Proceedings of the 2018 Winter Simulation Conference
M. Rabe, A. A. Juan, N. Mustafee, A. Skoogh, S. Jain, and B. Johansson, eds.

A STOCHASTIC PROGRAMMING APPROACH TO OPTIMAL RECRUITMENT IN AUSTRALIAN NAVAL AVIATION TRAINING

Cameron Pike
Ana Novak

Defence Science and Technology Group
506 Lorimer St
Port Melbourne, VIC 3207, AUSTRALIA

Bill Moran
David Kirszenblat
Brendan Hill

The University of Melbourne
Grattan St
Parkville, VIC 3010, AUSTRALIA

ABSTRACT

We tackle some of the challenges of optimal recruitment strategies of pilots and other aircrew for Royal Australian Navy aviation. Aside from financial costs, too many students in the training system can create bottlenecks and harm morale, while too few poses a risk to delivering operational capability. We propose a stochastic programming approach that can determine the minimum number of students to recruit and how to distribute those students amongst the training system, in order to meet a capability target with a prescribed probability. This approach is parallelizable and reasonably computationally efficient, and utilizes multiple randomized trials of a greedy method to seek a global minimum. Our work provides insights into the relationship between minimization of entry recruitment numbers and minimization of total student numbers in the system, as well as the relationship between risk constraints and convergence to local minima.

1 INTRODUCTION

We aim to find policies for minimal recruitment levels of students to adequately supply and sustain the Royal Australian Navy (RAN) aviation squadrons. The RAN aviation training continuum is a sequence of prescribed courses, each with specified pre-requisites, terminating in operational squadrons. Failure rates in some courses are large and highly variable. Despite this, sufficient pilots are required to graduate to maintain operational capability in the squadrons. To further complicate the problem, wastage rates of operational pilots, observers, and aircrewmen are also variable. Too many students in the system can cause bottlenecks and excessive waiting, can result in loss of morale for the students and, importantly, incur financial cost. The number of students in the system is a cost that needs to be kept low. The problem we address is how to keep training system costs as low as possible while limiting the risk of failure to achieve operational capability to an acceptably low level over a number of years.

We invoke a model based on ideas in stochastic programming (Birge and Louveaux 2011). Failure rates are modeled by course intake numbers m being mapped probabilistically to the number of passing students n according to a specified distribution. A similar model is used for wastage from the squadron. Failure and wastage rate distributions for courses and squadrons can be estimated from historical data; that process is not discussed in this paper. We assume given transition matrices for all courses and squadrons, as well as a pre-determined operational capability for each squadron.

In reality, the number of students that can participate are limited. This is particularly so for the kind of training path discussed here. The instructor-to-student ratio is very low thus emphasizing the pressures to minimize student numbers. Significant and extensive infrastructure such as aircraft and simulators, are needed to train pilots.

The dynamics of this training continuum, specified by the transition matrices of course pass-fail rates and squadron wastage, are controlled at each course and time, by the number of available successful students

978-1-5386-6572-5/18/$31.00 ©2018 IEEE

from the preceding course(s). For recruitment into the initial course, it is reasonable to assume that the number available is unbounded. A later course or squadron, can only recruit from students who have passed the prerequisites (either in the immediately preceding period or at an earlier time) for that course or squadron. This number is limited by the stochastic dynamics of the previous assignment and pass rates of students. The specification of the entry recruitment numbers, and the assignment of available qualified students from previous courses into each course or squadron comprise the control variables here.

Students successfully passing a given course remain in its *buffer* until assigned to a succeeding course or squadron. While in the buffer, a student entails a cost to the system, thereby providing an incentive on the control process to keep numbers in buffers small. On the other hand, retention of some students in a buffer reduces the future risk of not having enough students available for assignment. In a sense, a buffer provides a bank or reservoir to be accessed at future times to mitigate risk.

The control process is based on the current snapshot (*state*) of the system: numbers of students in courses and buffers, and pilots in squadrons. Any proposed solution method must predict several years ahead. Its future predictions are based on unknown future pass-fail rates in the courses and wastage in the squadrons. Accordingly, the expected cost is optimized. Similarly, the number of operational pilots is a random variable dependent on wastage rates and numbers of suitably qualified students available for assignment into the squadron. The control variables are then functions $m = \phi(n)$: how many students m are assigned into the next course/squadron of the (random) number n of available qualified students. Optimal choice of those functions into the future is based on the current state of the system. The constraint that operational capability be achieved is treated as a *chance constraint*. At any time, the number of pilots, random variable N, is intended to achieve operational capability, C, but we can have no guarantees of this. Accordingly, we accept a certain level of risk $\alpha > 0$, and require that $\Pr(N \geq C) > 1 - \alpha$. Evidently the smaller α is, the harder the constraint is to achieve.

A precise formulation of the problem is given in Section 3 along with a simple example that follows on from a literature review. Then we describe a specific case that is close to that found in the RAN training continuum, the approach to its solution, and simulation results. The solution adopted here is, effectively, a greedy approach with multiple random starts to overcome the local minimum problem. Simulation results suggest that this approach is both computationally feasible and capable of finding a global minimum relatively quickly on problem sizes commensurate with the real world problem we address.

