A flexible integer linear programming formulation for scheduling clinician on-call service in hospitals

David Landsman[a, b], Huiting Ma[*a], Jesse Knight[*a, c], Kevin Gough[d], and Sharmistha Mishra[a, c, d, e]

[a] *MAP Centre for Urban Health Solutions, St. Michael's Hospital, Unity Health Toronto, Toronto, ON, Canada*

[b] *Department of Computer Science, University of Toronto, Toronto, ON, Canada*

[c] *Institute of Medical Sciences, University of Toronto, Toronto, ON, Canada*

[d] *Department of Medicine, Division of Infectious Disease, St. Michael's Hospital, Unity Health Toronto, Toronto, ON, Canada*

[e] *Institute of Health Policy, Management and Evaluation, Dalla Lana School of Public Health, University of Toronto, Toronto, ON, Canada*

October 11, 2019

Abstract

Scheduling of personnel in a hospital environment is vital to improving the service provided to patients and balancing the workload assigned to clinicians. Many approaches have been tried and successfully applied to generate efficient schedules in such settings. However, due to the computational complexity of the scheduling problem in general, most approaches resort to heuristics to find a non-optimal solution in a reasonable amount of time. We designed an integer linear programming formulation to find an optimal schedule in a clinical division of a hospital. Our formulation mitigates issues related to computational complexity by minimizing the set of constraints, yet retains sufficient flexibility so that it can be adapted to a variety of clinical divisions.

We then conducted a case study for our approach using data from the Infectious Diseases division at St. Michael's Hospital in Toronto, Canada. We analyzed and compared the results of our approach to manually-created schedules at the hospital, and found improved adherence to departmental constraints and clinician preferences. We used simulated data to examine the sensitivity of the runtime of our linear program for various parameters and observed reassuring results, signifying the practicality and generalizability of our approach in different real-world scenarios.

1 Introduction

Hospital departments must allocate and use their limited resources efficiently in order to provide a high quality of care for their patients. In particular, on-call schedules for a fixed number of healthcare providers are central to the efficient running of hospitals. Carefully allocated schedules should balance sufficient staff with workload to maximize quality of care. It is common for on-call schedules in

[*] Contributed equally to this work.

hospitals to be created manually. However, manually-created schedules are subject to three problems. First, when there are a large number of clinicians, or the constraints that need to be satisfied by the schedule are complex, it becomes infeasible to find a satisfying schedule by hand. Second, when creating a schedule manually it is difficult to ensure that all constraints are met while also trying to satisfy all staff preferences. Third, manual scheduling is often time-consuming even for relatively small departments, and can take up time and resources that are better used for improving patient care. For these reasons, it is important to develop automated methods that can efficiently generate schedules that satisfy the given constraints.

Automated methods to generate schedules have been studied and applied in many industries, including transportation [1, 2, 3], manufacturing [4, 5, 6], retail [7, 8], and military [9, 10]. Of special interest to a clinician scheduling problem are the approaches to scheduling nurses, who often work in shifts. In the nurse scheduling problem, the goal is to find an optimal assignment of nurses to shifts that satisfies all hard constraints (such as hospital regulations), and as many soft constraints (such as nurse preferences) as possible. Hard constraints must be satisfied by any candidate solution to the nurse scheduling problem, while soft constraints can be used to rank the candidate solutions. For instance, a nurse scheduling problem may include a hard constraint to assign at most a single shift for each nurse per day. It can also incorporate nurse preference for shift time (that is, day versus night shifts) as a soft constraint that is meant to optimize the schedule, but is not guaranteed to be fulfilled. A wide variety of approaches, including exact and heuristic approaches, have been used to solve the nurse scheduling problem: integer linear programming [11, 12, 13], network flows [14], genetic algorithms [15, 16, 17], simulated annealing [18], and artificial intelligence [19, 20]. A comprehensive literature review of these and other methods applied to nurse scheduling is presented in [21].

Many of the approaches to nurse scheduling were designed to satisfy the requirements of a specific hospital department, which results in a large number of variables and constraints to be incorporated into the problem formulation. While these department-specific approaches allow end-users to find precise schedules that satisfy the needs of that department and the preferences of nurses and clinicians in that department, they are difficult to adapt to other departments. Moreover, the large number of variables and constraints also leads to computational complexity issues [22], especially when trying to find the most optimal solution. In particular, difficult instances of these formulations become impossible to solve in a reasonable amount of time. In this paper, we tackle the clinician scheduling problem arising from a case study of one clinical division, providing two different services (general infectious disease (ID) consults; and HIV consults) at St. Michael's Hospital in Toronto, Canada. The clinician scheduling problem involves creating a yearly schedule that assigns clinicians to on-call work on a weekly basis, while ensuring a fair and balanced workload. Our goals in this paper are to (1) present an integer linear programming (ILP) formulation for our problem, and describe the flexibility of this formulation for solving similar problems; (2) compare the performance of our tool for solving the ILP formulation to the results of a manual approach; and (3) analyze the robustness of this approach in difficult instances of the problem, by exploring the change in runtime with changes to: the number of clinicians, the number of services provided by the department, the number of requests per clinician, and the time-horizon of the schedule.

