Timetabling of Workplace Training: A Combination of Mathematical Programming and Simulated Annealing

Oliver G. Czibula · Hanyu Gu · Yakov Zinder

Abstract The paper is concerned with the problem of scheduling workplace training, which arises in a broad range of organisations, from the hospital placements of nursing students to apprentice training at organisations such as electricity distributors. The problem can be viewed as a generalisation of the open shop scheduling problem. The paper discusses the complexity of the considered problem, and presents an optimisation procedure, which is a sequential application of integer linear programming and simulated annealing. The effectiveness of the proposed optimisation procedure was demonstrated by computational experiments using data with typical characteristics of a real-world problems arising at large electricity distributors. The computational experiments show that the proposed optimisation procedure produces superior solutions on average compared to those from a general purpose MIP solver.

1 Introduction

The paper is concerned with the problem of scheduling workplace training, which arises in a broad range of organisations. This practical training is frequently in the form of a set of so-called "practice placements", sometimes also referred to as "rotations" [10]. Each practice placement is undertaken by one or more students simultaneously and is designed to provide them with real-world hands-on experience while under the supervision of a competent instructor.

This training model is encountered in a number of disciplines and in a number of industries. It is perhaps most notable in the area of clinical education, for example in nursing, where students that have completed the theoretical components of their studies must then gain practical experience in a number of areas prior to becoming fully qualified [2], [11].

Another example, and the main motivation for this research, is the problem of assigning apprentices of different types to placements at large electricity

School of Mathematical and Physical Sciences, University of Technology Sydney
15 Broadway Ultimo NSW 2007, Australia

distributors. Such organisations may take in hundreds of new apprentices each year, and apprenticeships can take several years to complete. Given the motivation of the paper, in what follows the discussion is conducted in terms of apprenticeship planning.

The set of placements is partitioned into a number of placement groups. Each placement in a particular group exposes the apprentices to the same core set of skills. Each apprentice has a set of required groups determined by their apprenticeship type, e.g. line worker, cable jointer, electro-mechanic, etc., and each apprentice must therefore complete one placement from each of their required groups in order to complete their practical training.

Apprentices require supervision; too few apprentices in a placement is impractical as it removes qualified staff from their regular duties with little benefit to the organisation, and too many apprentices in a placement can be too burdensome for the team. Therefore, at any given time, each placement is permitted to have either no apprentices, or between a minimum and maximum number of apprentices. Each apprentice must spend a minimum amount of time in each assigned placement, typically a number of weeks or months, in order to gain the necessary skills and experience. There is no maximum permissible amount of time for placements.

Although any placement in a particular group provides the same type of skills, suitability for each apprentice varies from placement to placement. For example, the travel distance for the apprentice is a serious consideration during the assignment process. Suitability and personal preferences for placements are modelled by means of a penalty for each apprentice-placement pair per unit time. The goal is to minimise the total penalty over a given planning horizon.

Much of the literature on educational timetabling focuses on high school and university course and examination timetabling [3], [4]. To the authors' knowledge, the considered problem of workplace training timetabling has not received the attention in the literature that it deserves. Among few publications relevant to the workplace training timetabling problem considered in this paper, [6] considers the resident scheduling problem, a special case of the multi-period staff assignment problem [1], in which medical residents are assigned to rotations at hospitals. The authors present a decision support system in which feasible integer solutions were successfully obtained by means of a rounding heuristic applied to the linear relaxation of the binary integer programming model.

The problem studied in this paper can be viewed as a generalisation of the classical open shop scheduling problem (OSSP) [12]. In the OSSP, there is a set of machines and a set of jobs. Each job must be processed on each of the machines in any order, and the processing time of each job on each machine is specified. Each machine can process as most one job at a time, and each job can be processed on at most one machine at a time.

In contrast to the OSSP, in the problem considered in this paper, each placement (which can be interpreted as a machine) can accommodate several apprentices (which can be viewed as jobs) simultaneously. The duration of the placement (which can be interpreted as the processing time of a job on

a machine) in our problem is variable, and must be at least the minimum required time for this placement. The requirement of OSSP that each job must be processed on each machine is replaced by the partition of all placements (machines) into groups; the association of each apprentice (job) to a subset of these groups; and the requirement that each apprentice must be assigned to exactly one placement in each associated group. Furthermore, the objective function in our problem differs from the objective functions commonly used in publications on OSSP.