It is intended that this optimization method will be integrated with a hybrid discrete event and agent-based simulator designed to provide strategic decision support to RAN aircrew workforce planners. This simulation tool allows the RAN aircrew planners to view the transient flow of students through the continuum and perform *what-if* scenarios on changes to the training pipeline. The optimization technique described in this paper is designed to accept information about the state of the system from the simulator and determine near-optimal student recruitments and assignments that can then be enacted and played-out by the agent-based dynamics of the simulator. The speed and ease with which planners can perform *what-if* scenarios is part of the motivation to make the optimization method run as fast as possible.

2 LITERATURE

Many approaches toward recruitment and manpower supply optimization are centered on mixed integer linear programming methods or Markov models. Mixed integer linear programming (MILP) has proven to be a popular technique in supply chain optimization. The objective is to determine optimal values of a linear supply model constrained by linear inequalities (see, e.g., Vanderbei 1996). It is widely used, but more closely aligned with our application, Akinyele (2007) used MILP to determine an effective manpower size while incorporating global constraints such as production capacity, demand rate, and allowable time-of-operation so as to account for differences between countries' rates of development. Mokhtari and Dadgar (2015) applied MILP alongside simulated annealing to a job-shop scheduling problem with time-varying machine failure rates.

In previous work, the authors successfully applied MILP to the solution of related Australian Defence Force (ADF) timetabling problems (Nguyen et al. 2018; Kirszenblat et al. 2017). To simplify these problems, the uncertainty associated with the exact number of students who pass a particular course was ignored, and pass rates were replaced by their respective means, i.e., expected values. This simplification made the problems amenable to MILP methods. In particular, we applied, on the one hand, a novel combination of a standard MILP in conjunction with a fast technique for solving inexact cover problems (Nguyen et al. 2018) and, on the other hand, MILP column generation (Kirszenblat et al. 2017).

Recruitment and manpower training have been modeled by several authors as stochastic dynamic systems using Markov models. An early development was that of Bartholomew (1971), in which basic recruitment and progression-through-training were defined as prior probabilities of being accepted into a program and progression was defined in terms of Markov transition matrices. This basic model has been applied to a large number of domains and extended to predict future horizons via Dynamic Programming (Bertsekas 2000) and to fit with required training continuum numbers (see, e.g., Mehlmann 1980). For example, Udom (2013) developed a Markov Decision Process (MDP) for optimal control of a Multi-level Hierarchical Manpower System (MHMS) dealing with departmental transfers and movement of staff through organizations. In previous work (Suvorova et al. 2018), we provided an elegant MDP solution to a toy version of the aircrew recruitment problem. In this, both MILP and Dynamic Programming approaches were explored; however, these did not solve the problem of the real Defence network in any reasonable amount of time. The work reported here arose in trying to formulate methods that are scalable to the full continuum.

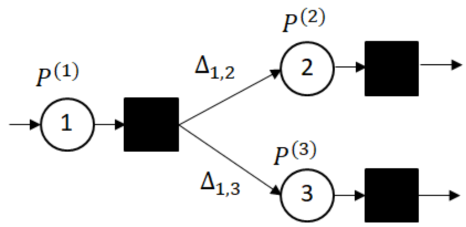
Given the limitations of MILP when it comes to handling the random elements inherent to the problem, i.e. course pass rates, we choose to adopt in this paper an approach that is based on ideas in stochastic programming. Stochastic programming approaches have been applied with success to military applications, e.g., to manpower planning for the Canadian Armed Forces (Martel and Price 1981). Stochastic control methods (Bertsekas and Shreve 1996) have been utilized for controlling risk in an e-commerce warehouse staff planning problem (Wruck et al. 2016). In considering limits on the amount of risk for failing to meet capability targets, we incorporate probabilistic or chance constraints into the problem formulation (Charnes and Cooper 1963). Because of the often large search space of stochastic programming models in similar problems such as job-shop scheduling (Tavakkoli-Moghaddam et al. 2005), stochastic local search methods (Spall 2005; Hoos and Stützle 2004) are often used to obtain sufficiently good solutions to these problems.

In order to consider these stochastic elements for a scalable optimization method, the authors developed two different strategies in parallel. The first is described in this paper in Section 3 and 4. The second (Hill et al. 2018) utilizes a technique of back-filling demand from squadron targets. These targets are then dynamically chosen using Monte Carlo simulations to introduce risk into the model and to determine the required levels of recruitment and student assignment.

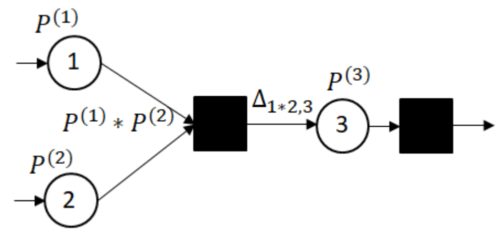
3 PROBLEM FORMULATION

A directed graph $G = (V, E)$ is used to describe the course structure with prerequisites. Each node is a course, and an arrow $e \in E$ from course c_i to course c_j, means that course c_i is an immediate prerequisite of course c_j. The head and tail of an arrow $e \in E$ are denoted by $h(e)$ and $t(e)$, respectively. For simplicity, and in keeping with the particular course structure to be modeled, we assume that the graph is "layered"; that is, $V = \bigcup_{r=0}^{R} V_{r+k}$, where $k \geq 1$. Here, V_r are disjoint subsets of V, and for every $e \in E$, there exists r such that $t(e) \in V_r$ and $h(e) \in V_{r+k}$. We will use the notation c_{rj} to mean a node at layer r, and c_r if there is only one such node. A further simplification that may not, ultimately, be necessary, is that every node c is one of the following types (see Figure 1): *continuation*, with one arrow entering and one exiting; *branching*, with one entry node and several exit nodes; and *joining*, with one exit node and several entry nodes.

