Resilient Structures and Sustainable Construction
Edited by Pellicer, E., Adam, J. M., Yepes, V., Singh, A., and Yazdani, S.

ISBN: 978-0-9960437-4-8

A LINEAR PROGRAMMING APPLICATION AND SOLUTION FOR MINIMIZING CLASS SCHEDULING CONFLICTS

MELISSA HUMPHREY[1] and AMARJIT SINGH[2]

[1]*Layton Construction Company, Honolulu, USA*
[2]*Dept of Civil and Environmental Engineering, University of Hawai'i at Mānoa, Honolulu, USA*

College course scheduling plays a pivotal role in a student's development, academically and professionally. Schools without a standardized class scheduling system can generate many scheduling conflicts – some with the potential to seriously impact the student's course progression, and thus delay their graduation dates. The optimization model proposed in this article can identify and eliminate those scheduling conflicts through the use of visual modeling and linear programming. Constraints for this model were taken from the individual department's requirements and prerequisites, and instructor preferences, while allowing for the maximum number of available classes by minimizing and/or eliminating the number of class overlaps. An iterative process of analyzing and improving the schedule between visual modeling and linear programming enables the optimal result to be attained. This modeling system was applied to eight different semester schedules at an ABET-accredited university. The applied methodology yielded results of improved scheduling by an average of 83.46% over the original schedule, with a statistical confidence of 95.14%. By reducing the overall number of possible scheduling conflicts, it provides the students with more options and the ability to succeed. This type of class scheduling technique is not known to exist outside this study. The success rate of this technique far outstrips any success rate by any other documented method.

Keywords: Constraints, Objective function, LINGO, Manual adjustment, Visual model, Course schedule, Input values.

1 INTRODUCTION

A common problem several Universities face is the appropriate scheduling of courses to best serve its student population. A poorly composed schedule with several overlaps and scheduling conflicts can greatly impact the students' academic and professional endeavors. The purpose of this study was to identify a scheduling model that would provide an optimal schedule with the fewest amount of overlaps and conflicts.

Somewhat similar – though not identical – applications of this method have been used in other university scheduling studies. Chin-Min University in Taiwan (Chen 2008) factored in the professor's seniority and preferences when forming the constraints for the linear programming model. Another study at the University of Patras (Daskalaki *et al.* 2004) accounted for the student schedules in order to minimize the number of classroom changes. A third study at Kuwait University prioritized class scheduling according to student and faculty gender

(Al-Yakoob and Sherali 2006). The constraints that were prioritized in those studies were not utilized in this study as they were not found applicable to the United States.

2 LINEAR PROGRAM DEVELOPMENT

Linear programs require an objective function, input variables, and carefully crafted constraints which direct the program towards providing an optimal desired solution. Practical constraints have to be applied in the formulation of this linear program, such as for room availability and keeping classes on their originally scheduled days. There was also some intent to keep class scheduling either within the morning block of classes or the afternoon block, based upon the professor's preference. In order to simplify this function, a single semester schedule was analyzed and broken into two linear programs according to Monday-Wednesday-Friday (MWF) and Tuesday-Thursday (TR) schedules. A small sample of the MWF schedule used for this study can been seen in Table 1.

Table 1. Truncated sample of Fall 2015 written schedule.

Dept	Course	Sec	Hrs	Course Title	Days	Begins	Ends	Instructor
CEE	270	1	3	Applied Mechanics 1	MWF	8:30	9:20	Prof. A
CEE	330	1	4	Environmental Engineering	MWF	10:30	11:20	Prof. G
CEE	330L	1		Lab	M	12:30	2:20	Prof. G
CEE	330L	2		Lab	W	12:30	2:20	Prof. G
CEE	330L	3		Lab	F	12:30	2:20	Prof. G
CEE	473	1	3	Construction Equip. & Materials	MWF	12:30	1:20	Prof. R
CEE	491	1	3	Loads on Structures	MWF	12:30	1:20	Prof. A
CEE	687	1	3	Prestressed Concrete	WF	3:00	4:15	Prof. R

The obejctive function for this scheduling program was a minimization function, assigned to reduce the overall number of conflicts within a schedule. The input values are created by assigning a variable for each individual class for each possible time slot throughout the day – broken into ten different one-hour time slots, starting at 7:30 am. Undergraduate classes were assigned an x_{ij} variable, and graduate classes were assigned a y_{ij} variable. An example of the input variable compilation can been seen in Table 2.