It is well-known that the classical OSSP is a difficult one. In particular, it is NP-hard for the criterion of makespan [8]. Since in the problem, considered in our paper, the planning horizon is given, which imposes a restriction on the makespan, even the problem of answering whether or not there exists a feasible solution for the workplace training timetabling problem is NP-complete. The complexity of the feasibility problem is caused also by the restrictions on the number of apprentices allowed by each placement. In the formulation below, the planning horizon is partitioned into intervals of equal length. It will be shown in Section 3 that even if the planning horizon is comprised of just a single interval, the feasibility problem remains NP-complete.

The remainder of the paper is organised as follows. Section 2 presents the integer programming (IP) formulation for the considered problem. Section 3 discusses the computational complexity. Section 4 presents the entire optimisation procedure. Section 5 presents the results of the computational experimentation. Section 6 gives concluding remarks.

2 Problem Formulation

Let $\mathcal{N} = \{1, \ldots, N\}$ be the set of apprentices, and $\mathcal{M} = \{1, \ldots, M\}$ be the set of placements. $\mathcal{M}$ is partitioned into placement groups $\{\mathcal{M}_1, \ldots, \mathcal{M}_k\} = \hat{\mathcal{M}}$. Define $G_i \subseteq \hat{\mathcal{M}}$ to be the set of required placement groups for $i \in \mathcal{N}$. The planning horizon is discretised into a number of time periods of equal length indexed from $1, \ldots, T$, and define $\mathcal{T} = \{1, \ldots, T\}$. Define $X_{i,j,t}$ to be 1 if $i \in \mathcal{N}$ starts $j \in \mathcal{M}$ at or before the beginning of period $t \in \mathcal{T}$, or 0 otherwise. Define $Y_{i,j,t}$ to be 1 if $i \in \mathcal{N}$ completes $j \in \mathcal{M}$ before the beginning of period $t \in \{1, \ldots, T+1\}$, or 0 otherwise. In order to facilitate the exposition of the model, we extend the planning horizon by an addition period $T+1$. Without the additional period $T+1$ used for the Y variables, no placement can take place at time T. $Z_{j,t}$ is 1 if $j \in \mathcal{M}$ contains apprentices at $t \in \mathcal{T}$, or 0 otherwise. The constant $c_{i,j}$ is the penalty per unit time of assigning $i \in \mathcal{N}$ to $j \in \mathcal{M}$. The constant l_j is the minimum duration for $j \in \mathcal{M}$. The constants a_j and b_j are the minimum and maximum number of apprentices that $j \in \mathcal{M}$ can hold, respectively. The following Integer Program (IP) describes the problem:

$$\text{(P) Minimise:} \quad \sum_{i=1}^{N} \sum_{j=1}^{M} \sum_{t=1}^{T} c_{i,j}(X_{i,j,t} - Y_{i,j,t}) \tag{1}$$

Subject To: $X_{i,j,t} \leq X_{i,j,t+1} \quad i = 1,\ldots,N; j = 1,\ldots,M; t = 1,\ldots,T-1$

(2)

$$Y_{i,j,t} \leq Y_{i,j,t+1} \quad i = 1,\ldots,N; j = 1,\ldots,M; t = 1,\ldots,T \tag{3}$$

$$X_{i,j,T} = Y_{i,j,(T+1)} \quad i = 1,\ldots,N; j = 1,\ldots,M \tag{4}$$

$$X_{i,j,t} \geq Y_{i,j,t} \quad i = 1,\ldots,N; j = 1,\ldots,M; t = 1,\ldots,T \tag{5}$$

$$\sum_{j \in g} X_{i,j,T} = 1 \quad i = 1,\ldots,M; g \in G_i \tag{6}$$

$$\sum_{j=1}^{M} (X_{i,j,t} - Y_{i,j,t}) \leq 1 \quad i = 1,\ldots,N; t = 1,\ldots,T \tag{7}$$

$$\sum_{t=1}^{T} (X_{i,j,t} - Y_{i,j,t}) \geq l_j X_{i,j,T} \quad i = 1,\ldots,N; j = 1,\ldots,M \tag{8}$$

$$\sum_{i=1}^{N} (X_{i,j,t} - Y_{i,j,t}) \geq a_j Z_{j,t} \quad j = 1,\ldots,M; t = 1,\ldots,T \tag{9}$$

