Constructing Pareto-Optimal Residency Call Schedules

Amy Cohn

Dept. of Industrial & Operations Engineering
1205 Beal Avenue, University of Michigan
Ann Arbor, MI 48109-2117, USA

ABSTRACT

In constructing year-long call schedules for medical residents, a key challenge is the multi-criteria nature of the problem -- the quality of a schedule depends on a number of (often competing) metrics. One way to address this is to assign weights to each metric to trade off between them in the objective function. This introduces a new challenge, however: requiring the schedulers (here, the Chief Residents) to identify weights that accurately represent their preferences. In this paper, we instead present an efficient way to generate the complete set of Pareto-optimal solutions. The scheduler can then select a preferred schedule from this set, rather than trying to implicitly represent their preferences by weights. Computational results from a real-world residency scheduling problem demonstrate the tractability of this approach.

Keywords: Multi-criteria optimization, residency scheduling, integer programming, optimization, operations research.

1. INTRODUCTION

This paper focuses on the construction of a year-long call schedule for medical residents at a major U.S. teaching hospital. There are many things that make this problem challenging. First is the length of the planning horizon -- in order to ensure that all educational requirements are met for the residents, while also providing adequate patient care, it is necessary to plan the entire year in a single schedule. Second, there are multiple types of residents to be considered simultaneously (first-year, second-year, etc.), each with different requirements and capabilities. Third, these residents staff multiple hospitals which also have varying requirements. Finally, there are myriad rules (typically ACEME mandates) placing limits on the number of hours that can be worked in a given time period.

But even more difficult than enforcing these constraints is the challenge of defining an appropriate objective function: What constitutes the "best" schedule? On the one hand, there is no obvious cost function to minimize. On the other hand, that does not mean that all feasible schedules (i.e. schedules that satisfy all constraints are requirements) are equally good. For example, a schedule in which every resident is assigned to roughly the same number of Friday night calls over the course of the year is definitely preferable, in the eyes of the Chief Residents (who build and oversee the schedule), to a schedule in which some residents are assigned to ten Friday night calls and others are assigned to only one or two. But what if the former means decreasing the number of vacation requests that can be granted? Is it worse to have inequity in Friday call assignments or to increase the number of vacations denied? There are many such metrics that matter to the individual residents. In addition, the Chief Residents have the concern of maintaining equity across residents: not providing a highly desirable schedule to one resident at the expense of a very poor schedule for another.

There is a vast literature in multi-criteria optimization (see [9] for an excellent text). Much of this literature is devoted to healthcare scheduling, in which the use of weights in the objective function is quite common (e.g. [1], [2], and [8]). The key obstacle that remains with these approaches is, of course, how to identify the weights. It is difficult for people to answer accurately questions like "What is the relative impact of one extra Friday night call versus one extra vacation request denied?" Furthermore, these relationships are often non-linear -- increasing a resident's schedule from including one Friday night call to including two is potentially very different from increasing from twelve to thirteen calls. Ultimately, if the weights entered into the objective function do not accurately represent the true preferences, then the resulting schedule will not necessarily be optimal -- in fact, it may be far from it.

Our goal, then, is to present not one single schedule but instead the complete set of Pareto-optimal schedules: ones for which no other schedule that has equal or better value for all metrics. If we provide this set of schedules to the Chief Residents, we can assure them that there is no schedule not in the set that they would prefer. In addition, we can intuit information about their preferences from the way in which they evaluate the Pareto-optimal solutions, providing feedback to more tightly limit this set in the future.

The contributions of this research are two-fold. First, we formulate and solve a complex real-world problem in healthcare scheduling. Second, we present a more general algorithm for identifying this Pareto-optimal set efficiently.

The remainder of the paper is organized as follows. In Section 2, we define the residency call scheduling problem. In Section [3], we describe the set of Pareto-optimal solutions and present an efficient algorithm for generating this set of solutions. We provide computational experiments in Section [4] to demonstrate the tractability of our approach. Conclusions and suggested areas for future research are offered in Section [5].

2. PROBLEM DESCRIPTION

Doctors typically spend three to five years as residents after medical school, actively practicing while still under the supervision of more senior physicians. Residents often have both daytime responsibilities (patient care, seminars, etc.) and the requirement of being on call many nights over the course of the year. On-call shifts provide additional training to the residents while also providing nighttime staffing for the hospitals. The Chief Residents are responsible for building the call schedule so as to meet both of these needs.

