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

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

[a] *MAP Centre for Urban Health Solutions, Unity Health Toronto, Toronto, ON, Canada*
[b] *Department of Computer Science, University of Toronto, Toronto, ON, Canada*
[c] *Department of Medicine, Division of Infectious Disease, St. Michael's Hospital, University of Toronto, Toronto, ON, Canada*
[d] *Institute of Health Policy, Management and Evaluation, Dalla Lana School of Public Health, University of Toronto, Toronto, ON, Canada*
[e] *Institute of Medical Sciences, University of Toronto, Toronto, ON, Canada*

September 7, 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 computational complexity issues by maintaining a minimal set of constraints, yet still provides the flexibility necessary to adapt it 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 generazability of our approach in different real-world scenarios.

1 Introduction

Hospital departments provide services where patient needs, and thus the system's demands, often exceed the available supply. In particular, on-call schedules for a fixed number of health-care providers are central to the efficient running of hospitals. Carefully allocated schedules are meant to simultaneously ensure sufficient resources are provided to patients while not overworking clinicians to prevent costly mistakes. It is common practice for on-call schedules in hospitals to be created manually, yet manually-created schedules are prone to errors and potential for various biases. First, when there is a large number of clinicians in a single department, or the constraints that need to be satisfied by the department are very complex, a manual method may not provide a schedule that satisfies all clinicians. Second, such methods are likely to overlook certain constraints that must be maintained to have an operational department, such as preventing many consecutive work blocks from being assigned or ensuring clinicians are allocated a specific amount of work blocks throughout the year. Third, manual scheduling is often time-consuming for the person developing the schedule. For these reasons, it is important to develop automated methods that can generate optimal schedules that satisfy the given constraints of the hospital department.

Automated methods to optimize 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 on-call scheduling problem are the approaches to schedule 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 of the hard constraints, such as hospital regulations, and as many soft constraints as possible, which may include nurse preferences. 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 rostering is presented in [21].

Many of the approaches to nurse scheduling were designed to satisfy the requirements of a specific hospital department which causes 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 the department and the preferences of the nurses and clinicians in that department, they are difficult to readily adapt to other departments in the same hospital or other hospitals. Moreover, the large number of variables and constraints also leads to computational complexity issues [22], especially when using exact methods for finding the solution. In this paper, we tackle a version of the nurse scheduling problem arising from a case study of one clinical division, providing two different services simultaneously (general

infectious disease (ID) consults; and HIV consults service) at St. Michael's Hospital in Toronto, Canada. Our goal is to (1) present a simple integer linear programming (ILP) formulation for the scheduling problem as developed for the hospital, and describe the adaptability of the formulation to solving similar problems in other departments; (2) compare the performance of the ILP scheduler to the results of the manual approach; and (3) analyze the robustness of the algorithm in difficult instances of the problem.

We begin by describing 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 (ID) offers general ID and HIV consultation on inpatients as two parallel services. Each service provides clinical care throughout the year, during regular work weeks as well as weekends and holidays. The clinicians in the division typically receive a schedule in advance, outlining their on-call service dates for the full year. In the yearly schedule each clinician is assigned to blocks of regular work weeks and weekends. Each block corresponds to two consecutive work weeks. 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.

Several constraints are placed on the on-call assignments. First, each clinician has limits on the number of blocks they can and must work during the year, depending on the type of consultation. For instance, a clinician might have to provide 3-5 blocks of general ID consultation as well as 2-3 blocks of HIV consultation throughout the year. These limits may change from year to year as the number of clinicians in the department changes. Moreover, the schedule should not assign a clinician to work for two consecutive blocks or two consecutive weekends in a row. The schedule should also distribute both regular and holiday weekends equally among all clinicians.

In addition to maintaining a balanced work load among clinicians, the schedule should also accommodate their preferences. Clinicians provide their requests for time off ahead of schedule generation so that the requests may be integrated into the schedule. Individuals 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 will generally avoid assigning the entire block to that clinician. Clinicians also prefer to have their weekend and block assignments close together, so the schedule should account for this when distributing assignments. A summary of the outlined constraints is given in Table 1.