The constraints applied to the linear program are what define the calculations in order to provide useful and relevant output values. This study utilized seven different constraints in order to create a schedule with fewer schedule conflicts. The first three constraints eliminate overlaps within the 300 level (sophomore-junior), 400 level (senior), and 600 level (graduate) courses. The fourth and fifth constraints identified professors who taught multiple courses per semester, and made it impossible for the program to schedule their classes during the same time slot or back to back. The sixth and seventh constraints addressed lab sections – ensuring that they weren't scheduled during the accompanying lecture time slot, and that the linear program allowed for these longer class sections to be scheduled during conescutive time slots. To formulate a constraint, the variables associated with the indicated classes are added and made to less than or equal to one. Each variable is assigned a value of one by the linear program, which is what ensures that only one class would be selected for any given time slot. An example of constraints 1 and 4 can be seen in Tables 3 and 4.

Once the input values and constraints are formulated and applied to the objective function, producing a series of output values. The output values in the LINGO software used are read as either a 0 or a 1. Those receiving a value of one are deemed as the optimal time slot for that specific class. The output values were analyzed, and if deemed optimal by the program, were applied to the visual model. Once in the visual model, the number of conflicts are analyzed and indicate what modifications need to be made to the constraints in order to reduce the overall conflicts. An example of this process will be explained in further detail in the following section.

Table 2. Truncated sample of Fall 2015 objective function (MWF, Fall 2015).

			7:30-8:20	**8:30-9:20**	**9:30-10:20**	**10:30-11:20**	**11:30-12:20**	**12:30-1:20**	**1:30-2:20**	**2:30-3:20**
			x_{i1}	x_{i2}	x_{i3}	x_{i4}	x_{i5}	x_{i6}	x_{i7}	x_{i8}
Course	*Instructor*		**MIN**							
270	*Prof. A*	x_{1j}	x11 +	x12 +	x13 +	x14 +	x15 +	x16 +	x17 +	x18 +
330	*Prof. G*	x_{2j}	x21 +	x22 +	x23 +	x24 +	x25 +	x26 +	x27 +	x28 +
330L1	*Prof. G*	x_{3j}	x31 +	x32 +	x33 +	x34 +	x35 +	x36 +	x37 +	x38 +
330L2	*Prof. G*	x_{4j}	x41 +	x42 +	x43 +	x44 +	x45 +	x46 +	x47 +	x48 +
330L3	*Prof. G*	x_{5j}	x51 +	x52 +	x53 +	x54 +	x55 +	x56 +	x57 +	x58 +
473	*Prof. R*	x_{18j}	x181 +	x182 +	x183 +	x184 +	x185 +	x186 +	x187 +	x188 +
491	*Prof. A*	x_{23j}	x231 +	x232 +	x233 +	x234 +	x235 +	x236 +	x237 +	x238 +
687	*Prof. R*	y_{4j}	y41 +	y42 +	y43 +	y44 +	y45 +	y46 +	y47 +	y48

Table 3. Truncated sample of 300 overlapping constraint (Fall 2015, MWF).

	CEE 270	**CEE 330**	**CEE 330L1**	**CEE 330L2**	**CEE 330L3**		
7:30-8:20	x11 +	x21 +	x31 +	x41 +	x51	< =	1
8:30-9:20	x12 +	x22 +	x32 +	x42 +	x52	< =	1
9:30-10:20	x13 +	x23 +	x33 +	x43 +	x53	< =	1

Table 4. Truncated sample of professors scheduled at different times constraint (Fall 2015, MWF).