In the problem that we consider (see [6] and [7] for more details), there are two general classes of residents who cover three different hospitals with four different shift types. The schedule spans a full year of 365 days. Hard constraints, defined by the Chief Residents as mandatory rules to be followed, include:

1) Each hospital must have adequate staffing on each night. Note that different hospitals have different requirements. Furthermore, within a given hospital, constraints can vary by day of week.

2) Each resident must fulfill a given number of calls over the course of the year. This number varies by resident.

3) A resident cannot be assigned to calls at two different hospitals on the same night.

4) A resident cannot be on call more than once in any four day period.

5) A resident cannot be assigned to more than five calls in any calendar month.

6) Each resident has specific days that s/he cannot be on call because of other commitments.

7) Each resident has been pre-assigned to work specific holidays; these assignments are in input to the scheduling process described here.

These constraints can be addressed by defining binary decision variables of the form x_{rhd} which takes value one if resident *r* is assigned to hospital *h* on day *d*. An example of how a hard constraint is enforced is

$$\Sigma_h x_{rhd} \leq 1 \text{ for all residents } r \text{ and days } d$$

which says that a resident cannot be assigned to more than one hospital on any given day.

In addition to the hard constraints, there are several preferences, i.e. metrics, to be taken into account. These include:

1) Each resident should complete a pre-specified number of calls (sub-divided further into weekday- and weekend- requirements) at each hospital. In practice, this may not be possible, in which case the same total number of calls should still be scheduled for each resident but some of these calls might be swapped between hospitals.

2) Residents can each provide vacation requests prior to the start of the scheduling process.

3) Residents prefer not to work Fridays or Saturdays.

4) Residents prefer that the hospital where they are on call at night matches the location where they are assigned for that day's daytime duties as well, so as to avoid extra commute time.

5) The Chief Residents prefer that the individual resident perceive their schedules as fair, relative to the schedules of the other residents.

3. IDENTIFYING PARETO-OPTIMAL SOLUTIONS

It is fairly easy to enforce the rules and compute the metrics outlined in Section [2] by formulating constraints in an integer programming (IP) framework (see [6] and [7] for more details). The challenge, as highlighted earlier, is in determining the objective function to use in comparing different feasible schedules. In our first experiences with this problem, we addressed this with the Chief Residents in an ad hoc, iterative fashion. We first constructed an arbitrary feasible schedule and sent it to them for feedback. They identified characteristics that they liked and disliked, then made suggestions for further improvement. For example, they might request that fewer "moonlighters" (physicians not in the residency program who can be hired on an ad hoc basis) be assigned to fill gaps in the schedule. We would attempt to fulfill these requests, pointing out the corresponding (and often negative) impact on other metrics. For example, hiring fewer moonlighters might mean having to deny more vacation requests. The Chief Residents would then decide whether or not to make the tradeoff, often suggesting alternatives to be evaluated in the model. This iterative process ultimately led to the development of a schedule that was viewed not only as acceptable but in fact as a significant improvement over those schedules constructed manually by the Chief Residents in the past.

This ad hoc approach is not sustainable in the long-term, however, and in subsequent research we have sought to find a way to automate this process without decreasing the quality of the final solution. In this paper, we outline this research. Specifically, we have chosen to eliminate the iterative feedback loop with the Chief Residents by instead providing them, in a single step, with the complete set of Pareto-optimal schedules from which to select. A Pareto-optimal schedule is one in which no improvement can be made in one metric without making another metric strictly worse. For example, if the two metrics are "number of moonlighters hired" and "number of vacation requests denied", with lowers numbers of each metric preferred, then a schedule with corresponding values (5, 8) is Pareto-dominant (i.e. "better", in the Pareto sense) relative to one with values (6, 10). However, we cannot say whether (5, 8) is better or worse than (6, 3) and thus must provide both schedules to the Chief Residents to choose between.

This approach is premised on the notion of "mutual preferential independence" ([10]), which suggest that, simply "less is better" in all categories. In this case, metrics have been defined to meet this criteria.

Our ability to generate the set of all Pareto-optimal solutions in a timely manner is based on three important facts:

1) Each of the metrics of interest in this problem takes on discrete and fairly limited values. For example, one key metric is the maximum number of Fridays any single resident is assigned to be on call. This metric takes only integer values and cannot be any less than three (this is the total number of Fridays in the year divided by the number of residents in the program) nor any more than thirteen (the total number of weekday calls a resident works in a year).