We begin by describing the details of the problem in Section 2, and presenting our ILP formulation in Section 3. Next, we compare the results of our formulation to manually-created schedules, and evaluate the performance of the algorithm on simulated data in Section 4. Finally, we discuss and interpret the results in Section 5.

2 Problem

At St. Michael's Hospital, the division of infectious diseases offers separate but concurrent services for general ID consultations and for HIV consultations. Each service provides clinical care throughout the year, during regular working hours and on weekends and holidays. The schedule is created in advance, outlining all work-week and weekend shifts for the full year. In the yearly schedule clinicians are assigned to "blocks" of two consecutive regular work weeks and individual weekends. Apart from long (holiday) weekends, a work week starts on Monday at 8 A.M. and ends on Friday at 5 P.M. Accordingly, weekend service starts on Friday at 5 P.M., and ends on Monday at 8 A.M. During the weekend, ID and HIV consultation services are combined and provided by one clinician. During the regular work week the ID and HIV services are led by one clinician each. Therefore, our objective is to assign a single clinician to cover each service for each block and each weekend of the year, while additionally ensuring a balanced workload.

In most scheduling problems, the constraints can be divided into hard and soft constraints. Hard constraints must be satisfied by any candidate solution, while soft constraints can be used to select a more favourable solution from a set of candidate solutions. Typically, soft constraints are therefore encoded as objective functions which are maximized when finding a solution. In the case of the clinician scheduling problem, we chose departmental regulations and workload balance as hard constraints, while clinician preference and requests serve as soft constraints. After the schedule is generated, clinicians may exchange certain weeks or days throughout the year, to fulfill any missed requests.

The following are the departmental and workload constraints placed on the clinician assignments. First, each clinician must work between a minimum and maximum number of blocks of each service during the year. For instance, one clinician might have to provide 3-5 blocks of general ID service and 2-3 blocks of HIV service throughout the year. These limits may be different for each clinician, and they may change from year to year as the number of clinicians in the department changes. Second, the schedule should not assign a clinician to work for two consecutive blocks or two consecutive weekends. The schedule should also distribute regular weekends and holiday weekends each equally among all clinicians.

In addition to balancing the workload among clinicians, the schedule should accommodate their preferences. Clinicians provide their requests for time off before schedule generation so that the requests may be integrated into the schedule. Clinicians may specify days, weeks or weekends off, with the understanding that any blocks overlapping with their request will be assigned to a different clinician where possible. For example, if a clinician only requests a given Monday and Tuesday off, the schedule should avoid assigning the entire block to that clinician. Clinicians also typically prefer to have their weekend and block assignments side by side, so the schedule should accommodate this where possible. A summary of the outlined constraints is given in Table 1.

3 Methods

In this section, we present an application of integer linear programming to solve the clinician scheduling problem presented in Section 2. An integer linear program (ILP) consists of a linear objective function and linear constraints, with integer-valued variables. The optimal solution to the ILP must lie within the space defined by the constraints while also maximizing the objective value. First, we describe the sets, indices and variables necessary for the formulation of the problem. We then write the constraints given in Table 1 as linear functions of the variables, and define the objective function of the ILP.

Table 1: Summary of the constraints for the clinician scheduling problem

Constraint Name	Abbreviation	Description	Type
Block Coverage	BC	each service needs to have exactly one clinician that covers any given block	Hard
Weekend Coverage	WC	every weekend needs to have exactly one clinician that covers it	Hard
Min/Max	MM	for a given service, each clinician can only work between the minimum and maximum number of allowed blocks	Hard
No Consecutive Blocks	NCB	any clinician should not work two consecutive blocks, across all services	Hard
No Consecutive Weekends	NCW	any clinician should not work two consecutive weekends	Hard
Equal Weekends	EW	weekends should be equally distributed between clinicians	Hard
Equal Holidays	EH	long weekends should be equally distributed between clinicians	Hard
Block Requests	BR	each clinician can request to be off service during certain blocks throughout the year	Soft
Weekend Requests	WR	each clinician can request to be off service during certain weekends throughout the year	Soft
Block-Weekend Adjacency	BWA	the block and weekend assignments of a given clinician should be adjacent	Soft

Hard constraints must be satisfied by any candidate schedule. Soft constraints are optionally satisfied, and are used to rank the set of candidate solutions.

3.1 Sets and Indices

We denote the set of all services that clinicians in a single department can provide as $\mathcal{S}$. The set of all clinicians in the department is denoted as $\mathcal{C}$. The sets of blocks and weekends are denoted as $\mathcal{B}$ and $\mathcal{W}$ respectively. The block size used in our experiments is 2 weeks, but the following LP formulation does not require a particular size for blocks, and so it can be adapted for other cases. A subset of weekends are denoted $\mathcal{L} \subset \mathcal{W}$, corresponding to statutory long/holiday weekends such as the Thanksgiving weekend in the United States and Canada. Lastly, the block and weekend time-off requests of clinicians are denoted as the following subsets of all blocks and weekends: $\mathcal{U}_c \subset \mathcal{B}$ and $\mathcal{V}_c \subset \mathcal{W}$, respectively. For instance, if clinician c's time-off requests intersect with blocks 1 and 2, and weekend 1, then $\mathcal{U}_c = \{1, 2\}$ and $\mathcal{V}_c = \{1\}$. Table 2 presents a summary of the sets and indices described.