At each node (course) c is specified a transition matrix $P^{(c)} = (p_{nm}^{(c)})$, where $p_{nm}^{(c)}$ is the probability that if m students enter the course, n will pass. As stated in the Introduction, these are assumed known, having



(a) Example of a branching node structure.



(b) Example of a joining node structure.

Figure 1: Two different node structures.

been estimated from historical records. A reasonable assumption is that the number of passing students at different nodes are mutually independent, conditioned on the number entering each course. Squadron nodes s, similarly, have transition matrices $P^{(s)} = (p^{(s)}_{nm})$ denoting the probability that of the m current pilots, n will remain at the end of the current time period and $m-n$ will leave. Again this random process is assumed to be independent of the course pass-fail processes. Each course node has a buffer that retains all passing students from that node until they are assigned into a subsequent course or squadron. Thus, each node c has attached a number of students s_c and a number in the buffer, b_c.

There may be multiple initial nodes from which it is possible to recruit students. Also there will be multiple final nodes — *squadrons*. A key feature of this problem is that, at each of these squadrons, it is important to retain an *operational capability*: there need to be enough pilots to perform the required functions of the squadron. We assume an operational capability of N_s for $s \in S$.

It is assumed, again for simplicity, that the exiting of students from courses into buffers and assignment into buffers from prior courses are synchronous across the entire graph.

Students in the system are a cost to be minimized while, of course, maintaining capability over several years. We assume, again for simplicity, that each student imposes the same cost on the system, no matter their level of progress through the training continuum, however, this could easily be modified. We also assume, that once a student fails, they are no longer a part of training continuum.

Maintenance of operational capability, of course, cannot be guaranteed. For instance, it is possible, though improbable, for all students in a given course to fail. Accordingly, rather than an infeasible requirement that operational capability is achieved, we impose the softer constraint, chance constraint (Charnes and Cooper 1963), that the probability of achieving capability is at least $1-\alpha$, where $0 < \alpha < 1$. Evidently the smaller α, the harder this constraint is to achieve.

At our disposal to control the system are the number of students recruited from outside into the initial course, and the assignment of students from earlier courses into later ones. These are the control variables. A control policy requires, for each arrow e in the graph, and each year t, a function ϕ_e^t that assigns for each number of available students in $t(e)$, including both passing students and those in the buffer, a number to be assigned to the following course $h(e)$. At an initial node, c, the control variable is a single number m_c^t at each time t, representing the number of recruits entering the initial courses of the training continuum.

The control problem is posed as a minimization of the total number of students conditioned on achievement of operational capability with probability $1-\alpha$. Initial values B_c^0, N_c^0, N_s^0 of students/pilots in buffers, courses, and squadrons are assumed known. A policy in terms of initial recruitments m_c for $c \in V_0$ and functions ϕ_e^t for $e \in E$, to plan T years ahead is to be designed. Let $\mathrm{E}_\mathbf{p}$ denote the mathematical expectation with respect to the parameter vector $\mathbf{p}$ of variable pass rates p^c_{nm}. Formally, with $\mathbb{N}$ denoting the non-negative integers,

$$
\begin{aligned}
&\min_{\phi_e^t, m_{c_0}} \mathbb{E}\Big[\sum_{t=1}^{T}\sum_{c\in V} b_c^t + m_c^t\Big] \quad \text{subject to } \Pr(n_s^t \geq N_s) \geq 1-\alpha \\
&\quad \text{where:} \\
&\qquad m_c^t = \sum_{e:h(e)=c} \phi_e^t(b_{t(e)}^{t-1} + n_{t(e)}^{t-1}), \\
&\qquad b_c^t = b_c^{t-1} + n_c^t - \sum_{e:c=t(e)} \phi_e^t(b_c^{t-1} + n_c^t), \\
&\qquad P^{(c)}(n_c^t = n | m_c^{t-1} = m) = p_{nm}^{(c)}, \\
&\qquad b_c^0 = B_c^0,\ n_c^0 = N_c^0,\ n_s^0 = N_s^0, b_c^t, n_c^T, m_c^t \in \mathbb{N}, (c_0 \in V_0,\ e \in E,\ c, s \in V,\ t = 1, 2, \ldots, T).
\end{aligned}
\tag{1}
$$

The control functions ϕ_e^t assigned to edge e at time t are implemented as (sparse) $0-1$ matrices $\Delta^{t,e} = (\delta_{r,k}^{t,e})$:

$$
\delta_{r,k}^{t,e} = \begin{cases} 1 & \text{if } \phi(r) = k \\ 0 & \text{otherwise.} \end{cases} \tag{2}
$$

As a result of the non-negativity of buffer sizes, Equation (1) enforces a flow conservation constraint on the δ-matrices associated with all edges leaving a given course c:

$$
\sum_{e:c=t(e)} \sum_{k} \delta_{r,k}^{t,e} k \leq r, \tag{3}
$$

so that the number of students from a given buffer cannot exceed the number in the buffer together with newly graduating students from the preceding course. While, in theory, these δ-matrices are infinite, they are constrained by reasonable numbers of students in the system, restrictions on course sizes and the (finite) time horizon, so that buffer sizes are limited.