$$\sum_{i=1}^{N} (X_{i,j,t} - Y_{i,j,t}) \leq b_j Z_{j,t} \quad j = 1,\ldots,M; t = 1,\ldots,T \tag{10}$$

$$\sum_{i=1}^{N} \sum_{j=1}^{M} X_{i,j,1} \geq 1 \tag{11}$$

$$X_{i,j,t} \in \{0,1\} \quad i = 1,\ldots,N; j = 1,\ldots,M; t = 1,\ldots,T \tag{12}$$

$$Y_{i,j,t} \in \{0,1\} \quad i = 1,\ldots,N; j = 1,\ldots,M; t = 1,\ldots,T+1 \tag{13}$$

$$Z_{j,t} \in \{0,1\} \quad j = 1,\ldots,M; t = 1,\ldots,T \tag{14}$$

where the expression $(X_{i,j,t} - Y_{i,j,t})$ results in 1 if apprentice $i \in \mathcal{N}$ is to attend placement $j \in \mathcal{M}$ at $t \in \mathcal{T}$, or 0 otherwise. Constraints (1) describes the objective, which is to minimise the total assignment penalty. Constraints (2) and (3) establish the temporal relationships between the $X_{i,j,*}$ and $Y_{i,j,*}$ variables, respectively. Constraints (4) ensure that placements must finish if they start, and (5) ensure that placements must start before they finish. Constraints (6) ensure that each apprentice's requirements are satisfied. Constraints (7) ensure that no apprentice is required to attend more than one placement at any given time. Constraints (8) ensure that apprentices spend the minimum amount of time in their assigned placements, and (9) and (10) ensure that each placement $j \in \mathcal{M}$ has either 0, or between a_j and b_j apprentices, respectively. The constraint (11) reduce the feasible region by forcing at least some assignments to happen at the beginning of the planning horizon.

3 Complexity of Finding a Feasible Solution

In this section we show that even the problem of detecting the existence of a feasible solution when $T = 1$ and there exists only one placement group is NP-complete. It is easy to see that a feasible solution for the problem with $T = 1$ and a single placement group exists if and only if there is a solution for the following system of two inequalities:

$$a_1x_1 + \ldots + a_Mx_M \leq N \tag{15}$$

$$b_1x_1 + \ldots + b_Mx_M \geq N \tag{16}$$

where $x_i \in \{0, 1\}$ for all $1 \leq i \leq M$. Let c be a nonnegative integer. Denote by PL(c) the decision problem, requiring to answer whether or not there exists a solution for (15) - (16), where $a_1, \ldots, a_M$ and $b_1, \ldots, b_M$ satisfy the inequalities

$$b_i - a_i \geq c \qquad \text{for all } 1 \leq i \leq M. \tag{17}$$

Observe that c is not part of the input, i.e. the input is only positive integers N, a_1, ..., a_M and b_1, ..., b_M, where all pairs a_i, b_i satisfy (17). In other words, all instances of PL(c) satisfy (17) for the same c.

In order to prove that, for any nonnegative integer c, PL(c) is NP-complete, consider the following decision problem which will be referred to as PARTITION:

Input: positive integers u_1, ..., u_n such that $\sum_{i=1}^{n} u_i$ is even.

Question: does there exists a solution to the equation

$$u_1x_1 + \ldots + u_nx_n = \frac{1}{2}\sum_{i=1}^{n} u_i, \tag{18}$$

where $x_i \in \{0, 1\}$ for all $1 \leq i \leq n$?
It is well known that PARTITION is NP-complete [7]

Lemma 1 *For any positive integer c, PARTITION $\propto$ PL(c) and therefore PL(c) is NP-complete.*

Proof Let u_1, ..., u_n be an arbitrary instance of the PARTITION and let c be an arbitrary nonnegative integer. If $c = 0$, then PARTITION $\propto$ PL(c) is achieved by setting $M = n$, $N = \frac{1}{2}\sum_{i=1}^{n} u_i$, and

$$a_i = b_i = u_i \qquad \text{for all } 1 \leq i \leq n.$$

If $c > 0$, then the corresponding instance of PL(c) is constructed as follows: $M = n$,

$$a_i = ncu_i \qquad \text{for all } 1 \leq i \leq n$$

$$b_i = a_i + c \qquad \text{for all } 1 \leq i \leq n. \tag{19}$$

and

$$N = \frac{nc}{2} \sum_{i=1}^{n} u_i.$$

Supposed that (18) holds for some $x'_1, \ldots, x'_n$. Then, by multiplying (18) by nc and taking into account that $M = n$,

$$a_1 x'_1 + \ldots + a_M x'_M = N$$

which by virtue of (19) gives also (16).