3.2 Variables

Since each clinician may be assigned to work on any service, during any block of the year, we denote such an assignment as a binary variable $X_{c,b,s} \in \{0, 1\}$. A value of 1 indicates that clinician c is assigned to service s during block b, while a value of 0 indicates they are not assigned. Weekend assignments are similarly defined using a binary variable $Y_{c,w} \in \{0, 1\}$, but without a service index, as clinicians are expected to provide all services during the weekends. We then define $m_{c,s}$ and $M_{c,s}$ to represent the minimal and maximal number of blocks of service s that clinician c is required to work during the year. Table 3 presents a summary of the constants and variables in the problem.

Table 2: Description of sets and indices in the problem

Set	Index	Description
$\mathcal{S} = \{1, \ldots, S\}$	s	services
$\mathcal{C} = \{1, \ldots, C\}$	c	clinicians
$\mathcal{B} = \{1, \ldots, B\}$	b	blocks
$\mathcal{W} = \{1, \ldots, W\}$	w	weekends
$\mathcal{L} \subset \mathcal{W}$		long weekends
$\mathcal{U}_c \subset \mathcal{B}$		block requests of clinician c
$\mathcal{V}_c \subset \mathcal{W}$		weekend requests of clinician c

Table 3: Description of variables and constants in the problem

Name	Description
$X_{c,b,s} \in \{0, 1\}$	assignment of clinician c to service s on block b
$Y_{c,w} \in \{0, 1\}$	assignment of clinician c on weekend w
$m_{c,s}$	minimum number of blocks clinician c should cover on service s
$M_{c,s}$	maximum number of blocks clinician c should cover on service s

3.3 Constraints

We now formalize the hard constraints in Table 1 using the variables defined above. The BC (block coverage) and WC (weekend coverage) constraints, are given by Eqns. (1) and (2), respectively.

$$\sum_{c=1}^{C} X_{c,b,s} = 1 \qquad \forall b \in \mathcal{B}, s \in \mathcal{S} \tag{1}$$

$$\sum_{c=1}^{C} Y_{c,w} = 1 \qquad \forall w \in \mathcal{W} \tag{2}$$

The MM (min/max) constraint is given by:

$$m_{c,s} \leq \sum_{b=1}^{B} X_{c,b,s} \leq M_{c,s} \qquad \forall\ c \in \mathcal{C}, s \in \mathcal{S} \tag{3}$$

The NCB (no consecutive blocks) and NCW (no consecutive weekends) constraints are given by Eqns. (4) and (5), respectively.

$$X_{c,b,s} + X_{c,b+1,s} \leq 1 \qquad \forall c \in \mathcal{C}, b \leq B - 1, s \in \mathcal{S} \tag{4}$$

$$Y_{c,w} + Y_{c,w+1} \leq 1 \qquad \forall c \in \mathcal{C}, w \leq W - 1 \tag{5}$$

The EW (equal weekend) and EH (equal holidays) constraints are given by Eqns. (6) and (7), respectively.

$$\left\lfloor \frac{W}{C} \right\rfloor \leq \sum_{w=1}^{W} Y_{c,w} \leq \left\lceil \frac{W}{C} \right\rceil \qquad \forall c \in \mathcal{C} \tag{6}$$

$$\left\lfloor \frac{|\mathcal{L}|}{C} \right\rfloor \leq \sum_{w \in \mathcal{L}} Y_{c,w} \leq \left\lceil \frac{|\mathcal{L}|}{C} \right\rceil \qquad \forall c \in \mathcal{C} \tag{7}$$

3.4 Objectives

As described in Section 2, the soft constraints of the clinician scheduling problem include: satisfying clinician block off requests (BR), satisfying clinician weekend off requests (WR), and assigning weekends closer to blocks (BWA). We convert these soft constraints into linear objective functions of the binary variables defined in Section 3.2. Objectives BR and WR are given in Eqns. (8) and (9) as linear functions of X and Y:

$$Q_1(X) = \sum_{c=1}^{C} \sum_{b=1}^{B} \sum_{s=1}^{S} (-1)^{\mathbf{1}(b \in \mathcal{U}_c)} \cdot X_{c,b,s} \tag{8}$$

$$Q_2(Y) = \sum_{c=1}^{C} \sum_{w=1}^{W} (-1)^{\mathbf{1}(w \in \mathcal{V}_c)} \cdot Y_{c,w} \tag{9}$$

where $\mathbf{1}(P)$ is the indicator function that has value 1 when predicate P holds and 0 otherwise. In the above two objectives, we penalize any assignments that conflict with a block or weekend request, and aim to maximize the non-conflicting assignments.

The BWA objective is optimized by considering the product $X_{c,b,s} \cdot Y_{c,w}$ for values of w "adjacent" to the value of b. This leads to the maximization objective:

$$Q_3(X, Y) = \sum_{c=1}^{C} \sum_{b=1}^{B} \sum_{s=1}^{S} X_{c,b,s} \cdot Y_{c,w=\varphi(b)} \tag{10}$$

where $\varphi(b)$ is a one-to-one mapping of a block to an adjacent weekend, by some appropriate definition of adjacency. For instance, clinicians might want to be assigned during a weekend that falls within an assigned block. In this case, we will have $\varphi(b) = 2b - 1$.