3.1 Continuation Type

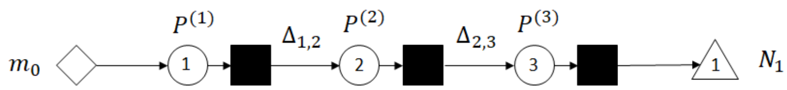
If there is just a linear progression of courses c leading from initial recruitment to deployment in a squadron, then a simple calculation using independence of pass-fails between courses yields that the operational capability constraint and the expected cost for a single year and three courses, as shown in Figure 2 are given by the following equations:

$$
\begin{aligned}
&1-\alpha \geq P^{(3)}(n_3|m_3)\Delta_{2,3}(m_3,n_2)P^{(2)}(n_2|m_2)\Delta_{1,2}(m_2,n_1)P^{(1)}(n_1|m_1) \\
&Y = m_1 + \sum m_2\Delta_{1,2}(m_2,n_1)P^{(1)}(n_1|m_1) + \sum m_3\Delta_{2,3}(m_3,n_2)P^{(2)}(n_2|m_2)\Delta_{1,2}(m_2,n_1)P^{(1)}(n_1|m_1)
\end{aligned}
\tag{4}
$$

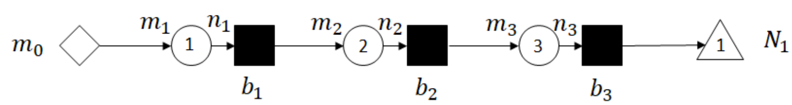
where Y is the cost as the expected number of students in the system. It can be seen that for the linear case, the chance constraint is computed from a Markov Chain with interspersed δ-matrices.

3.2 Branching Type

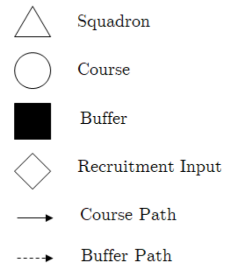
To handle course structures with branching pathways, as shown in Figure 1a, a similar formulation to that in (4) can be used but it must include the flow conservation constraint given in (3). In this way, the δ-matrices of the *branching* must be chosen such that, for a given input m_c^t (corresponding to the rows of the δ-matrices), the summation of the output of students into each branch (corresponding to the column index of the *one* in the m_c^t row of the δ-matrices) does not exceed m_c^t. The matrices are lower triangular reflecting the impossibility of sending more students out than the number coming in.



(a) Probability transition and δ-matrices representation.



(b) Students in courses and buffers representation.



(c) Legend for course structure diagrams.

Figure 2: Example of three course linear structure.

3.3 Joining Type

To handle course structures with joining pathways, shown in Figure 1b, multiple functions ϕ_e^t are needed to express the number assigned from c_i (via e_i) into c at time t. We reduce the complexity by effectively constructing a virtual buffer to retain the qualified but as yet unassigned students from all of the courses c_i. Then we only need one function, or equivalently, δ-matrix to control assignment from all of c_i into c. In order to effect this virtual buffer scheme, it is useful to combine the probability transition matrices at the various nodes c_i into a single one. This is accomplished through a convolution of the probability transition matrices from each node, which for a two node join is given by Equation (5).

$$P^{(1)} * P^{(2)} = \sum_{r=0}^{n} P^{(1)}(r|n_1)P^{(2)}(n-r|n_2) \tag{5}$$

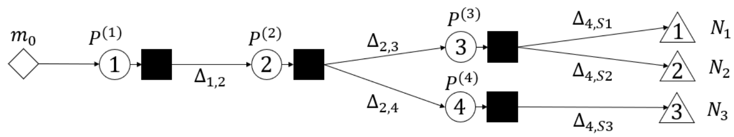
The convolution probability transition matrix must account for every possible input from all edges e_i leading into the join and account for every possible output from the sum of those inputs. Therefore, the new probability transition matrix is multi-dimensional, with a dimension for the input of students n_i from each edge e_i, and a single dimension for the output. It is also larger than the size of the incoming matrices, as it must account for the summation of the largest inputs from n_i.

3.4 Multiple Time Periods

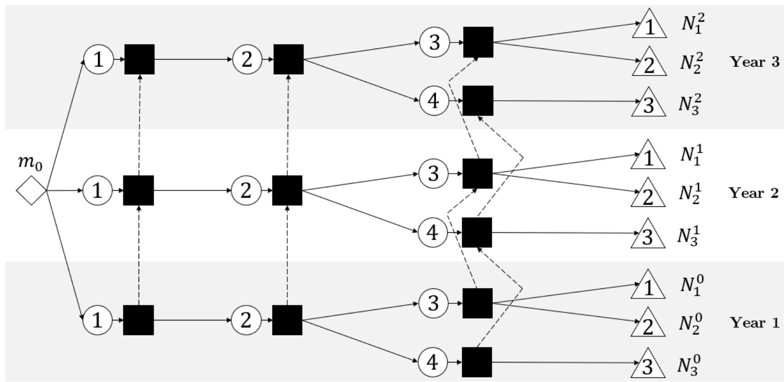
The model described so far only handles a single time period. To move the simple formulation into multiple time periods, we repeat the course structure by the number of periods and join the course structures for each time period by buffer arrows. An example of this is shown in Figure 3. Here, Figure 3a is the part of the training continuum actually being analyzed. Figure 3b is a yearly temporal representation. For instance, the students that are in the buffer directly following the first course in the first year, can either be held in the buffer until the second year or they can proceed with the next course (Course 2) in the same year (Year 1). The joining structures in this case are slightly different to those for joining course pathways. There are no probability transition matrices between the buffers, so the convolution is performed with the transition matrix of course node c and an identity matrix representing the lack of stochasticity from buffer b.