Supposed that (18) does not have a solution. Consider arbitrary $x'_1, \ldots, x'_n$ such that

$$a_1 x'_1 + \ldots + a_M x'_M \leq N.$$

By dividing by nc and taking into account that $M = n$,

$$u_1 x'_1 + \ldots + u_n x'_n \leq \frac{1}{2} \sum_{i=1}^{n} u_i.$$

Hence,

$$x'_1 + \ldots + x'_n < n. \tag{20}$$

Furthermore,

$$u_1 x'_1 + \ldots + u_n x'_n < \frac{1}{2} \sum_{i=1}^{n} u_i,$$

because of the assumption that (18) does not have a solution. Since the left-hand side and the right-hand side are integer,

$$\frac{1}{2} \sum_{i=1}^{n} u_i \geq u_1 x'_1 + \ldots + u_n x'_n + 1$$

which, by multiplying by nc and taking into account (20), $M = n$ and $N = \frac{nc}{2} \sum_{i=1}^{n} u_i$, gives

$$N \geq a_1 x'_1 + \ldots + a_M x'_M + Mc > a_1 x'_1 + \ldots + a_M x'_M + c(x'_1 + \ldots + x'_M)$$
$$= b_1 x'_1 + \ldots + b_M x'_M.$$

Hence, $x'_1, \ldots, x'_M$ is not a solution to (16)-(15).

Consequently, (18) has a solution if and only if the system (16)-(15) has a solution. □

4 Solution Approach

The problem is very challenging, and the required computational effort for straight-forward solution of (P) grows very rapidly as the problem size increases. Even for very small test cases, a commercial IP solver struggles to solve it on a modern stand-alone computer. Since real world problems are much larger, typically involving hundreds of apprentices, we propose a sequential construction stage to produce an initial feasible solution, followed by an improvement stage based on a local search metaheuristic. The details of these approaches are given in the remainder of this section.

4.1 Sequential Construction

Solving (P) directly for hundreds of apprentices is not currently possible in practically acceptable time. It is, however, possible to incrementally construct a complete solution by scheduling a small set of apprentices $\hat{\mathcal{N}} \subseteq \mathcal{N}$ at a time by solving an IP model. A similar IP-based constructive approach was considered in Czibula et al. [5] for a class timetabling problem.

In order to incrementally construct a schedule, we first introduce an augmented version of (P):

$$\text{(AP) Minimise:} \quad \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{t=1}^{T} c_{i,j}(X_{i,j,t} - Y_{i,j,t}) + \mu \sum_{j=1}^{M}\sum_{t=1}^{T}(U_{j,t} + V_{j,t}) \quad (21)$$

$$\text{Subject To:} \quad (2) - (8)$$

$$\sum_{i=1}^{N}(X_{i,j,t} - Y_{i,j,t}) + U_{j,t} \geq a_j Z_{j,t} \quad j = 1, \ldots, M; t = 1, \ldots, T \quad (22)$$

$$\sum_{i=1}^{N}(X_{i,j,t} - Y_{i,j,t}) - V_{j,t} \leq b_j Z_{j,t} \quad j = 1, \ldots, M; t = 1, \ldots, T \quad (23)$$

$$(11) - (14)$$

$$U_{j,t} \in \mathbb{Z}^+ \quad j = 1, \ldots, M; t = 1, \ldots, T \quad (24)$$

$$V_{j,t} \in \mathbb{Z}^+ \quad j = 1, \ldots, M; t = 1, \ldots, T \quad (25)$$

where the augmenting variables $U_{j,t}$ and $V_{j,t}$ represent the shortfall and excess apprentices in placement $j \in \mathcal{M}$ at time $t \in \mathcal{T}$, respectively. The objective function (21) penalises these values, weighted by a very large coefficient μ.

The (AP) model permits solutions that violate the constraints (9) and (10) from (P), albeit with considerable penalty. Allowing these violations is necessary when constructing a solution a small number of apprentices at a time. For example, if $\min_{j \in \mathcal{M}} a_j > |\hat{\mathcal{N}}|$, then no feasible solution can exist. Even when $\min_{j \in \mathcal{M}} b_j < |\hat{\mathcal{N}}|$, a feasible solution may not exist without the augmenting variables, even if a feasible solution exists for (P).