	CEE 473	**CEE 687**		
7:30-8:20	x181 +	y41	< =	1
8:30-9:20	x182 +	y42	< =	1
9:30-10:20	x183 +	y43	< =	1

Note: Professor R, 473 and 687.

3 EXAMPLE: MWF FALL 2015

The original schedule for Fall 2015 had twelve class overlaps – ten in the undergraduate classes, and two within the graduate classes – as illustrated in Figure 1. After two iterations of the proposed methodology, the overall number of conflicts was reduced to four. A flowchart detailing the methodology applied in this process is shown in Figure 2.

The initial trial of the Fall 2015 linear program was deemed infeasible by the software. This was due to the fact that there were so many classes and only ten allowable time slots. Hence, there was no way to not allow for some class overlaps.

The DEBUG feature identifies which constraints are too rigid, and thus need to be reformulated or compromised in order to allow for a solution. By following the steps dictated by the DEBUG feature, an “optimal” solution was achieved by the program and transferred to the visual model. This first trial yielded sixteen conflicts between the MWF and TR schedules, all within the undergraduate category. However, the original two graduate class conflicts were resolved, and were therefore excluded from the second linear program iteration.

Upon further analysis, the primary source of conflicts generated were seen to be due to the lab sections, as there were three two-hour lab sections per lecture per week, which made it nearly