Constraint Name	**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

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

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 all candidate solutions. Typically, soft constraints are encoded as objective functions rather than actual constraints. These objective functions are maximized or minimized when finding a solution. In the case of our clinician scheduling problem, we chose Block Requests, Weekend Requests and Block-Weekend Adjacency as our soft constraints, while the rest of the constraints are hard. Though it is important to take clinician requests into account when constructing the schedule, it is crucial that the work-load of the schedule is balanced among all clinicians, and the needs of the patients are fulfilled. It is important to note that - irrespective of how the schedule is generated - clinicians may exchange certain weeks or days throughout the year after the schedule is implemented. The approach to solving the clinician scheduling problem does not account for these future exchanges, and only focuses on the full time horizon.

3 Methods

In this section, we present an application of ILP to solve the clinician scheduling problem presented in Section 2. First, we describe the sets, indices and variables present in the formulation of the program. We then convert the constraints given in Table 1 into mathematical terms, and outline the objective function of the linear program.

3.1 Sets and Indices

We denote the set of all services that clinicians in a single department can provide as $\mathcal{S}$. The following formulation assumes that all clinicians are able to provide all services. The set of all clinicians in the department is denoted as $\mathcal{C}$. The sets of blocks and weekends that clinicians will be assigned to 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 specify a particular size for blocks, and so it can be adapted to the needs of the department in question. A subset of weekends are denoted $\mathcal{L}$, corresponding to established long/holiday weekends such as the Canadian Civic Day weekend in August. Lastly, we denote with $\mathcal{BR}_c$ and $\mathcal{WR}_c$ the block and weekend requests of clinicians as subsets of all blocks and weekends, respectively. For instance, if clinician c's requests intersect with blocks 1 and 2, and weekend 1, then $\mathcal{BR}_c = \{1, 2\}$ and $\mathcal{WR}_c = \{1\}$. Table 2 presents a summary of the sets and indices described.

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
$\mathcal{BR}_c \subset \mathcal{B}$		block requests of clinician c
$\mathcal{WR}_c \subset \mathcal{W}$		weekend requests of clinician c

Table 2: Description of sets and indices in the problem

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 $X_{c,b,s}$. A value of 1 indicates that the given clinician c is assigned to provide service s during block b. Weekend assignments are similarly defined using a binary variable $Y_{c,w}$, but without a service index, as clinicians are expected to provide all services during the weekends. We then introduce a set of constants $m_{c,s}$ and $M_{c,s}$ to constrain the minimal and maximal number of blocks each clinician is allowed to work during the year. Table 3 presents a summary of the constants and variables 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

Table 3: Description of variables and constants in the problem

3.3 Constraints

We now formalize the hard constraints in Table 1 using the variables defined above.

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

$$\sum_{c=1}^{C} Y_{c,w} = 1 \qquad \forall w \in \mathcal{W} \qquad \text{(WC)}$$

$$m_{c,s} \leq \sum_{b=1}^{B} X_{c,b,s} \leq M_{c,s} \qquad \forall\ c \in \mathcal{C}, s \in \mathcal{S} \qquad \text{(MM)}$$

$$X_{c,b,s} + X_{c,b+1,s} \leq 1 \qquad \forall c \in \mathcal{C}, b \leq B-1, s \in \mathcal{S} \qquad \text{(NCB)}$$

$$Y_{c,w} + Y_{c,w+1} \leq 1 \qquad \forall c \in \mathcal{C}, w \leq W-1 \qquad \text{(NCW)}$$

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

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

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) can be written as the following linear functions:

$$Q_1(X) = \sum_{c=1}^{C} \sum_{b=1}^{B} \sum_{s=1}^{S} (-1)^{\mathbb{1}(b \in \mathcal{BR}_c)} \cdot X_{c,b,s} \qquad \text{(BR)}$$

$$Q_2(Y) = \sum_{c=1}^{C} \sum_{w=1}^{W} (-1)^{\mathbb{1}(w \in \mathcal{WR}_c)} \cdot Y_{c,w} \qquad \text{(WR)}$$

where $\mathbb{1}(P)$ 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 Block-Weekend Adjacency 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