3.5 Specific Case

To test the validity and performance of the model, a part of the RAN course structure as shown in Figure 3a was used to run simulations, performed over a three year period on the representative course structure given in Figure 3b. This case involves the first two node types and joining buffer types over multiple years.



(a) Single year structure.

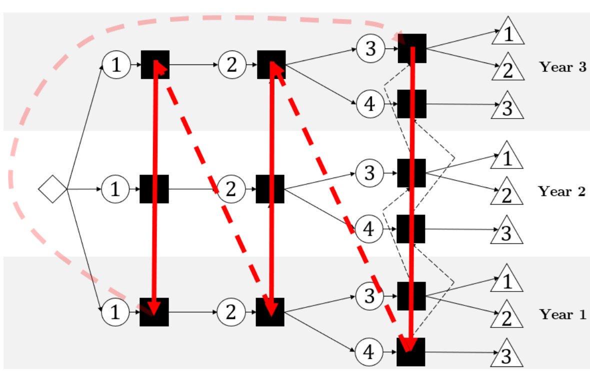


(b) Three year structure.

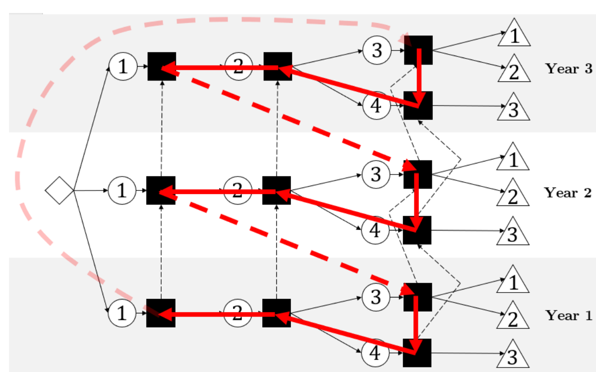
Figure 3: Part of the RAN course structure with a 3 year representation depicting the action of the buffers.

4 APPROACH

Our aim is to produce an algorithm to give close to optimal solutions to the problem described in (1), with given current buffer numbers, operational capability, and risk of achievement of the latter. This computation needs to be done in a relatively short time (a few minutes). Computation of the global optimal choice of matrices $\Delta^{t,e}$ over a reasonably large sized graph over even a few years is infeasible here. Instead, we resort to a suboptimal approach, referred to as the *Delta-Cycling Algorithm* (DCA). This first involves cycling through the nodes and their δ-matrices, of which there are many different ways as illustrated in Figure 4.



(a) Time boustrophedon.



(b) Progress boustrophedon.

Figure 4: Methods of cycling through courses.

For each δ-matrix, the algorithm starts at the bottom row and determines the change in cost for increasing or decreasing the output by one for a given input. In terms of the values of the δ-matrix, this means shifting the *1* in that row either to the left or to the right, and calculating the cost change associated with doing so, here on out referred to as *one-shifting*. As such, this part of the algorithm is effectively performing a descent local search. A one-shift that violates the chance constraint is considered to have an infinitely high cost. The one-shift that generates the greatest reduction in the cost function is the one-shift that is performed on that row of the δ-matrix. If both one-shifting left and right increases the cost, then neither one-shift is performed. The process is then repeated with the next row above and so on. Once the top row has been completed, the process is restarted from the bottom row and repeated, until there is no one-shift that can be actioned to decrease the cost without violating the constraints. This process is shown in Figure 5.

$$\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&1&0&0\\0&\leftarrow&1&\rightarrow\end{bmatrix} \Rightarrow \begin{bmatrix}1&0&0&0\\0&1&0&0\\\leftarrow&1&\rightarrow&0\\0&0&0&1\end{bmatrix} \Rightarrow \begin{bmatrix}1&0&0&0\\\leftarrow&1&0&0\\1&0&0&0\\0&0&0&1\end{bmatrix} \Rightarrow \begin{bmatrix}1&0&0&0\\0&1&0&0\\1&0&0&0\\0&0&0&1\end{bmatrix} \Rightarrow \begin{bmatrix}1&0&0&0\\0&1&0&0\\1&0&0&0\\0&0&\leftarrow&1\end{bmatrix}$$

Figure 5: One-shifting within a single δ-matrix.

Importantly, the flow conservation constraint described in (3) must be adhered to for every one-shift in the δ-matrices. Each one-shift — or change in the output — in the δ-matrices must correspond to an equal and opposite one-shift in one of the dependent matrices (which correspond to the other branches at that node) in order to preserve the number of students. Thus, when calculating the cost change for a one-shift, it must be summed with the smallest increase (or largest decrease) in cost for the opposite one-shift in one of the dependent δ-matrices.

This process is performed for all δ-matrices. The algorithm will also cycle through the entire course structure and continue to do so until no more one-shifts are possible. It is evident that this approach will only find a local cost minimum. The greedy nature of the one-shift selection, means that one-shifting in a single row of a single δ-matrix may limit choices of one-shifting in another by dint of hitting the limits of the chance constraint. As a result, the initialization of the δ-matrices and the order in which they are optimized becomes very important. Techniques such as simulated annealing instead of descent could be used to avoid becoming stuck in a local minimum. We have resorted to a simpler approach, as the descent method is relatively computationally quick. We randomly seed the greedy search by choosing a number of random, but viable, initializations of the δ-matrices. The Delta-Cycling Algorithm is applied for each of these initial random seeds and the optimal among them chosen.