However, as it is, Q_3 is not a linear function of the assignment variables X and Y, and cannot be optimized in a linear programming framework. An approach used to convert such functions into linear objectives involves introducing a helper variable and additional constraints [23]. In our case, we introduce a variable $Z_{c,b,s}$ for every product $X_{c,b,s} \cdot Y_{c,w}$ with $w = \varphi(b)$, and constraining Z such that

$$Z_{c,b,s} \leq X_{c,b,s} \tag{11}$$

$$Z_{c,b,s} \leq Y_{c,w=\varphi(b)} \qquad \forall s \in \mathcal{S} \tag{12}$$

Since $X_{c,b,s}$ and $Y_{c,w}$ are binary variables, $Z_{c,b,s}$ will be constrained to 0, unless both $X_{c,b,s}$ and $Y_{c,w}$ are 1. Therefore, it suffices to maximize the following linear function of Z,

$$Q_3(Z) = \sum_{c=1}^{C} \sum_{b=1}^{B} \sum_{s=1}^{S} Z_{c,b,s} \tag{13}$$

to get the correct adjacency maximization objective.

In order to optimize all objectives simultaneously, we optimize a weighted sum of the normalized objective functions,

$$\max_{X,Y,Z} \alpha_1 \bar{Q}_1(X) + \alpha_2 \bar{Q}_2(Y) + \alpha_3 \bar{Q}_3(Z) \tag{14}$$

subject to the constraints defined in Section 3.3, where $\bar{Q}_i$ is the normalization of objective Q_i, and $0 \leq \alpha_i \leq 1$. This method guarantees an optimal solution to be Pareto optimal [24].

Currently, the most efficient approach to finding an exact solution for an ILP is called Branch-and-Cut [25]. This method involves iteratively solving LP relaxations of the ILP, then constraining the

relaxed problems and considering various sub-problems until it finds integral solutions to the original ILP. In the intermediate relaxations, the integer assignment variables can take on real values, allowing the problem to be solved efficiently using the Simplex method [26]. The complexity of finding an optimal integral solution thus lies in the branching search structure of Branch-and-Cut.

4 Experiments

Our goal was to determine if the schedules provided by solving the ILP could successfully (i) enforce all hard constraints; (ii) improve fulfillment of soft constraints compared to the manual approach; and (iii) assess whether our ILP formulation can be used for a wide range of configurations. First, we compared the schedules created by solving the ILP formulation given in Section 3 to schedules that were manually generated, with respect to adherence to the hard and soft constraints outlined in Section 2. We then examined the efficiency of the ILP approach in generating schedules by its runtime on a variety of instances that may be found in the real-world.

4.1 Implementation

We developed a Python software package with a user interface that implements the above linear program and allows configuration of clinicians, to be used by the ID division at St. Michael's Hospital [27]. The software was used to generate the results in the following sections, using real data as well as simulated data as input. All the following experiments were conducted on an Intel Core i7-4770k CPU @ 3.50 GHz with 16 GB of RAM running 64-bit Windows 10. Our software package uses COIN-OR Branch-and-Cut open source solver version 2.9.9 [28].

4.2 Comparison with Manually Generated Schedules

We used clinician time-off requests and minimum/maximum requirements from 2015-2018 as input data for the ILP problem. Table 4 compares the optimal schedule generated using the software with the manually-created schedule for data from 2018. The schedule is color-coded to distinguish between the different clinicians.

First, we evaluated the ILP solution by comparing it with the manual generation, as in Table 4. Specifically, we examined the adherence of each schedule to the constraints presented in Table 1. As shown in Table 5, the ILP solution satisfied all hard constraints. In contrast, manual generation did not satisfy all hard constraints. In particular, we see that the manual generation assigned clinicians to multiple consecutive blocks in all four years. Moreover, the manual generation did not have an equal distribution of weekends and holidays for all four years of data. Considering all objectives, we see that the ILP solution outperforms manual generation in all four years, by accommodating almost all time-off requests and ensuring that weekends are always assigned close to blocks.

4.3 Influence of Problem Complexity on Runtime

Next, we examined the influence of the following four parameters on the runtime of the ILP solver using simulated data: number of clinicians; number of services offered; number of time-off requests per clinician per year; time-horizon of the schedule.

Table 4: Comparison of automatically generated (ILP solution) and manually generated schedules for 2018. In the ILP solution there are rigid 2-week block assignments, unlike manual generation that often assigns 3-4 weeks in a row to a single clinician. Moreover, in the ILP solution we see for each block either the HIV or ID clinician was assigned weekend coverage, indicating improved Block-Weekend Adjacency.

