# Comparing Exact and Heuristic Algorithms for a Course-Timetabling Problem

**Daniel S. Myers · Jordan Rickman · Jay Yellen · Ruzgar Zere**

## 1 A Dual-Objective Course-Timetabling Model

We present results in the development of a course-timetabling system that will be used to construct the complete schedule for the 2017-2018 academic year at Rollins College. Our system builds on a previous system that was used to construct the Fall 2011 schedule for the Science Division classes at Rollins College [Wehrer and Yellen(2014)].

The approach is based on a weighted-graph model. The endpoints of each edge correspond to course pairs that are likely to have students and/or faculty in common or require a common resource. Each edge has two weights that correspond to the two objectives of our model. The first, dubbed the *conflict penalty*, is incurred when the endpoints are assigned overlapping timeslots. The second weight, the *proximity penalty*, is incurred when the endpoints are assigned timeslots having a large gap on the same day. The overall objective is to find a coloring that minimizes a linear combination of the two penalties (currently $25 \cdot conflict + 1 \cdot proximity$) incurred across all edges.

Our presentation includes a description of a mixed-integer programming (MIP) formulation [Toffolo(2015)] for the vertex-coloring problem and a detailed comparison between the optimal solutions it produces and those obtained by various heuristic approaches. We also report on our results experimenting with genetic algorithms (GA) to adjust the relative weights of our heuristics, and difficulties encountered in scaling both the exact algorithm and the heuristic approaches.

Daniel S. Myers · Jay Yellen · Ruzgar Zere
Rollins College
Winter Park, FL , USA
E-mail: jyellen@rollins.edu

Jordan Rickman
University of California-Irvine
Irvine, CA, USA

In addition to testing on the actual course-scheduling problem at Rollins, we are comparing performance on a series of randomly generated instances created from the actual Fall 2015 schedule that was constructed manually by the college registrar. Using an actual schedule as a seed for the randomization was motivated by the desire to create test problems that are similar to the actual timetabling problem. We intend to make these datasets available for the timetabling research community

## 2 Approximate Scheduling Algorithms

The system that created the Fall 2011 Science Division schedule was based on a one-pass approximate algorithm that used a combination of various heuristics to assign timeslots and rooms. The heuristics include the total penalty of the schedule so far (i.e., the partial coloring of the graph) and a projection of the difficulty of coloring the remaining uncolored vertices.

The one-pass strategy is limited in that it generates only one path to a complete coloring without backtracking. [Rickman and Yellen(2014)] extended the one-pass system by incorporating a beam search approach that maintains a fixed-size priority queue of partial colorings [Bisiani(1987)]. At each step, the beam search selects the most promising partial coloring from the queue and expands it by generating a set of successor colorings, each of which includes one more colored vertex. If each expansion step selects $m$ vertices and assigns each vertex one of $n$ colors, the total number of successors (i.e., the *branching factor*) is $mn$.

Heuristics are used in various components of the beam search: ordering the priority queue; setting the branching factor; and selecting the child nodes that are generated during each expansion. The weights for the different heuristics were tuned using an evolutionary local search algorithm [Norvig(1992)]. The tuned beam search achieved the same result as the optimal MIP scheduler on the Fall 2011 Science Division problem.

## 3 Comparing Exact and Approximate Approaches

In preparation for constructing the complete schedule for Rollins College for the 2017-2018 academic year, we obtained the complete College schedule for Fall 2015, including records on student cross-enrollments in each pair of courses. The complete schedule contains 711 individual course sections with more than 300 instructors meeting in 157 distinct meeting times (e.g., MWF 8:00-8:50, MW 8:00-9:15, etc.). Using this data, we generated a set of 10 test problems by randomizing the acceptable timeslots, acceptable rooms, and student cross-enrollment weights between pairs of courses. The randomized problems were constructed by extending the original Fall 2015 schedule in such a way that every course has at least one valid timeslot and room assignment.

Table 1 summarizes the performance of the exact MIP scheduler (using the Gurobi solver [Gurobi(2016)]) on the 10 randomized problems based on the

**Table 1** Results of the MIP scheduler on the set of 10 randomized problems

| Problem | Total penalty | Heavy conflicts | Medium conflicts | Light conflicts | Unassigned rooms | Run time (minutes) |
|---|---|---|---|---|---|---|
| 1 | 12503 | 0 | 1 | 52 | 0 | 110 |
| 2 | 13386 | 0 | 2 | 40 | 0 | 9 |
| 3 | 14050 | 0 | 4 | 46 | 0 | 179 |
| 4 | 13456 | 0 | 1 | 50 | 0 | 13 |
| 5 | 15088 | 0 | 6 | 38 | 0 | 8 |
| 6 | 15100 | 0 | 5 | 42 | 0 | 73 |
| 7 | 15236 | 0 | 5 | 35 | 0 | 70 |
| 8 | 15101 | 0 | 5 | 43 | 0 | 8 |
| 9 | 14027 | 0 | 4 | 32 | 0 | 4 |
| 10 | 13971 | 0 | 2 | 40 | 0 | 15 |

Fall 2015 schedule. The optimal scheduler is able to place every course in an acceptable room and avoids creating any heavy conflicts. The average penalty over the 10 problems is 14138.