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

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 its variables 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, introducing 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

$$Z_{c,b,s} \leq X_{c,b,s} \tag{1}$$
$$Z_{c,b,s} \leq Y_{c,w=\varphi(b)} \qquad \forall s \in \mathcal{S} \tag{2}$$

allows us to rewrite Q_3 as a linear function of Z,

$$Q_3(Z) = \sum_{c=1}^{C} \sum_{b=1}^{B} \sum_{s=1}^{S} Z_{c,b,s} \tag{3}$$

Indeed from Eqns. (1) and (2), whenever $X_{c,b,s} \cdot Y_{c,w} = 1$ (respectively, 0), $Z_{c,b,s}$ can attain a maximum value of 1 (respectively, 0), giving us the correct adjacency maximization objective.

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

$$\max_{X,Y,Z} \alpha \bar{Q}_1(X) + \beta \bar{Q}_2(Y) + (1 - \alpha - \beta)\bar{Q}_3(Z) \tag{4}$$

where $\bar{Q}_i$ is the normalization of objective Q_i, and $0 \leq \alpha, \beta \leq 1$. This method guarantees an optimal solution to be Pareto optimal [24].

4 Results

4.1 Software

We developed a software package with a user interface that implements the above linear program and allows configuration of clinicians at [ref **??**], to be used by the ID division at St. Michael's Hospital. 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 and 16 GB of RAM running 64-bit Windows 10. The software was implemented using the COIN-OR Branch-and-Cut open source solver version 2.9.9 [25].

4.2 Infectious Diseases Division

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

Week #	LP Solution			Historical Data		
	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

Table 4: Comparison of schedules for 2018

First, we evaluated the software-generated schedule by comparing it with the manually-created schedule. Specifically, we examined the adherence of each schedule to the constraints presented in Table 1. As shown in Table 5, the optimal schedule generated using the software satisfied all mandatory constraints. In contrast, the manually-created schedule was not able to satisfy all mandatory constraints. In particular, we see that the manual schedule assigned clinicians to multiple consecutive blocks in all four years. Moreover, the manual schedule did not have an equal distribution of weekends and holidays for all four years of data. Evaluating the objectives, we see that LP outperforms the manually created schedule in all four years, by accommodating almost all time-off requests and ensuring that weekends are always assigned close to blocks.

	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

Table 5: Comparison of constraint satisfaction and objectives in LP-generated and historical schedules. For requests, the denominator represents the total number of requests submitted by clinicians. For adjacency, the denominator represents the total number of possible adjacencies.

4.3 Simulations

We then examined the properties of the ILP problem using simulated data. Specifically, we examined the influence of expanding the following parameters on the runtime of the algorithm: number of clinicians; number of services offered in the division; amount of time-off requests per clinician throughout the year. We also examined the effect of a longer time-horizon on the runtime.

The effect of an increasing number of requests per clinician on the runtime of the ILP solver is shown in Figure 1. We executed the program on a department with 10 clinicians offering two services, similar to the department at St. Michael's Hospital. For simplicity, each clinician was configured with $|\mathcal{BR}| = \{0, 5, 10, 15\}$ non-overlapping block requests. The increase in the number of requests did not affect the runtime of the ILP solver, indicating that it can accommodate a lot of flexibility in clinician requests. Moreover, we see that all four runs were completed in under 1 second.

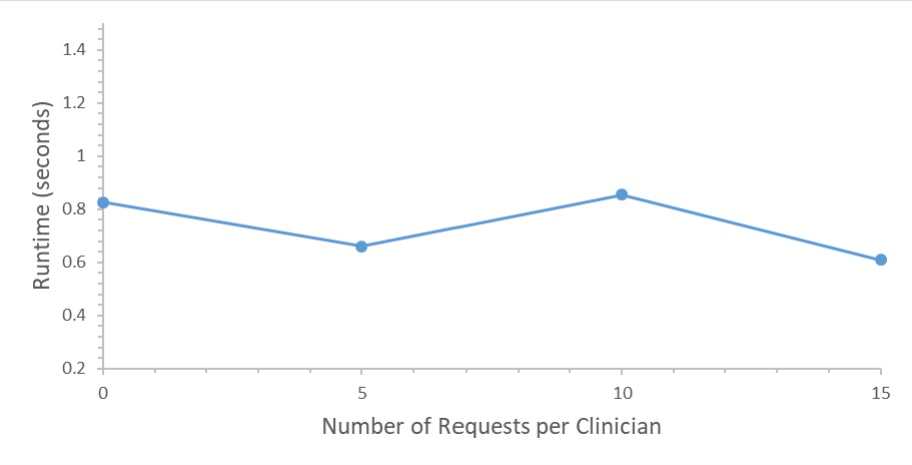

Figure 1: Runtime of ILP solver with an increasing number of requests per clinician