Week #	ILP Solution			Manual Generation		
	HIV	ID	Weekend	HIV	ID	Weekend
1	A	E	E	A	E	H
2	A	E	H	A	E	A
3	B	F	F	B	H	G
4	B	F	H	B	H	I
5	A	G	A	A	G	F
6	A	G	E	A	G	C
7	B	C	C	A	F	B
8	B	C	G	D	C	G
9	D	I	D	B	C	D
10	D	I	H	B	D	H
11	A	B	B	A	B	F
12	A	B	I	A	B	A
13	C	F	F	C	H	H
14	C	F	I	C	H	I
15	A	H	H	B	I	C
16	A	H	D	B	I	E
17	B	E	B	A	E	D
18	B	E	H	A	E	E
19	A	I	I	A	C	F
20	A	I	D	A	C	C
21	C	D	C	B	G	A
22	C	D	I	B	G	C
23	B	F	F	C	F	D
24	B	F	G	C	F	C
25	A	H	A	C	C	G
26	A	H	H	D	I	D
27	C	D	D	A	B	E
28	C	D	E	A	B	I
29	A	B	A	B	D	D
30	A	B	B	B	D	A
31	C	E	C	C	F	E
32	C	E	A	C	F	F
33	B	D	D	B	F	I
34	B	D	E	B	I	C
35	A	I	A	A	G	G
36	A	I	G	A	G	I
37	D	C	C	D	C	A
38	D	C	F	D	D	E
39	B	E	E	A	B	D
40	B	E	B	B	I	I
41	A	G	G	B	I	G
42	A	G	E	D	F	F
43	C	I	C	C	F	C
44	C	I	I	D	E	I
45	A	F	A	D	E	E
46	A	F	F	A	B	A
47	B	G	B	A	B	D
48	B	G	I	A	D	G
49	D	C	D	A	D	F
50	D	C	B	B	G	G
51	B	G	G	B	G	E

The "HIV/ID" columns represent the assignments for the two concurrent services offered at the department. The "Weekend" column represents the assignments for weekend coverage in both services. Different colours and letters are used to distinguish different clinicians.

Table 5: Comparison of constraint satisfaction and objectives values in LP-generated and historical schedules.

	2015		**2016**		**2017**		**2018**	
	LP	Historical	LP	Historical	LP	Historical	LP	Historical
Constraint								
Block Coverage	✓	✓	✓	✓	✓	✓	✓	✓
Weekend Coverage	✓	✓	✓	✓	✓	✓	✓	✓
Min/Max	✓	✓	✓	✓	✓	✓	✓	✓
No Consecutive Blocks	✓		✓		✓		✓	
No Consecutive Weekends	✓		✓		✓	✓	✓	✓
Equal Weekends	✓		✓		✓		✓	
Equal Holidays	✓		✓	✓	✓		✓	✓
Objective								
Satisfied Block Requests	123/129	121/129	120/126	116/126	99/99	95/99	124/128	121/128
Satisfied Weekend Requests	113/113	111/113	119/119	112/119	75/75	75/75	115/115	113/115
Adjacent Block-Weekend Assignments	26/26	9/26	26/26	6/26	26/26	7/26	26/26	5/26

The objective denominator represents: the total number of requests submitted by clinicians for request objectives; and the total number of possible adjacencies for adjacency objectives.

The effect of increasing the number of clinicians and number of services on the runtime of the program is shown in Table 6. We executed the algorithm for $S = \{1, 2, 3\}$ total services and $C = \{10, 20, 30, 50\}$ clinicians in total across all services. In a department providing a single service, increasing the number of clinicians did not affect the runtime, and we were able to find an ILP solution in all four cases within 1 second. For 2 concurrent services, a roster of 30 or more clinicians becomes impractical to schedule, as searching for a solution required over 24 hours. We saw similar issues for a roster of 20 or more clinicians assigned to a division with 3 concurrent services. However, when removing the NCB constraint, we saw a great improvement in runtime for divisions with 2 and 3 services, and we were able to generate a schedule with upwards of 50 clinicians in under 1.5 seconds.

Table 6: Comparison of program runtime (in seconds) for different numbers of services and total clinicians in the division.

	Number of Services				
Number of Clinicians	1	2	2 (*)	3	3 (*)
10	0.16	0.74	0.16	1.40	0.23
20	0.25	7468.86	0.32	–	0.43
30	0.42	–	0.49	–	0.66
50	0.62	–	0.82	–	1.14

Notes: "–" indicates that no solution was found within 24 hours; "(*)" indicates that the No Consecutive Blocks (NCB) constraint was removed.

For the remaining experiments, we simulated a department with 10 clinicians offering two services, similar to the department at St. Michael's Hospital. The effect of an increasing number of requests per clinician on the runtime of the ILP solver is shown in Figure 1. In this experiment, each clinician was configured with 1 to 15 total block requests. The runtime of the algorithm is constant with respect to the number of requests, indicating that it can accommodate a lot of flexibility in clinician requests. Moreover, we see that all runs were completed in under 2 seconds.