While the performance of the MIP scheduler on the 10 problems is good, but there are still compelling reasons to investigate heuristic schedulers. In particular,

– The run time of the MIP is highly variable: four of the 10 problems require over an hour to solve and one requires almost three hours.
– The efficiency of the scheduler depends on the power of the underlying MIP solver, but not all organizations performing scheduling have access to commercial packages like Gurobi.
– Our experience and discussions with the college's registrar have shown that the course schedule will need a series of edits after its initial construction. In 2011, this editing process was enhanced by re-running the scheduler to evaluate the impact that each potential change would have on the overall schedule. Even the fastest MIP run times in Table 1 are too slow to support this process.

The initial version of the one-pass system used a two-step process when scheduling each course, first selecting the most suitable timeslot, then selecting an acceptable open room at that timeslot. We found that this approach failed to construct good solutions on the 10 randomized problems. We obtained significant improvement by switching to an approach that assigns a timeslot-room pair to a course in a single step. We then further improved the system's performance by applying evolutionary search to tune the parameter settings in the timeslot-room selection heuristics.

Table 2 summarizes the results for the one-pass system on the set of 10 randomized problems. The average penalty over the 10 problems is 18488. The resulting schedules have no heavy conflicts, and only four courses are left without rooms out of the almost 7000 courses considered across the set of 10 problems. The one-pass system is fast–our current Python implementation constructs the entire 700-course schedule in under three seconds.

**Table 2** Results of the one-pass scheduler on the set of 10 randomized problems

| Problem | Total penalty | Heavy conflicts | Medium conflicts | Light conflicts | Unassigned rooms | Run time (seconds) |
|---|---|---|---|---|---|---|
| 1 | 17130 | 0 | 2 | 58 | 0 | 2 |
| 2 | 17295 | 0 | 4 | 55 | 0 | 2 |
| 3 | 16941 | 0 | 5 | 48 | 0 | 2 |
| 4 | 18406 | 0 | 2 | 50 | 1 | 3 |
| 5 | 19274 | 0 | 6 | 56 | 1 | 3 |
| 6 | 19145 | 0 | 7 | 54 | 0 | 2 |
| 7 | 18621 | 0 | 5 | 54 | 0 | 2 |
| 8 | 21122 | 0 | 5 | 50 | 2 | 3 |
| 9 | 18844 | 0 | 6 | 38 | 0 | 2 |
| 10 | 18102 | 0 | 2 | 49 | 0 | 2 |

**Table 3** Results of the beam search scheduler on the set of 10 randomized problems

| Problem | Total penalty | Heavy conflicts | Medium conflicts | Light conflicts | Unassigned rooms | Run time (seconds) |
|---|---|---|---|---|---|---|
| 1 | 15299 | 0 | 1 | 59 | 0 | 135 |
| 2 | 16838 | 0 | 3 | 53 | 0 | 125 |
| 3 | 17153 | 0 | 4 | 50 | 0 | 159 |
| 4 | 16135 | 0 | 2 | 46 | 0 | 139 |
| 5 | 19680 | 0 | 6 | 49 | 1 | 135 |
| 6 | 18973 | 0 | 6 | 55 | 0 | 145 |
| 7 | 18304 | 0 | 5 | 36 | 0 | 135 |
| 8 | 20084 | 0 | 6 | 44 | 0 | 148 |
| 9 | 16944 | 0 | 4 | 32 | 0 | 139 |
| 10 | 17246 | 0 | 2 | 59 | 0 | 121 |

The beam search scheduler is slower than the one-pass approach, but its optimal performance on the Fall 2011 Science Division schedule led us to apply it to the 700-course problem. Table 3 reports results on the 10 randomized problems for a version of the beam search scheduler that adapts the vertex and timeslot-room selection heuristics from the one-pass system. The heuristics for selecting the partial coloring to expand at each step have been tuned using evolutionary local search. The average over the 10 problems is 17665, which is within 25% of the optimal result obtained by the MIP scheduler, and all courses except one receive an acceptable room.

## References

[Bisiani(1987)] Bisiani R (1987) Beam search. In: Shapiro S (ed) Encyclopedia of Artificial Intelligence, John Wiley and Sons, pp 56–58

[Gurobi(2016)] Gurobi (2016) URL http://www.gurobi.com/index

[McCollum(2007)] McCollum B (2007) University timetabling: Bridging the gap between research and practice. In: Proceedings of the 5th International Conference on the Practice and Theory of Automated Timetabling, Springer, pp 15–35

[Norvig(1992)] Norvig P (1992) Paradigms of Artificial Intelligence Programming: Case Studies in Common Lisp, 1st edn. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA

[Qu et al(2009)Qu, Burke, McCollum, Merlot, and Lee] Qu R, Burke E, McCollum B, Merlot LT, Lee SY (2009) A survey of search methodologies and automated system development for examination timetabling. J of Scheduling 12(1):55–89

[Rickman and Yellen(2014)] Rickman J, Yellen J (2014) Course timetabling using graph coloring and a.i. techniques. In: Proceedings of the 9th International Conference on the Practice and Theory of Automated Timetabling, Springer

[Toffolo(2015)] Toffolo T (2015) Private communication

[Wehrer and Yellen(2014)] Wehrer A, Yellen J (2014) The design and implementation of an interactive course-timetabling system. Annals of Operations Research 218(1):327–345