2) Each of the metrics can be expressed in such a way that higher values are always worse than lower values (e.g. "number of vacations denied" rather than "number of vacations granted").

3) If an upper bound is provided for each metric (e.g. "at most five Fridays per resident, at most eight vacations denied, and at most twelve moonlighters hired"), then solving the corresponding feasibility problem (i.e. determining whether a schedule exists that satisfies these upper bounds) is easy to do, using traditional integer programming techniques and off-the-shelf commercial solvers.

Based on these facts, we have developed an algorithm that generates the set of all Pareto-optimal solutions through an enumeration-based approach, using the following simple steps. [See [3], [4], and [5] for examples of our prior experience in using a related algorithm to solve problems in other application domains.]

1. Define a *metrics vector,* where each element of the vector represents the maximum value for a given metrics.

2. Enumerate all metrics vectors representing the combinations of valid values for all metrics.

3. For each vector in this list, solve an IP to assess the feasibility of satisfying the scheduling constraints subject to the upper bounds imposed by the metric values in the vector.

4. Report to the user (i.e. the Chief Residents) all metrics vectors (and, for each, a corresponding schedule) that are both feasible and Pareto-dominant relative to the other metrics vectors.

To improve the tractability of this approach, we further note the opportunities for pruning the list. First, whenever a metrics vector is feasible, we can disregard all other metrics vectors for which the values of all elements are the same or larger -- these would never be preferred by the user. Second, whenever a metrics vector is infeasible, we can disregard all other metrics vectors for which the values of all elements are the same or smaller -- these are more tightly constrained and thus will be infeasible as well. In our experience, we have observed that drawing from the middle of the list improves performance by increasing the amount of pruning that occurs.

We demonstrate this approach with a simple example in Figure 1. In this example, we assume three metrics, each of which can take values 0, 1, or 2. This results in twenty-seven metrics vectors in the initial list. Suppose that we solve the feasibility problem corresponding to upper bounds of 1, 1, and 1 on the three metrics, and that this problem is infeasible. Then we are able to eliminate this metrics vector from the list and, at the same time, can eliminate seven others ([0,0,0], [0,0,1], [0,1,0], [1,0,0], [0,1,1], [1,0,1], and [1,1,0]) by propagating the infeasibility to more tightly constrained problems. If we next test (1, 2, 1) and determine it to be feasible, we set this vector aside as a potential candidate solution. We are then able to prune three dominated solutions ([2,2,1], [1,2,2], and [2,2,2]), resulting in a new list of size fifteen. The algorithm proceeds accordingly until the list is emptied. Note that if, in a later step, (1, 2, 0) is found to be feasible, then the pending solution (1, 2, 1) will be discarded as dominated.

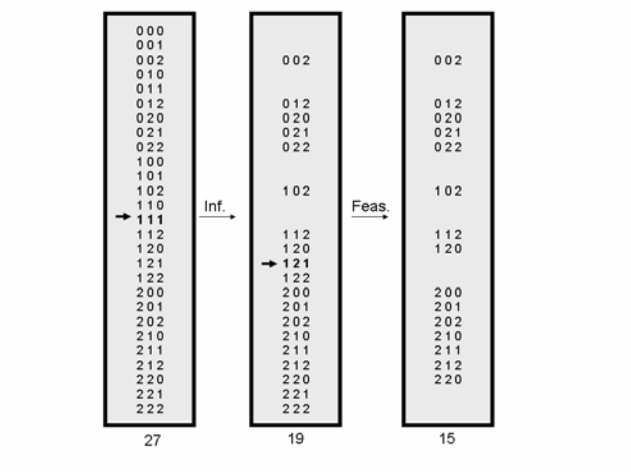

Figure 1: Example of algorithmic approach

4. COMPUTATIONAL EXPERIMENTS

To evaluate the computational performance of the proposed algorithm, we conducted two computational experiments. Both were based on data from a major U.S. teaching hospital in one of its corresponding residency programs.

In the first experiment we considered three metrics -- number of moonlighters hired during the winter holiday season, maximum number of Friday calls per resident, and maximum number of denied vacation request by resident. The initial list of candidate vectors contained 16,632 elements. Of these, only 807 feasibility problems needed to be solved; the remainder were pruned through dominance. These feasibility problems had on the order of 5,000 constraints, 8,000 binary variables, and 60,000 non-zero entries in the constraint matrix.

Ultimately, it was established that there is only one Pareto-optimal solution to this problem. This schedule was found (and proven to be optimal) in approximately 18 minutes.

