be modeled by the following L equations:

$$\sum_{i=1}^{N} x_{il} = SEC_l \,, \forall\, l \in [1 - L] \quad (4)$$

Where SEC_l indicates the total number of sections of course l that must be offered (row 3 in Figure 2). This set of L equations ensures that, for each course l of the L courses, the sum of the assigned sections across all N instructors is equal to SEC_l.

The second set of constraints ensures that the sum of the TLCs corresponding to all the courses/sections assigned to an instructor does not exceed their TLC capacity. This set can be modeled by the following N equations:

$$\sum_{i=1}^{L} x_{ni}.TLC_i \leq TLC_{CAP,n} \,, \forall\, n \in [1 - N] \quad (5)$$

Where $TLC_{CAP,n}$ indicates the maximum number of TLCs that instructor n may teach (column B in Figure 2). These N equations ensure that, for each instructor n of the N instructors, the sum of their assigned TLCs across all courses does not exceed their maximum capacity $TLC_{CAP,n}$.

3.4 The output

The output of the system is a list of the teaching assignments for all available faculty members in a way that maximizes the objective function while satisfying all the constraints. A sample output of the system is shown in Figure 3. This output was obtained when the spreadsheet shown in Figure 2 was uploaded to the web app or given as a command line argument. In this sample, a total of 24 course sections 63 TLCs) are assigned to the 7 faculty members in the department.

The results show that Prof A is assigned to teach 2 sections of the course GE100, and also each of ECE322 and its lab ECE322L. Prof B is assigned the one section of ECE221 and both ECE221L lab sections, these also match their highest preference course. Adjunct Prof H is assigned two sections of ECE211 for a total of 2 TLCs.

It is important to note that the number of TLCs assigned to Prof C is 8, slightly lower than their TLC capacity of 9. The same is true with Prof E who is assigned only 11/12 TLCs and Prof H who is assigned 2/3 TLCs. That is not surprising because, going back to the input spreadsheet in Figure 2, the number of TLCs to cover (63) is slightly less than the total TLC capacity (66). This may be acceptable in most departments as it offers some flexibility and also because it is not always possible to have a perfect 100% load distribution efficiency. Conversely, a department chair may look at the 63/66 TLC mismatch and conclude that an adjunct professor, Prof H in our case, may not be needed for this particular semester.

Results from **Sample.xlsx** generated on 2024-02-04 22:44:04 GMT:
Optimizer status: Optimal

ProfA (12 / 12)
2 - GE100
1 - ECE322
1 - ECE322L

ProfB (6 / 6)
1 - ECE221
2 - ECE221L

ProfC (8 / 9)
1 - ECE340
2 - ECE340L
1 - ECE424

ProfD (12 / 12)
1 - ECE450
3 - GE497

ProfE (11 / 12)
2 - ECE251
1 - ECE360

ProfG (12 / 12)
1 - ECE263
2 - ECE263L
1 - ECE471

ProfH (2 / 3)
2 - ECE211

Figure 3. Optimization results output.

4 Summary

In this paper, we presented an LP-based faculty assignment optimization tool to help schedule makers with the faculty assignment aspect of the course scheduling problem. The tool is embedded in a web application and is available for public use. The tool takes as an input a spreadsheet including the classes to be covered, the teaching load capacity of every professor, and the amount of interest every professor has to teach every class. The output of the tool is a list of faculty assignments that maximizes the class-interest overlap while covering all classes without exceeding any professor's teaching capacity limit. It is important to note that the tool user has the choice of either using the output as is or tweaking it for other considerations if they choose.

5 References

[1] M. Elmohamed, P. Coddington, and G. Fox, "A comparison of annealing techniques for academic course scheduling," International Conference on the Practice and Theory of Automated Timetabling. Berlin, Heidelberg. 1997.

[2] Y-Z. Wang, "Using genetic algorithm methods to solve course scheduling problems," *Expert Systems with Applications*, vol. 25, 2003.

[3] E. Aycan and T. Ayav, "Solving the Course Scheduling Problem Using Simulated Annealing," *2009 IEEE International Advance Computing Conference*, Patiala, India, 2009.

[4] D-F. Shiau, "A hybrid particle swarm optimization for a university course scheduling problem with flexible preferences," *Expert Systems with Applications,* vol. 38, 2011.

[5] R. Alvarez-Valdes, E. Crespo, and J. Tamarit, "Design and implementation of a course scheduling system using Tabu Search," *European Journal of Operational Research*, vol. 137, 2002.

[6] M. Badri, D. Davis, D. Davis, and J. Hollingsworth, "A multi-objective course scheduling model: Combining faculty preferences for courses and times," Computers & Operations Research. Vol. 25, 1998, pp. 303-316.

[7] M. Wyne, A. Farahani, E. Atashpaz-Gargari, and L. Zhang, "Optimal Faculty Staffing Using Depth-First Search," ASEE Annual Conference and Exposition. Baltimore, MD. 2023.

[8] L. Kyriakides, C. Christoforou, and C. Charalambous, "What matters for student learning outcomes: A meta-analysis of studies exploring factors of effective teaching," *Teaching and Teacher Education*, vol. 36, 2013, pp. 143-152.

[9] B. Trammell, & R. Aldrich, "Undergraduate Students' Perspectives of Essential Instructor Qualities," *Journal of the Scholarship of Teaching and Learning*, vol. 16. 2016.

[10] https://pypi.org/project/PuLP/