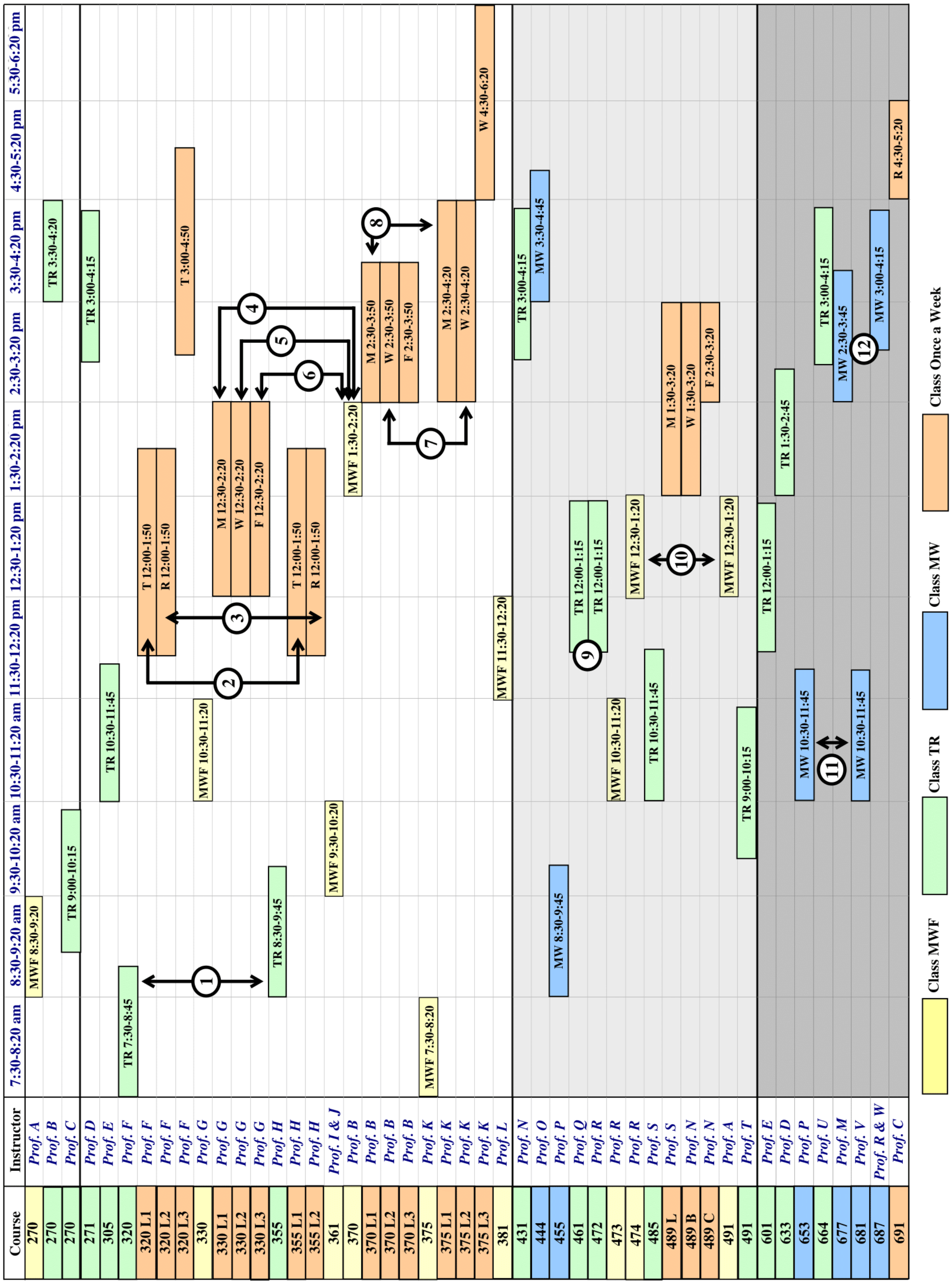

Figure 1. Original visual model, Fall 2015 schedule.

impossible for the linear program to eliminate overlaps. However, each lab section is only held one day a week, rather than lectures which occur three times a week. By combining the lab sections into one variable, it simplified the program by minimizing the number of variables it had to account for in terms of scheduling and overlaps. This combination of lab sections into one variable meant that it would be treated similarly to a lecture, in that the labs would be scheduled for the same time slot on each day rather than be scheduled independently of each other. It was upon the discrection of the visual modeller to distinguish these labs from each other when inputting the values into the visual model and assign class values of L1, L2, and L3 in the visual model. An example of this revised input value and objective function combining labs under one variable can be seen in Table 5 (compare to Table 2).

After making this change to the linear program, the second trial reduced the number of conflicts from twelve to ten. Once imported into the visual model, it was discovered that several of these conflicts could be avoided if adjusted manually by moving classes either earlier or later – some by a matter of one time slot in either direction. This manual adjustment process eliminated six conflicts from the schedule, leaving four conflicts in the Fall 2015 schedule. The linear programming methodology provides a logical and objective scheduling starting point, and provides the scheduler with the most optimal schedule that can then be easily adjusted manually.