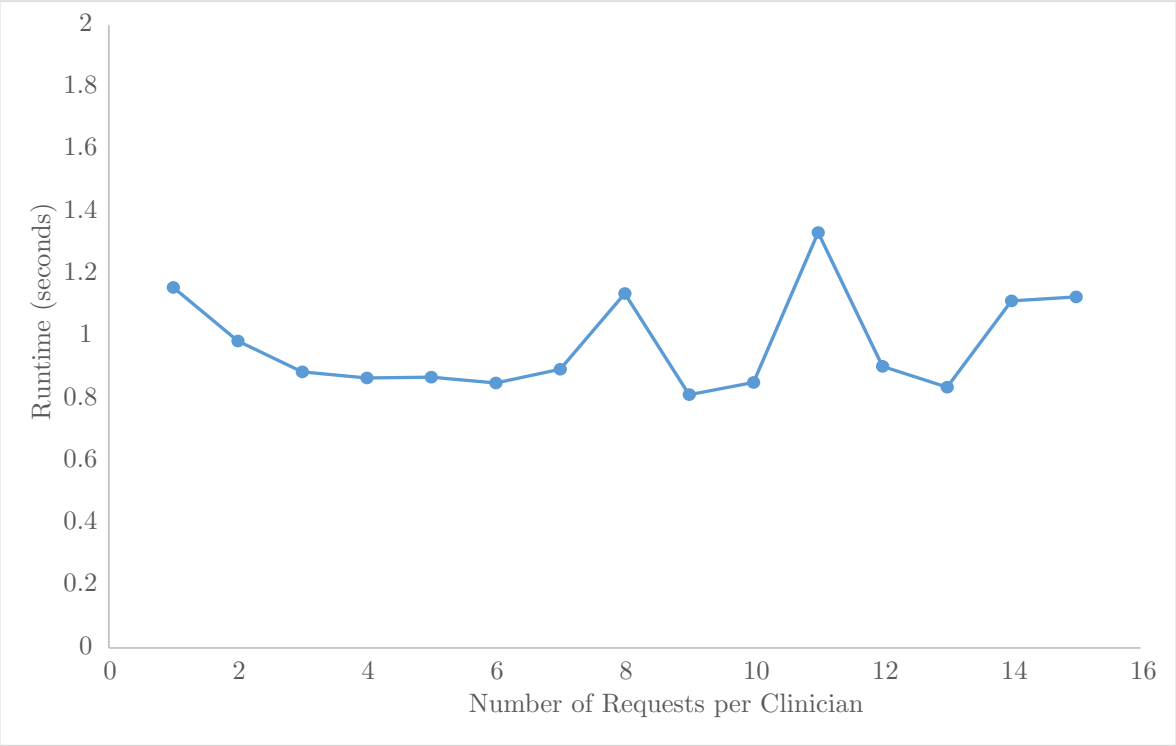
Figure 2 presents the change in runtime when increasing the number of 2-week blocks in a department with 10 clinicians offering two services. In this experiment, we investigated time horizons from 5 to 110 blocks. This is equivalent to generating a schedule for up to 4 years ahead. The trend in the graph indicates a linear growth in runtime with respect to the time-horizon. Notably, the ILP solver was able to find all solutions in under 6 seconds, indicating very good performance for long-term scheduling.

5 Discussion

In this paper, we present a simple, yet flexible, integer linear programming formulation to generate schedules for clinical departments at hospitals. The challenge in applying ILP to the task of scheduling clinicians lies in the computational complexity of finding an optimal solution. As the size of the scheduling problem grows, due to a larger roster of clinicians or more complicated constraints, the time it takes to generate an optimal schedule may grow exponentially in the worst case. Many previous approaches to creating schedules in similar scenarios have avoided this problem by using heuristics to find an approximately optimal solution in a shorter amount of time [21].

We presented a formulation that includes both hard constraints to ensure the schedule satisfies hospital and logistics requirements, and a multi-goal objective function to satisfy soft constraints

Figure 1: Runtime of ILP solver with an increasing number of requests per clinician

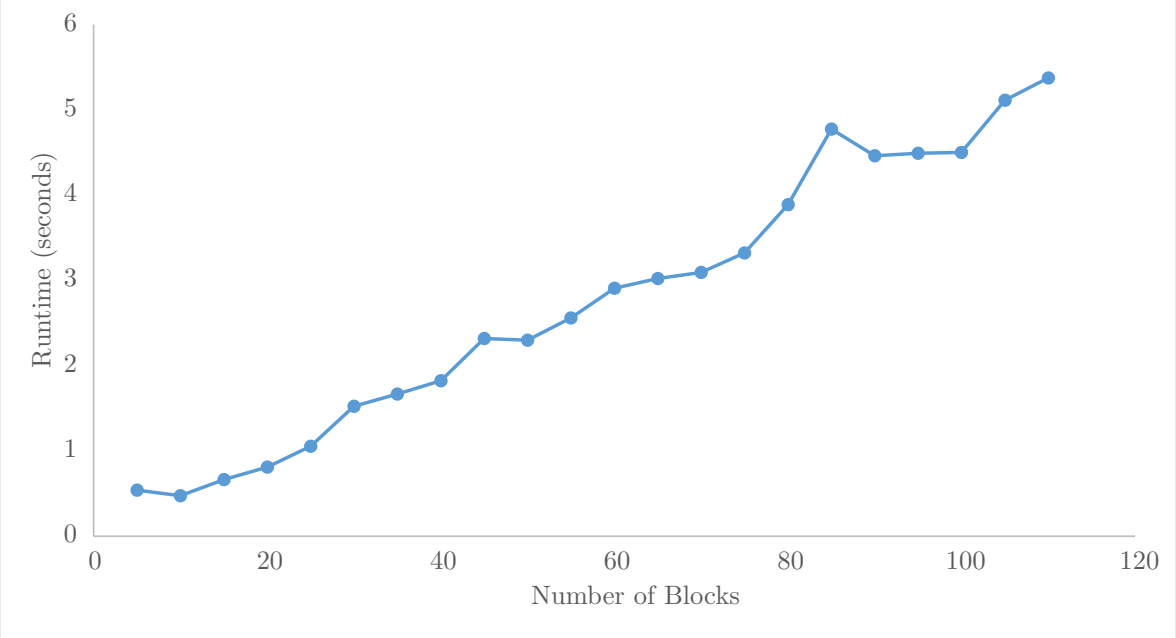


(work preferences of clinicians). Although we restricted our application of the formulation to a set of constraints for the particular needs of the case study (St. Michael's Hospital Division of Infectious Diseases), our formulation can be adapted to various clinical departments at different hospitals. The flexibility of our ILP allows changing the number of services provided in a division, the length of a work block, clinicians' preference for block to weekend adjacency as well as clinicians' requests for time off.