To perform several trials with different initializations, the algorithm must compute quickly. Its speed is governed by the size of the δ-matrices, which may be very large if student and buffer numbers are large. Efficiency is increased by only optimizing the δ-matrices for a small bounded viable number of rows and columns (*region*), excluding inputs or outputs that are either too low or far exceed what is needed to meet the required capability, as shown in Figure 6.

To determine these δ-matrix bounds, worst-case and best-case scenarios are considered. For the minimum bound, the best-case at each squadron of $\Pr(n_s^T \geq N_s) = 1-\alpha$ is first calculated. The input m_s^t to achieve this is determined from a lookup table of the course probability distribution. The process is repeated for the preceding course node and continues moving backward through the directed graph, calculating new values of m_c^t and n_c^t, where the probability of obtaining the required output is always $1-\alpha$. This best-case scenario, underestimates assignment/recruitment numbers as the risk at an individual node will have to be larger than $1-\alpha$ so as to meet the chance constraint $\Pr(n_s^t \geq 1-\alpha)$ because of compounding of risk (multiplication of probabilities). The m_c^t and n_c^t values then serve as the lower bounds for the viable region of each δ-matrix.

Likewise, upper bounds are found by the same method, but for an individual course probability of $1-\varepsilon$ where ε is a much smaller number. This use of $\varepsilon > 0$ prevents an infinite output of students that arises from certainty of achieving capability. These bounds along with the flow constraint (unable to have a larger output than input) describe a smaller *viable region* to be optimized for each δ-matrix. As the worst-case and best-case scenarios are unlikely to occur, a tolerance can be specified to tighten these bounds as desired.

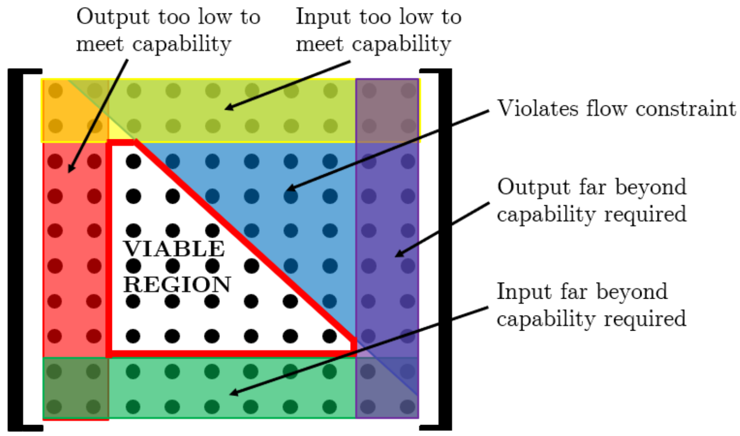


Figure 6: The viable region for optimizing a δ-matrix.

5 SIMULATION RESULTS

The Delta-Cycling Algorithm was implemented in Python for the course structure given in Figure 3b. The distributions were Binomial with a parameter p of 0.8 for course pass rates and 0.95 for squadron wastage. A yearly squadron target of four was desired with 80% probability of meeting capability.

Twelve trials were conducted with randomized initial δ-matrices that adhered to the flow conservation constraint. The optimization order was that given in Figure 4b and individual δ-matrices were optimized from bottom to top (greatest input to least input). The initial number of students in the system (including in the buffers) was initialized to zero.

The simulation was run on an Intel Core i7 7500U processor with 16GB of RAM. The total simulation time for all twelve trials was 501.04 seconds (8.35 minutes) with an average of 41.75 seconds per trial. Along with the main simulation results, computation speed trials were conducted to determine the scalability of the algorithm with regards to the number of time periods and size of matrices. These times are given in Table 1.

Table 1: Computational speed for varying time steps and matrix sizes (minutes).

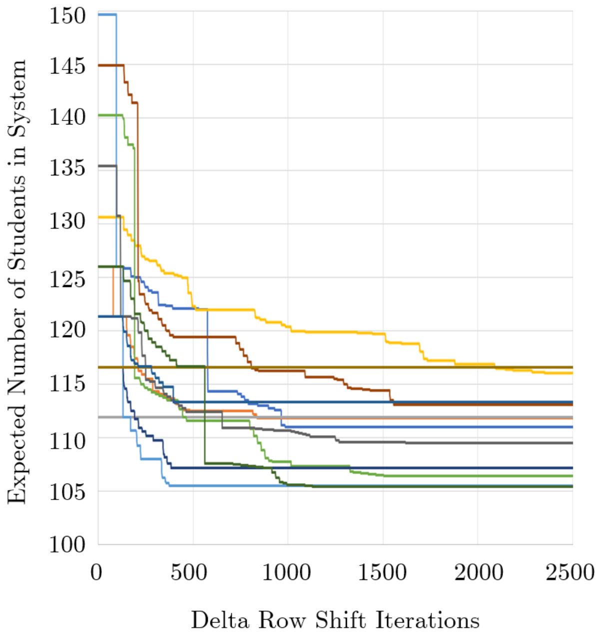
	10	20	30	40	50	60	70
2 Years	0.05	0.20	0.41	0.54	0.81	1.16	1.80
3 Years	0.08	0.27	0.71	1.08	1.65	2.28	3.22
4 Years	0.18	0.44	1.12	1.85	2.66	3.73	5.15
5 Years	0.34	0.69	1.71	2.87	4.44	7.14	9.30