Apprentices are considered one or more at a time, in a particular order. At each iteration, the considered apprentices $\hat{\mathcal{N}}$ are scheduled by solving (AP), subject to all apprentices that have been scheduled in previous iterations. A complete solution is produced once all apprentices have been scheduled. The order in which the apprentices are scheduled is likely to have a significant affect on the final constructed solution. We propose three basic ordering rules, and their relative performance is presented in Section 5:

1. Indexed order: Schedule apprentices in the order in which they are defined.
2. Random order: Schedule apprentices at random.
3. Average placement weighted penalty: Schedule apprentices in decreasing order of $\sum_{g \in G_j}(\sum_{j \in g}(l_j c_{i,j})/|g|)$.

While the sequential construction approach can produce a solution with substantially less computational effort than by direct solving (P), the solution quality is not guaranteed to be good. The produced solution is, however, a good starting point for some other solution approaches, such as local search metaheuristics.

4.2 Simulated Annealing

The solution obtained from the sequential constructive approach may be improved by a local search metaheuristic such as Simulated Annealing (SA) [9]. A straight forward implementation of SA is proposed.

A solution is represented by the set of all apprentices, each of whom is characterised by: ***(i)*** a list of placements they will attend, ***(ii)*** a list of durations corresponding to their attended placements, and ***(iii)*** a list of starting offsets—the starting offset of a given placement for an apprentice is the unallocated time between the start of the placement and end the of the previous placement.

The neighbourhood function produces at each iteration a candidate neighbour solution σ', based on the current solution σ, by applying one of the following operations to a random apprentice:

- swapping the order of two placements at random,
- substituting one random placement for another from the same placement group,
- incrementing or decrementing the amount of time spent at a random placement, or
- incrementing or decrementing the starting offset for a random placement.

Selection of the apprentice and neighbourhood operation is at random with uniform probability. The first and and second types of operations listed above will always produce a feasible solution to the augmented (AP) model. The third and fourth types of operations should be applied in such a way that the minimum placement duration is not violated, and that starting offsets are always non-negative.

5 Results

In order to test the proposed solution approaches, a set of 36 test cases were randomly generated with similar characteristics to real-world cases found at Ausgrid, Australia's largest electricity distributor. All test cases and corresponding solution files can be downloaded from `https://goo.gl/c8yt8H`. The testing system was an Intel i7-4790K quad core CPU with 16GB RAM, running Microsoft Windows 10 64-bit. Code was written in C# 4.0, and we used IBM ILOG CPLEX 12.5.0.0 64-bit using the ILOG Concert API to solve the mathematical programming models. In each case, CPLEX was given a time limit of two hours.

Case	$\|\mathcal{N}\|$	G	$\|\mathcal{M}\|$	ρ	Case	$\|\mathcal{N}\|$	G	$\|\mathcal{M}\|$	ρ
01	100	4	16	2	19	200	4	16	2
02	100	4	16	4	20	200	4	16	4
03	100	4	32	2	21	200	4	32	2
04	100	4	32	4	22	200	4	32	4
05	100	4	64	2	23	200	4	64	2
06	100	4	64	4	24	200	4	64	4
07	100	8	16	2	25	200	8	16	2
08	100	8	16	4	26	200	8	16	4
09	100	8	32	2	27	200	8	32	2
10	100	8	32	4	28	200	8	32	4
11	100	8	64	2	29	200	8	64	2
12	100	8	64	4	30	200	8	64	4
13	100	16	16	2	31	200	16	16	2
14	100	16	16	4	32	200	16	16	4
15	100	16	32	2	33	200	16	32	2
16	100	16	32	4	34	200	16	32	4
17	100	16	64	2	35	200	16	64	2
18	100	16	64	4	36	200	16	64	4

Table 1 Outline of the 36 test cases

We set $T = 1.5 \times \max_{i \in \mathcal{N}} \left\{ \sum_{g \in G_i} \max_{j \in g} \{l_j\} \right\}$ in order to approximate the required length of the timeline. We constructed timetables with the sequential heuristic using all three list permutations mentioned in Section 4.1. In each case, we scheduled apprentices one at a time. We then chose the timetable with the best objective value and used that as a starting point for the SA metaheuristic mentioned in Section 4.2.