When comparing the optimal schedule generated by our tool to the manually-created schedules at St. Michael's Hospital, we found that the ILP formulation was always able to find an optimal schedule satisfying all required hard constraints, unlike the manual schedule, which often did not satisfy all constraints. Moreover, due to the multi-goal objective function in the ILP, the algorithm was able to fulfill the majority of clinician preferences and requests, more so than the manually-created schedule. These observations reinforce the benefits of automated tools when generating schedules in hospital departments to balance the workload of clinicians and improve the service provided to patients. The use of automated tools alleviates the time spent on designing the schedule by hand, and provides clinical departments with a more fair distribution of work that helps improve the overall satisfaction of both employees and patients [29].

In our simulations, we also found that increasing the number of requests per clinician did not affect the runtime of the algorithm, highlighting the flexibility of the tool to incorporate clinician preferences. Further, we saw that the algorithm can accommodate an increase in time-horizon up to four years with little impact on runtime, suggesting the algorithm can be used generate schedules far in advance. A key limitation we identified was the sensitivity of the runtime to larger numbers of services offered by a single department. One solution to mitigate the runtime issues created by a larger number of

Figure 2: Runtime of ILP solver with an increasing number of 2-week blocks



services would be removing the constraint that prevents assignment of consecutive blocks, followed by manual readjustment from the generated schedule. Overall, our sensitivity analyses using simulated data provided reassurance that the ILP formulation can be applied to schedule clinicians across real-world variability between clinical departments. Next steps include expanding the generalizability of the tool beyond smaller clinical departments to larger departments within and outside of health-care – especially those that provide multiple services in parallel for patients and other clients. As well, additional work can be done to incorporate other clinician preferences, such as the ability to request time-on slots or preference for time of year.

6 Acknowledgements

The authors are grateful to Julie Veitch (St. Michael's Hospital) for her contributions to testing and designing the scheduling software. SM is supported by a Canadian Institutes of Health Research New Investigator Award. JK is supported by a Natural Sciences and Engineering Research Council doctoral award. DL conducted part of this project as a Keenan Research Summer Student, Li Ka Shing Knowledge Institute, St. Michael's Hospital, University of Toronto.

7 Funding

The work was jointly funded by the Ontario Ministry of Science and Innovation Early Researcher Award Number ER17-13-043; the Division of Infectious Diseases, St. Michael's Hospital, University of Toronto; and the University of Toronto Work Study Program.

References

[1] Uwe Aickelin, Edmund K. Burke, and Jingpeng Li. Improved Squeaky Wheel Optimisation for Driver Scheduling. In Thomas Philip Runarsson, Hans-Georg Beyer, Edmund Burke, Juan J. Merelo-Guervós, L. Darrell Whitley, and Xin Yao, editors, *Parallel Problem Solving from Nature - PPSN IX*, Lecture Notes in Computer Science, pages 182–191. Springer Berlin Heidelberg, 2006.

[2] Asvin Goel, Claudia Archetti, and Martin Savelsbergh. Truck driver scheduling in Australia. *Computers & Operations Research*, 39(5):1122–1132, May 2012.

[3] Maik Günther and Volker Nissen. Combined Working Time Model Generation and Personnel Scheduling. In Wilhelm Dangelmaier, Alexander Blecken, Robin Delius, and Stefan Klöpfer, editors, *Advanced Manufacturing and Sustainable Logistics*, Lecture Notes in Business Information Processing, pages 210–221. Springer Berlin Heidelberg, 2010.

[4] Salem M. Al-Yakoob and Hanif D. Sherali. Mixed-integer programming models for an employee scheduling problem with multiple shifts and work locations. *Annals of Operations Research*, 155(1):119–142, November 2007.

[5] S. M. Al-Yakoob and H. D. Sherali. A column generation approach for an employee scheduling problem with multiple shifts and work locations. *Journal of the Operational Research Society*, 59(1):34–43, January 2008.

[6] H. K. Alfares. A Simulation Approach for Stochastic Employee Days-Off Scheduling. *International Journal of Modelling and Simulation*, 27(1):9–15, January 2007.

[7] Nicolas Chapados, Marc Joliveau, and Louis-Martin Rousseau. Retail Store Workforce Scheduling by Expected Operating Income Maximization. In Tobias Achterberg and J. Christopher Beck, editors, *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, Lecture Notes in Computer Science, pages 53–58. Springer Berlin Heidelberg, 2011.