Because the metrics in the first experiment did not conflict (i.e. the value that was optimal for each metric individually could also be achieved collectively), we conducted a second experiment where we knew conflicts would arise. Specifically, for each of fifteen residents, we considered a vacation request in either the Fourth of July week or the week between Christmas and New Year's, recognizing that granting some residents' requests would have to come at the expense of denying others. In this instance, because each request corresponds to a metric with two values (1 if yes, 0 if no), there are 2^{15} = 32,768 allocations to evaluate. Of these, only 8,864 were tested; the remainder were pruned. Of the 32,768 possibilities, 280 were shown to be Pareto-optimal. Total run time to identify the 280 Pareto-optimal schedules was approximately 6.6 hours.

5. CONCLUSIONS AND FUTURE RESEARCH

In this paper, we demonstrate a new way to solve a real-world residency scheduling problem, providing the Chief Residents with an exhaustive set of Pareto-optimal schedules from which to choose. Leveraging the discrete nature of the metrics used for evaluating the schedules, the fact that feasibility problems with fixed bounds on the metrics are easier to solve than optimization problems using weights in the objective function, and the opportunity to prune candidate metrics vectors both when the current vector is feasible and also when it is infeasible, we are able to solve real-world instances in acceptable run times.

In the future, we plan to expand this research in the following ways.

1) Extend the metrics vector definition to include all metrics of relevance to the Chief Residents.

2) Investigate other pruning opportunities for those cases where the number of metrics is large and thus dominance is more difficult to establish (leading to less pruning and therefore longer run times). In particular, we plan to investigate graph-based partitioning methods for the set of metrics vectors.

3) Evaluate effective ways to present the set of schedules to the Chief Residents when the Pareto-optimal set is large. We are also interested in developing feedback loops that allow us to learn more about the Chief Residents' preference functions and thus further prune the candidate set of schedules.

4) Apply the proposed algorithm to other discrete, multi-criteria optimization problems, particularly (but not exclusively) in healthcare scheduling.

6. ACKNOWLEDGEMENTS

The author wishes to acknowledge the assistance of Tian Jia Zhou, Ada Barlatt, Mary Jo Luppino, and Marshall Weir in the computational implementation of this research.

7. REFERENCES

[1] M. Azaiez and S. Al Sharif, "A 0-1 Goal Programming Model for Nurse Scheduling", **Computers and Operations Research**, Vol. 32, 2005, pp. 491 - 507.

[2] J. Bard and H. Purnomo, "Preference Scheduling for Nurses Using Column Generation", **European Journal of Operational Research,** Vol. 164, 2005, pp. 510 - 534.

[3] A. Barlatt, A. Cohn, Y. Fradkin, O. Gusikhin, and C. Morford, "Using Composite Variable Modeling to Achieve Realism and Tractability in Production Planning: An Example from Automotive Stamping", 2007, http://wwwpersonal.umich.edu/~amycohn/ PAPERS/ stamp.pdf.

[4] A. Barlatt, A. Cohn, M. Luppino, and T. Zhou, "A Parallel Algorithm for Solving Resource Allocation and Scheduling Problems", INFORMS Midwestern Conference 2007 http://www-personal.umich. edu/~amycohn/ CONFERENCES/Chicago.ppt.

[5] A. Barlatt, A. Cohn, Y. Fradkin, O. Gusikhin, and C. Morford, "A Hybridization of Mathematical Programming and Search Techniques for Integrated Operation and Workforce Planning", Proceedings from IEEE Conference on Systems, Man, and Cybernetics 2007, to appear.

[6] A. Cohn and S. Root, "Scheduling Medical Residents at Boston University School of Medicine", Working paper, 2007. http://www-personal.umich.edu/~ amycohn/PAPERS/Interfaces%20oncall.pdf.

[7] A. Cohn and S. Root, "Lessons Learned Applying Mathematical Programming Techniques to a Healthcare Scheduling Problem", Working paper, 2007, http://www-personal.umich.edu/~amycohn/ PAPERS/ joorscall.pdf.

[8] T. Dias, D. Ferber, C. de Souza, and A. Moura, "Constructing Nurse Schedules at Large Hospitals", **International Transactions in Operational Research,** Vol. 10, 2003, pp. 245-265.

[9] M. Ehrgott, **Multicriteria Optimization. Springer**, Berlin, 2005.

[10] H. Raifa and R. Keeney, **Decisions with Multiple Objectives: Preferences and Value Tradeoffs.** Wiley, 1976.