Neighbour solutions in SA were chosen randomly with uniform probability. We set the initial temperature to 10^6, the termination temperature to 10^{-3}, and used a geometric cooling rate of 0.99, which was applied at each iteration. These values were obtained using trial-and-error on some small, medium, and large test cases. It is a matter of future research to determine what parameter values perform best.

Table 1 outlines the 36 test cases that were tested. The tables outlines the test case identifier (Case), the number of apprentices ($|\mathcal{N}|$), the total number of placement groups (G), the number of placements ($|\mathcal{M}|$), and the number of groups per student (ρ).

Table 2 shows the time taken, in minutes, for each procedure tested. The columns are as follows: The test case (Case); the time taken for sequential construction using the indexed order (Idx), random order (Rnd), and average placement weighted penalty order (Pty); the time taken for the SA algorithm to converge (SA), and the time taken for the IP to terminate (IP).

The sequential construction procedures took about 3.75 minutes on average for the test cases with 100 students, and about 8.71 minutes on average for the test cases with 200 students. Simulated annealing took about 10.15 minutes on average to converge for the test cases with 100 students, 19.12 minutes to converge for the test cases with 200 students.

Case	Idx	Rnd	Pty	SA	IP	Case	Idx	Rnd	Pty	SA	IP
01	0.3	0.3	0.3	1.5	1.2	19	1.9	1.9	1.9	2.8	4.2
02	0.7	0.8	0.8	4.6	19.3	20	9.7	8.9	9.4	8.2	120
03	0.6	0.6	0.6	1.4	10.8	21	4.3	4.3	4.5	3.3	21.1
04	3.2	3.9	3.7	26.6	120.5	22	16.6	15.5	14.9	32	120.1
05	1.6	1.7	1.6	3.8	120	23	6.2	6.1	6.1	4.3	120
06	7.6	8.1	8.2	8.6	120.1	24	27.4	29.1	30.7	37.17	120
07	0.3	0.3	0.3	4.1	120.2	25	1.6	1.5	1.4	7	2.5
08	3	1.2	1.3	15.8	120	26	13.2	6.6	6.1	3.3	23
09	3.5	2.7	2.6	2.8	6.9	27	3.4	3.2	3.6	3.3	3.1
10	7.6	8.6	7.6	3.9	120	28	11.7	12.2	10.5	58	120
11	5.5	5.5	5.5	2.5	120	29	5.4	5.5	5.6	5.1	71.4
12	14.1	14.6	14.5	83.7	120.1	30	21.8	21.5	22.8	26.9	120
13	1	0.8	0.8	1.6	0.1	31	1.2	1.3	1.1	2.9	0.2
14	2.5	2.5	2.7	7.4	2.4	32	5.7	5.2	5.7	17	7.9
15	1.7	2.2	1.6	4.1	0.9	33	2.2	2.2	2.1	8.9	7.5
16	4.6	4.8	4.9	2.8	4.5	34	7.4	7.5	7.5	5.6	120
17	3.2	3.1	3.1	1.8	30.1	35	2.1	2.1	2.4	30.5	120.6
18	6.7	6.6	6.5	5.7	120	36	18.4	19.5	19.8	88	120

Table 2 The time taken, in minutes, for each tested procedure.

Case	Idx	Rnd	Pty	SA	IP	Case	Idx	Rnd	Pty	SA	IP
01	29	29	31	0	0	19	20	19	19	0	0
02	4	20	18	0	0	20	61	10	10	0	1076
03	0	0	0	0	0	21	0	0	1	0	0
04	12	23	28	0	465	22	30	13	18	0	845
05	0	1	0	0	2	23	0	0	0	0	0
06	0	0	0	0	83	24	4	4	4	1	-
07	122	118	114	18	5	25	118	120	120	18	0
08	78	40	72	5	-	26	2	4	2	0	0
09	9	7	4	0	0	27	12	27	43	0	0
10	1	0	0	0	46	28	219	214	232	1	474
11	1	2	0	0	0	29	0	0	0	0	0
12	8	3	2	2	47	30	1	1	0	0	-
13	11	3	9	8	0	31	111	100	92	0	0
14	13	20	10	3	0	32	214	194	210	101	0
15	4	14	1	3	0	33	58	55	46	1	0
16	0	0	4	0	0	34	0	0	2	0	0
17	0	0	0	0	0	35	14	18	9	5	0
18	0	0	0	0	0	36	1	29	8	2	365

Table 3 $\sum_{j=1}^{M} \sum_{t=1}^{T} (U_{j,t} + V_{j,t})$ for solutions produced by each tested procedure.