[8] Volker Nissen and Maik Günther. Automatic Generation of Optimised Working Time Models in Personnel Planning. In Marco Dorigo, Mauro Birattari, Gianni A. Di Caro, René Doursat, Andries P. Engelbrecht, Dario Floreano, Luca Maria Gambardella, Roderich Groß, Erol Şahin, Hiroki Sayama, and Thomas Stützle, editors, *Swarm Intelligence*, Lecture Notes in Computer Science, pages 384–391. Springer Berlin Heidelberg, 2010.

[9] M. E. T. Horn, H. Jiang, and P. Kilby. Scheduling patrol boats and crews for the Royal Australian Navy. *Journal of the Operational Research Society*, 58(10):1284–1293, October 2007.

[10] Manuel Laguna and Terry Wubbena. Modeling and Solving a Selection and Assignment Problem. In Bruce Golden, S. Raghavan, and Edward Wasil, editors, *The Next Wave in Computing, Optimization, and Decision Technologies*, Operations Research/Computer Science Interfaces Series, pages 149–162. Springer US, 2005.

[11] M.N. Azaiez and S.S. Al Sharif. A 0-1 goal programming model for nurse scheduling. *Computers & Operations Research*, 32(3):491–507, March 2005.

[12] Lorraine Trilling, Alain Guinet, and Dominiue Le Magny. Nurse scheduling using integer linear programming and constraint programming. *IFAC Proceedings Volumes*, 39(3):671–676, 2006.

[13] Maya Widyastiti, Amril Aman, and Toni Bakhtiar. Nurses Scheduling by Considering the Qualification using Integer Linear Programming. *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, 14(3):933, September 2016.

[14] Ahmed Ali El Adoly, Mohamed Gheith, and M. Nashat Fors. A new formulation and solution for the nurse scheduling problem: A case study in Egypt. *Alexandria Engineering Journal*, 57(4):2289–2298, December 2018.

[15] Uwe Aickelin and Kathryn A. Dowsland. Exploiting problem structure in a genetic algorithm approach to a nurse rostering problem. *Journal of Scheduling*, 3(3):139–153, 2000.

[16] A. Jan, M. Yamamoto, and A. Ohuchi. Evolutionary algorithms for nurse scheduling problem. In *Proceedings of the 2000 Congress on Evolutionary Computation. CEC00 (Cat. No.00TH8512)*, volume 1, pages 196–203 vol.1, July 2000.

[17] H. Kawanaka, K. Yamamoto, T. Yoshikawa, T. Shinogi, and S. Tsuruoka. Genetic algorithm with the constraints for nurse scheduling problem. In *Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No.01TH8546)*, volume 2, pages 1123–1130 vol. 2, May 2001.

[18] Andrzej Jaszkiewicz. A metaheuristic approach to multiple objective nurse scheduling. *Foundations of Computing and Decision Sciences*, 22(3):169–184, 1997.

[19] Slim Abdennadher and Hans Schlenker. Nurse scheduling using constraint logic programming. In *AAAI/IAAI*, pages 838–843, 1999.

[20] Haibing Li, Andrew Lim, and Brian Rodrigues. A Hybrid AI Approach for Nurse Rostering Problem. In *Proceedings of the 2003 ACM Symposium on Applied Computing*, SAC '03, pages 730–735, New York, NY, USA, 2003. ACM.

[21] Edmund K. Burke, Patrick De Causmaecker, Greet Vanden Berghe, and Hendrik Van Landeghem. The State of the Art of Nurse Rostering. *Journal of Scheduling*, 7(6):441–499, November 2004.

[22] Tim B. Cooper and Jeffrey H. Kingston. The complexity of timetable construction problems. In Gerhard Goos, Juris Hartmanis, Jan Leeuwen, Edmund Burke, and Peter Ross, editors, *Practice and Theory of Automated Timetabling*, volume 1153, pages 281–295. Springer Berlin Heidelberg, Berlin, Heidelberg, 1996.

[23] P. L. Hammer and S. Rudeanu. *Boolean Methods in Operations Research and Related Areas*. Ökonometrie und Unternehmensforschung Econometrics and Operations Research. Springer-Verlag, Berlin Heidelberg, 1968.

[24] Ivan P. Stanimirovic, Milan Lj Zlatanovic, and Marko D. Petkovic. On the linear weighted sum method for multi-objective optimization. *Facta Acta Univ*, 26(4):49–63, 2011.

[25] John E. Mitchell. Branch-and-cut algorithms for combinatorial optimization problems. *Handbook of applied optimization*, 1:65–77, 2002.

[26] Ron Shamir. The Efficiency of the Simplex Method: A Survey. *Management Science*, 33(3):301–334, 1987.

[27] David Landsman. On-call scheduling tool for clinicians. `https://github.com/c-uhs/scheduler`, 2019.

[28] johnjforrest, Stefan Vigerske, Ted Ralphs, Haroldo G. Santos, Lou Hafer, Bjarni Kristjansson, jpfasano, Edwin Straver, Miles Lubin, rlougee, jpgoncal1, h-i-gassmann, and Matthew Saltzman. coin-or/Cbc: Version 2.9.9, 2019.

[29] Rhian Silvestro and Claudio Silvestro. An evaluation of nurse rostering practices in the National Health Service. *Journal of Advanced Nursing*, 32(3):525–535, 2000.