6 RESULTS & DISCUSSION

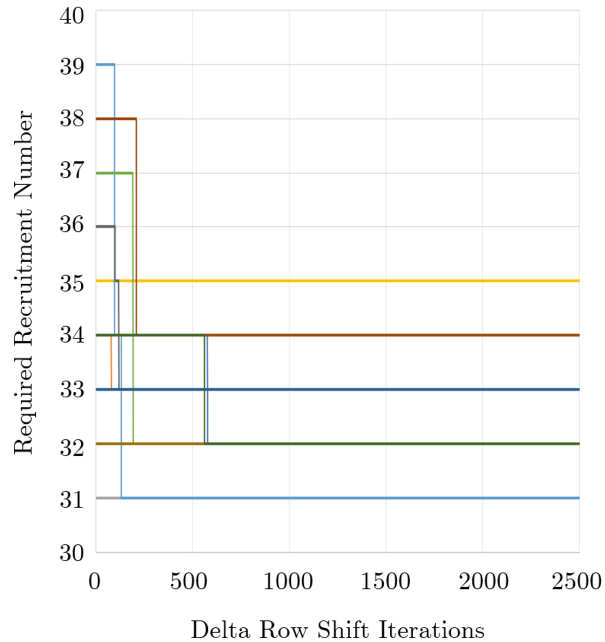
Figure 7a and 7b depict the total expected number of students entering each course and the minimum number of recruits required to meet capability for a given risk, respectively. These are shown for δ-matrix one-shifting iterations ranging from 0 to 2500. Each line represents a different random δ-matrix initialization.

6.1 Local Minima

From Figure 7a and 7b it is clear that both the expected number of students in the system (the cost) and recruitment numbers converge to local minima — different initializations leading to different minima. This is expected as the algorithm uses a greedy descent method. By running many trials with different δ-matrix initializations, different local minima are accessed and compared to give a close-to-optimal recruitment and



(a) The expected number of students of the system for increasing numbers of δ-matrix one-shifting iterations. The different colored lines represent different random initializations of the δ-matrices.



(b) The minimum recruitment required to meet a given risk for increasing numbers of δ-matrix one-shifting iterations. The different colored lines represent different random initializations of the δ-matrices.

Figure 7: Simulation results of the Delta-Cycling Algorithm for different random initializations.

form an upper bound on the global minimum. We expect that as the number of trials increases eventually the global minimum will be found, even if it cannot be determined that it has been achieved. Each trial takes a reasonably short time to converge as shown in Table 1, though, the trials take longer for larger recruitment sizes and more time periods. Importantly, this process is highly parallelizable as each trial can be generated from a random δ-matrix initialization seed. Inspection of the δ-matrices and using brute force on small cases, indicates that the δ-matrices have a regular pattern between student inputs and outputs which opens up future possibilities to explore this structure to more carefully find the functions ϕ explicitly, but further analysis will need to be conducted to confirm this.

6.2 Minimizing Recruitment vs. Minimizing Students in the System

Comparison of Figures 7a and 7b yields noticeable similarities. Firstly, both plots exhibit a convergent decreasing trend. The δ-matrix initializations that provide the least expected number of students at each course are also the initializations that provide the least entry recruitment number (light-blue line). The converse is also true as exhibited by the yellow and brown lines. Secondly, large decreases in cost occur in the same iterations as large decreases in entry recruitment number. These two similarities are to be expected, as decreases in the entry recruitment number have both a direct and indirect effect on cost as given in (1). It appears then that one-shifts in the δ-matrices that lead to a reduction in entry recruitment number are the ones that have the greatest impact in reducing cost. This, albeit phenomenological feature, might be used to improve the Delta Cycling Algorithm by targeting rows that drive reduction in recruitment first.

Interestingly however, the minimum recruitment number does not necessarily correspond to the minimum value of the cost function. For instance, the gray line in Figure 7a has a consistent cost of approximately 112, greater than other initializations. Yet it has the lowest (along with the light-blue line) recruitment number. The relationship between recruitment number and the expected number of students in the system is not always monotonic. That being said, the two values appear to correlate with one another to an extent.

7 CONCLUSION

We have described a stochastic programming model for the optimization of expected costs while minimizing risk of failure to achieve operational capability in the context of the Royal Australian Navy aircrew training continuum. The Delta-Cycling Method, for determining the minimum number of students to recruit for RAN training as well as how to assign those students amongst the different courses, has proved to be a reasonably efficient computational approach to solution of this optimization problem. Multiple trials of the algorithm can be performed quickly to obtain a reasonable estimate of the recruitment number. It has been found that the greedy Delta-Cycling Method causes rapid convergence of the expected number of students in the system to local minima. This, combined with random seeding, has been seen to yield improvements to what appear from simulations to be global minima. Other factors observed in simulations indicate the possibility of using initial recruitment, which correlates well, though not perfectly with the overall expected cost, as a ”first pass” cost to speed up computation, as well as bounds on course assignment numbers to reduce the search space. Finally patterns observed in the δ-matrices used for controlling student assignment provide a hint that alternative and less computationally expensive approaches will be achievable in future.

REFERENCES

Akinyele, S. T. 2007. “Determination of the Optimal Manpower Size using Linear Programming Model”. *Research Journal of Business Management* 1(1):30–36.

Bartholomew, D. J. 1971. “The Statistical Approach to Manpower Planning”. *The Statistician* 20(1):3–26.

Bertsekas, D. P. 2000. *Dynamic Programming and Optimal Control*. Belmont, USA: Athena Scientific.