Figure 2 presents the change in runtime when increasing the number of 2-week blocks $B = \{26, 52, 78, 104\}$ in a department with 10 clinicians offering two services. There does not seem to be a strong effect on runtime when we lengthen the time-horizon of the schedule, and in fact the solver was able to find an optimal schedule for all time horizons within 3 seconds.

The effect of increasing the 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. For 2 concurrent services, a roster of 30 or more clinicians becomes impractical to schedule, as

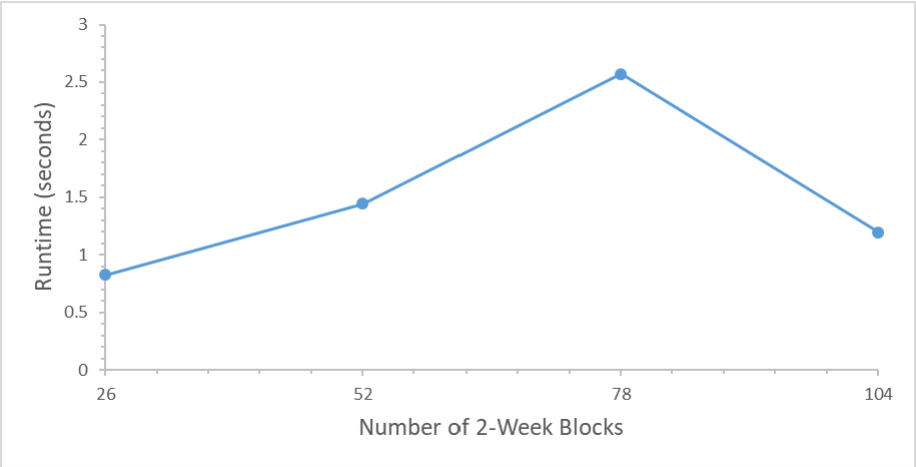

Figure 2: Runtime of ILP solver with an increasing number of 2-week blocks

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 relaxing 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.

	Number of Services				
Number of Clinicians	1	2	2 (without NCB)	3	3 (without NCB)
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

Table 6: Comparison of program runtime (in seconds) for different number of services and total clinicians in the division. '-' indicates that a solution was not found within 24 hours. 'without NCB' indicates that the No Consecutive Blocks (NCB) constraint was relaxed.

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. As a result, the majority of approaches taken to create schedules for similar scenarios tend to use heuristics in order to find an approximately optimal solution in a shorter 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 the work preferences of clinicians. Although we restricted our use of the formulation to a minimal 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 take all constraints into account. Moreover, due to the multi-goal objective function in the ILP, the tool was able to maximize and 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 work-load 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 [26].

In our simulations, we also found that increasing the number of requests per clinician does not affect the runtime of the algorithm, highlighting the flexibility of the tool in incorporating clinician preferences. Further, we saw that the algorithm can accommodate an increase in time-horizon up to four years with little impact on runtime, which can be applied to departments that generate schedules far in advance. One key limitation we identified was the sensitivity of the runtime to larger numbers of services offered in a single department. Such cases are unlikely to be encountered in the real-world because most clinical departments tend to provide a single service or at most two services by the same roster of clinicians [SM/Kevin, could you help with reference here?].

One solution to mitigate the runtime issues created by a larger number of services would be relaxing the constraints that prevent assignment of consecutive blocks, followed by manual readjustment from the generated schedule. Overall, our sensitivity analyses using simulated data provide 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.

6 Acknowledgements

The authors are grateful to Julie Veitch for her contributions to testing and designing the scheduling software. SM is supported by a Canadian Institutes of Health Research New Investigator 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. page 6.

[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] 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.

[26] 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.