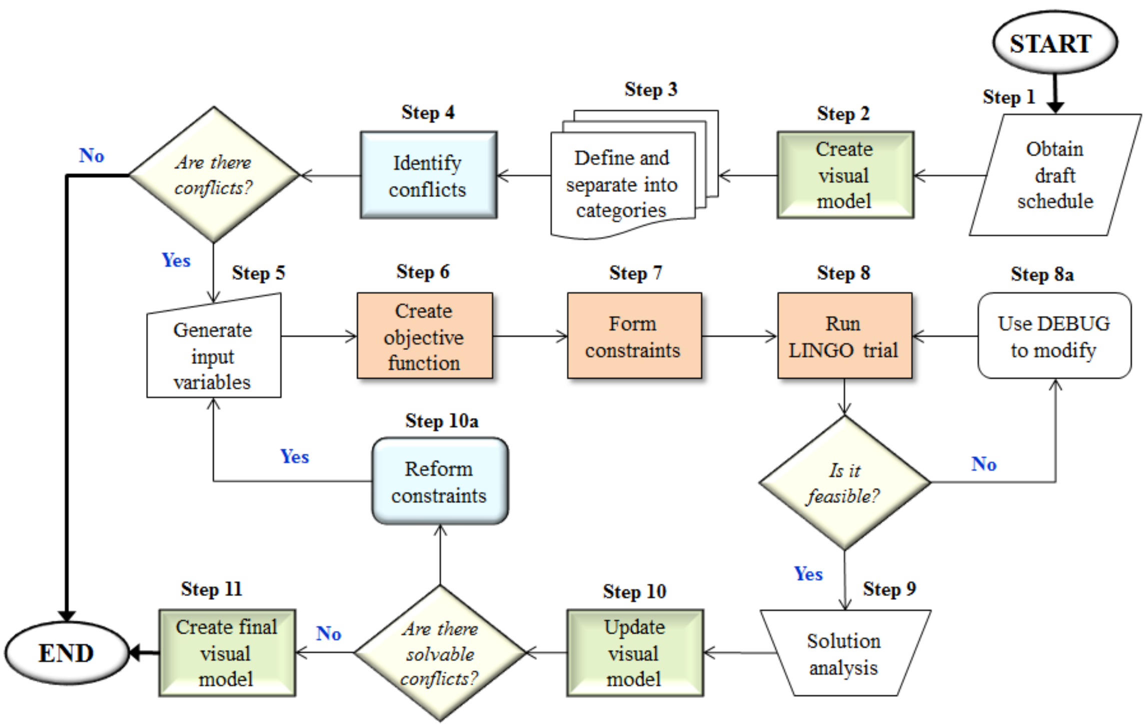

Figure 2. Methodology flowchart.

Table 5. Sample of Fall 2015 objective function (MWF, Fall 2015).

			7:30-8:20	**8:30-9:20**	**9:30-10:20**	**10:30-11:20**	**11:30-12:20**	**12:30-1:20**	**1:30-2:20**	**2:30-3:20**
			x_{i1}	x_{i2}	x_{i3}	x_{i4}	x_{i5}	x_{i6}	x_{i7}	x_{i8}
Course	*Instructor*		**MIN**							
270	*Prof. A*	x_{1j}	x11 +	x12 +	x13 +	x14 +	x15 +	x16 +	x17 +	x18 +
330	*Prof. G*	x_{2j}	x21 +	x22 +	x23 +	x24 +	x25 +	x26 +	x27 +	x28 +
330L1, 2, 3	*Prof. G*	x_{3j}	x31 +	x32 +	x33 +	x34 +	x35 +	x36 +	x37 +	x38 +
473	*Prof. R*	x_{18j}	x181 +	x182 +	x183 +	x184 +	x185 +	x186 +	x187 +	x188 +
491	*Prof. A*	x_{23j}	x231 +	x232 +	x233 +	x234 +	x235 +	x236 +	x237 +	x238

4 ANALYSIS AND CONCLUSION

By comparing the number of original conflicts to the number of conflicts remaining after applying the proposed methodology, it provides a statistical analysis proving the validity of this application to course scheduling. In this example alone, eliminating eight conflicts from the schedule improved the Fall 2015 schedule by 67%. This methodology was applied to eight semester schedules, producing an average improvement of 85.13% with a confidence limit of 95.14%. This methodology removed conflicts in every semester tested, and in some cases, eliminated all of the scheduling conflicts completely.

This linear programming application is flexible, can be easily adjusted and applied to any semester schedule, and can provide an optimal solution designed to meet the respective requirements and limitations. The visual modelling methodology aides the scheduler in identifying and possibly eliminating conflicts from the schedule, and helps with the understanding of the outputs generated by the linear program. In some cases, several trials were required, but ultimately always led to schedules with less conflicts for students. Utilizing linear programming allows for the most optimal and logical schedule of classes, which can then by manually adjusted to better suit the faculty or the students if necessary. By reducing the overall number of scheduling conflicts, students have more class options in their academic pursuits. In addition, the aim of universities to assist in timely graduation of students moves one step closer.

References

Al-Yakoob, S. and Sherali, H., Mathematical Programming Models and Algorithims for a Class-Faculty Assignment Problem. *Eur. J. of Oper. Res.*, 173, 488-507, 2006.

Chen, Y. C., Using Integer Programming to Solve the School Timetabling Problem at Chin-Min Institute of Technology. *J. of Amer.Academy of Business, Cambridge.* 13, 1, 279, 2008.

Daskalaki, S., Burbas, T., and Housos, E., “An Integer Programming For a Case Study in University Timetabling.” *Eur. J. of Oper. Res.*, 153, 117-135, 2004.