Bertsekas, D. P., and S. E. Shreve. 1996. *Stochastic Optimal Control: The Discrete-Time Case*. Belmont, USA: Athena Scientific.

Birge, J. R., and F. Louveaux. 2011. *Introduction to Stochastic Programming*. New York, USA: Springer.

Charnes, A., and W. W. Cooper. 1963. “Deterministic Equivalents for Optimizing and Satisficing under Chance Constraints”. *Operations Research* 11(1):18–39.

Hill, B., D. Kirszenblat, B. Moran, and A. Novak. 2018. “Optimising Recruitment to Achieve Operational Capability Conditional on Appetite for Risk”. In *Proceedings of the 2018 Winter Simulation Conference*, edited by M. Rabe et al. Piscataway, New Jersey: IEEE.

Hoos, H. H., and T. Stützle. 2004. *Stochastic Local Search: Foundations and Applications*. San Francisco, USA: Morgan Kaufmann Publishers.

Kirszenblat, D., B. Hill, V. Mak-Hau, B. Moran, V. Nguyen, and A. Novak. 2017. “Using Column Generation to Solve an Aircrew Training Timetabling Problem”. In *22nd International Congress on Modelling and Simulation*, edited by G. Syme et al., 667–673. Hobart, Australia: Modelling and Simulation Society of Australia and New Zealand Inc.

Martel, A., and W. Price. 1981. “Stochastic Programming Applied to Human Resource Planning”. *The Journal of the Operational Research Society* 32(3):187–196.

Mehlmann, A. 1980. “An Approach to Optimal Recruitment and Transition Strategies for Manpower Systems using Dynamic Programming”. *Journal of the Operational Research Society* 31(11):1009–1015.

Mokhtari, H., and M. Dadgar. 2015. “Scheduling Optimization of a Stochastic Flexible Job-Shop System with Time-Varying Machine Failure Rate”. *Computers and Operations Research* 61(3):31–45.

Nguyen, V., B. Moran, T. Caelli, B. Hill, and A. Novak. 2018. “Dancing Links for Optimal Timetabling”. *Journal of Military Operations Research (to appear)*.

Spall, J. C. 2005. *Introduction to Stochastic Search and Optimization: Estimation, Simulation and Control*. New York, USA: John Wiley & Sons.

Suvorova, S., A. Novak, B. Moran, and T. Caelli. 2018. "Optimal Recruitment Strategies for Australian Naval Aviation Training using Linear Programming and Dynamic Programming Solutions - "The Price of Thinking Short Term"". *Journal of Military Operations Research (to appear)*.

Tavakkoli-Moghaddam, R., F. Jolai, F. Vaziri, P. K. Ahmed, and A. Azaron. 2005. "A Hybrid Method for Solving Stochastic Job Shop Scheduling Problems". *Applied Mathematics and Computation* 170(1):185–206.

Udom, A. U. 2013. "A Markov Decision Process Approach to Optimal Control of a Multi-level Hierarchical Manpower System". *CBN Journal of Applied Statistics* 4(2):31–49.

Vanderbei 1996. *Linear Programming: Foundations and Extensions*. Boston, USA: Kluwer Academic Publishers.

Wruck, S., I. F. A. Vis, and J. Boter. 2016. "Risk Control for Staff Planning in e-Commerce Warehouses". *International Journal of Production Research* 55(21):6453–6469.

AUTHOR BIOGRAPHIES

CAMERON PIKE is a Defence Research Scientist who joined the Defence Science and Technology Group in 2018. He received a B.Sc in engineering systems and M.Eng in mechanical engineering from the University of Melbourne. His interests include simulation, modeling and optimization. His e-mail address is Cameron.Pike@dst.defence.gov.au.

BILL MORAN A Professor in the Department of Electrical and Electronic Engineering at the University of Melbourne. Moran is a Fellow of the Australian Academy of Science. He has worked as a both as a Mathematician and as an Engineer across many research areas. Between 2011 and 2014 he was Director of the Defence Science Institute at the University of Melbourne. His current research interests lie in Optimization, Sensor Management, Statistical Signal Processing, Radar, Number Theory, and Harmonic Analysis. His e-mail address is wmoran@unimelb.edu.au.

DAVID KIRSZENBLAT is a mathematics Ph.D candidate at the University of Melbourne. He received a B.A/B.Sc (Honors) and M.Phil in mathematics from the University of Melbourne. His research interests include optimal control theory, geometric network optimization and the computational complexity and applications of integer linear programming. His e-mail address is d.kirszenblat@student.unimelb.edu.au.

BRENDAN HILL Brendan is a Researcher at the University of Melbourne specializing in Operations Research. Brendan holds a B.Sc (Mathematics and Statistics) from the University of Melbourne. He has accumulated 18 years experience in commercial software development across many industries. His research interests include optimization, simulation, machine learning and artificial intelligence. His e-mail address is brendan.hill@gmail.com.

ANA NOVAK Ana Novak joined DST Group, in 2010 as an Operations Research Scientist after four years as a specialist consultant in a boutique demand, inventory and supply chain optimization firm. She received her B.Eng. (Honors) in Information Technology and Telecommunications from University of Adelaide in 2003, followed by a Ph.D degree in Mathematics from The University of Melbourne in 2006 in Queueing Theory. Her research interests include planning, scheduling systems, simulation, modeling and optimization. Her e-mail address is Ana.Novak@dst.defence.gov.au.