The IP approach terminated at the two hour time limit for half of the test cases, although these were spread evenly between those cases with 100 and those with 200 students. The average time overall for the IP approach, including for those that terminated at the time limit, was 66 minutes, and there was no significant difference in average time between the test cases with 100 and with 200 students. For the test cases where CPLEX reported an optimal solution before reaching the time limit, the average solve time was 12 minutes, and again there was no significant difference in the cases where there were 100 or 200 students.

Table 3 shows the total number of violations lower and upper bound violations of placement size, given by $\sum_{j=1}^{M}\sum_{t=1}^{T}(U_{j,t}+V_{j,t})$, for the solutions produced by each procedure. The columns are as follows: The test case (Case); the number of bound violations produced by sequential construction for indexed order (Idx), random order (Rnd), and average placement weighted penalty order (Pty); the number of bound violations for the solution produced by the SA algorithm, and the number of bound violations for the solution produced by the IP approach (IP). CPLEX did not report any integer solutions for test cases `08`, `24`, and `30` prior to terminating at the time limit.

Each of the sequential construction procedures produced violation-free solutions for 10 of the 36 test cases. For 6 of those 10, all three list permutations were able to produce violation-free solutions, whereas for the other 4, some list permutations produced violation-free solutions while other permutations did not. The average number of bound violations were 32, 30, and 31, respectively, for the three tested permutations.

Both SA and IP produced 23 violation-free solutions for the 36 test cases. For 16 of those 23 cases, SA and IP both produced violation-free solutions. For test case `20`, SA produced a solution with no bound violations, while the IP approach produced 1076 violations after terminating at the two hour time limit. For test case `22`, SA again produced a solution with no bound violations, while the IP approach produced a solution with 845 bound violations.

The quality of solutions, with respect to the number of bound violations, produced by SA appear to be of roughly similar quality to the solutions produced by the IP approach. The SA approach, however, did not have a time limit, and took on average about 14.6 minutes to converge, whereas CPLEX was given a two hour limit for the IP approach. It is reasonable to assume that if the SA procedure was tuned to run for a longer duration, it would produce better solutions.

Table 4 shows the solution quality, given by $\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{t=1}^{T} c_{i,j}(X_{i,j,t} - Y_{i,j,t})$, for the solutions free of bound violations produced by each procedure. The columns are as follows: The test case (Case); the solution quality produced by sequential construction for indexed order (Idx), random order (Rnd), and average placement weighted Penalty order (Pty); the solution quality for the solution produced by the SA algorithm, and solution quality for the solution produced by the IP approach (IP). We do not report the solution quality where the number of bound violations is nonzero.

For the 16 cases where both the SA and the IP approach were able to produce solutions free of bound violations, the solutions produced by SA were, on average, 25.35% worse than the solutions produced by the IP approach. CPLEX reported optimal solutions to 12 of these 16 cases. The total solve time for these 16 cases, on average, was 3.35 minutes for SA, and 42.23 minutes for the IP approach.

Overall, it is difficult to say with any great certainty which approach was the clear winner. The three permutations for the sequential construction approach performed roughly as well as each other, and were clearly inferior to the SA and IP approach, however they were only intended to produce quick,

Case	Idx	Rnd	Pty	SA	IP
01	-	-	-	23466.74	12218.00
02	-	-	-	31456.12	19539.15
03	11380.88	11380.88	11380.88	11380.88	8950.06
04	-	-	-	58500.79	-
05	30538.88	-	29993.70	26038.54	-
06	20413.69	20413.69	20413.69	20413.69	-
07	-	-	-		-
08	-	-	-		-
09	-	-	-	34522.71	14452.74
10	-	29381.33	29379.06	29379.06	-
11	-	-	20402.50	20402.50	9927.31
12	-	-	-	-	-
13	-	-	-	-	32044.13
14	-	-	-	-	44299.37
15	-	-	-	-	19000.53
16	48001.84	48003.71	-	48001.45	47292.00
17	25266.19	25267.24	25267.24	23561.66	15840.75
18	23908.51	23910.29	23911.83	23908.51	22841.66
19	-	-	-	33116.71	22657.04
20	-	-	-	77746.94	-
21	28490.52	28484.20	-	27858.08	15261.23
22	-	-	-	81712.63	-
23	12898.13	12899.39	12898.53	12898.13	11411.07
24	-	-	-	-	-
25	-	-	-	-	37379.71
26	-	-	-	89275.82	83455.80
27	-	-	-	36460.34	27029.42
28	-	-	-	-	-
29	41506.05	41644.64	40808.95	35225.88	16731.11
30	-	-	45126.83	45126.83	-
31	-	-	-	70340.18	69970.71
32	-	-	-	-	119561.60
33	-	-	-	-	38290.64
34	102606.22	102603.19	-	102603.19	100615.49
35	-	-	-	-	25512.38
36	-	-	-	-	-

Table 4 $\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{t=1}^{T} c_{i,j}(X_{i,j,t} - Y_{i,j,t})$ for solutions with no bound violations produced by each tested procedure.

initial solutions. The SA and IP approach appear to be roughly similar to one another in solution quality. The SA approach appears to be slightly superior to the IP approach with respect to bound violations, however the IP approach appears superior to the SA approach when looking only at solutions free of bound violations. One factor that cannot be ignored, however, is that the SA approach produced solutions in significantly less time than the IP approach, and also avoids the need to use an IP solver.

6 Conclusion

The paper considers the problem of assigning apprentices to practice placements. It is shown that even the problem requiring the answer to the question:

"Does there exist a feasible solution?", is NP-complete. The paper presents an optimisation procedure comprised of a sequential application of integer linear programming, and a simulated annealing metaheuristic.

The proposed heuristics were tested by means of computational experiments on a number of randomly generated test cases, with similar characteristics to relevant data at Ausgrid, Australia's largest electricity distributor. The straight forward approach of solving the IP model subject to a two-hour time limit produced good solutions in most cases, however this approach is not suitable when the size of the problem grows large. The approach of sequentially constructing a solution one apprentice at a time by solving an augmented version of the IP produced solutions quickly, but with many violations on the lower and upper bounds on the permissible number of students in each placement. The approach of improving a sequentially constructed schedule using simulated annealing was shown to produce solutions of similar quality to the approach of solving the IP, but required substantially less time.

Future work can further investigate different apprentice ordering rules, as well as mapping the performance versus quality trade-off of considering a greater or fewer number of apprentices at each iteration of the sequential constructive stage. For the SA implementation, applying the operations in the neighbourhood function in a non-uniform manner, in order to better direct the optimisation away from poor solutions, as well as other neighbourhood definitions such as Kempe chains, should be investigated.

References

1. Aronson, J.E.: The multiperiod assignment problem: A multicommodity network flow model and specialized branch and bound algorithm. European journal of operational research 23(3), 367–381 (1986)
2. Barnett, T., Cross, M., Jacob, E., Shahwan-Akl, L., Welch, A., Caldwell, A., Berry, R.: Building capacity for the clinical placement of nursing students. Collegian 15(2), 55–61 (2008)
3. Burke, E., Jackson, K., Kingston, J.H., Weare, R.: Automated university timetabling: The state of the art. The computer journal 40(9), 565–571 (1997)
4. Burke, E.K., Petrovic, S.: Recent research directions in automated timetabling. European Journal of Operational Research 140(2), 266–280 (2002)
5. Czibula, O., Gu, H., Russell, A., Zinder, Y.: A multi-stage ip-based heuristic for class timetabling and trainer rostering. Annals of Operations Research pp. 1–29 (2014)
6. Franz, L.S., Miller, J.L.: Scheduling medical residents to rotations: solving the large-scale multiperiod staff assignment problem. Operations Research 41(2), 269–279 (1993)
7. Garey, M.R., Johnson, D.S.: Computers and intractability, vol. 29. wh freeman New York (2002)
8. Gonzalez, T., Sahni, S.: Open shop scheduling to minimize finish time. Journal of the ACM (JACM) 23(4), 665–679 (1976)
9. Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P.: Optimization by simmulated annealing. science 220(4598), 671–680 (1983)
10. Leigh, J., Withnell, N., Finnigan, N., Winder, H., O'Flanagan, S., Buliptt, S., Fishburn, N., Drury, S., Dean, S.: Innovative placement allocation model for pre-registration student nurses. Journal of Education and Training Studies (2014)
11. Murray, S.C., Williamson, G.R.: Managing capacity issues in clinical placements for pre-registration nurses. Journal of Clinical Nursing 18(22), 3146–3154 (2009)
12. Pinedo, M.L.: Scheduling: theory, algorithms, and systems. Springer Science & Business Media (2012)