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

CAPACITY ANALYSIS FOR AIRCREW TRAINING SCHOOLS - ESTIMATING OPTIMAL MANPOWER FLOWS UNDER TIME VARYING POLICY AND RESOURCE CONSTRAINTS

Pooia Lalbakhsh

School of Aerospace, Mechanical
and Manufacturing Engineering
RMIT University
Plenty Road, Bundoora
Vic 3083, AUSTRALIA

Vicky Mak-Hau

School of Information Technology
Deakin University
Waurn Ponds, Geelong
Vic 3216, AUSTRALIA

René Séguin

Center for Operational Research
and Analysis
Defence Research Development Canada
Ottawa, ON, CANADA

Vivian Nguyen
Ana Novak

Defence Science and Technology Group
Melbourne
Vic 3207, AUSTRALIA

ABSTRACT

Capacity analysis for systems with time varying constraints is still an open problem in Operations Research due to the non-stationarity of the problem domain. This is particularly true for Defence manpower supply which is subject to frequent temporal policy and resource changes. As such, the problem cannot be completely covered with a single overriding simulation or optimisation solution, but, rather, better described using piecewise interplay between simulation and optimisation. This paper describes such an approach for a flexible, interactive capacity analysis simulator with an embedded integer linear programming (ILP) optimiser.

1 INTRODUCTION

Capacity analysis for systems with time varying constraints remains a challenge in Operations Research. This is particularly relevant when dealing with Defence manpower supply which is subject to frequent temporal policy and resource changes. Past solutions of the problem typically resort to simulating training flows, and in recent years simulation combined with optimisation solutions have become popular (andrei Borshchev 2013). This paper is motivated by the need to optimally provide aircrew supply in one of the Australian Navy's training schools.

The Royal Australian Navy (RAN) has recently acquired a fleet of new aircraft. In order to achieve and maintain an adequate workforce level, the relevant training flows for the new aircrew has to be established within a given lead time. The training pipeline for the aircrew (pilots, observers and aircrewmen) finishes with conversion training, where students specialise in flying a particular aircraft type. Defence Science and Technology (DST) Group within the Australian Department of Defence was asked to determine the number of aircrew that could graduates in a conversion school, by scheduling the available resources: aircraft, simulators and instructors. To this end we have developed a simulation-optimisation approach for analysing the capacity of this training unit. It uses a Discrete Event Simulation (DES) approach to model the day-to-day operations of the training school, representing the training schedules, allocate aircraft, simulators and supplementary workforce resources. The DES also models constraints pertaining to resource availability including instructors' available hours, the training schools's operating hours and other business rules. An integer linear programming optimisation (ILP) algorithm is invoked by the simulation at every time slot to create step-wise optimal schedule to assign students, resources and instructors to classes.

Capacity analysis is a popular approach for analysing systems in education (Roberts-Gray et al. 2007), agriculture (Swidar et al. 2018), railway transportation and airports (Abril et al. 2008; Ignaccolo 2003; Lindfeldt 2015; Barrer et al. 2005; Özkan et al. 2016), Defence (Wiggins et al. 1992) and many more. These typically range from flow modelling using System Dynamics to multi-method simulation modeling and various types of programming (Linear Programming, Constraint Programming etc.) (andrei Borshchev 2013). The focus of this paper is on conducting capacity analysis using simulation-optimisation framework.

This paper is organised as follows. Section 2 describes the modelling structure, assumptions and constraints as well as model input and output. Section 3 describes a simulation environment architecture. The ILP solution for the optimisation of stepwise assignment of students to lessons is covered in Section 4. A sample scenario and results are presented in Section 5 and the paper is concluded in Section 6.

2 MODELING STRUCTURE

The training school in question uses a combination of live flying and synthetic training devices to provide a training program for Navy students. Each student, depending on their type (pilot, observer and aircrewman), has to complete a predefined set of lessons, each of which can vary in duration, instructor and physical resources requirements. Moreover, the lessons have prerequisite lessons that need to be satisfied in order to progress. The training school has a fixed number of available physical resources (aircraft and simulators) and instructors, who, depending on their specific qualifications, can teach some, but not all, lessons.

Our simulation-optimisation was developed in the AnyLogic™ software enhanced with custom Java classes and leveraging on the IBM CPLEX Optimiser for the optimisation module. The simulation replicates the operations of the training school, including dynamic allocations and releases of resources with time discretised to 15 minute intervals creating a sequence of smaller optimisation problems which are further solved by the ILP optimisation algorithm. The ILP optimisation algorithm finds an updated optimal solution among all combination of assignment possibilities, which, in turn, changes the system's overall state. Simultaneously, the model can highlight where bottlenecks might occur due to a lack of instructors or physical resources, providing an indication of potential problems in the system capacity. DES was selected as the preferred modeling approach to satisfy these objectives. Additionally, its ability to animate the modeled systems internal processes facilitated communication with the clients, to both validate the model and explain the potential issues and gaps between proposed and required resources for a given schedule.

2.1 Initialisation of the Model

The initial input into the model include: training syllabi for each student types, numbers of available human and physical resources as well as operational policies. Each syllabus, specified in the form of an array of variables, consisted of a sequence of lessons in the order of prerequisites. Each lesson has a duration and requirements for resources.

Resource capacity serves as additional input into the model, where the physical resources such as aircraft and different types of simulators have a window of hours during which they are available for use. Instructors have specified total daily working hours and maximum daily working spans. Furthermore, for instructors, only a portion of this time is allocated to teaching. Each instructor has a set of specialisations which determine which lesson they can instruct, availability profile as well as currency status. Currency is the amount of hours on specific resources, each instructor must meet every month. The daily working hour policy is valid for additional staff as well. Finally, the model uses student intake details such as intake dates of each cohort, type of students in each cohort and the number of students in each cohort.

2.2 Assumptions and Constraints

Following is a list of assumptions and constraints used during the modelling process.

- **Prerequisites** - All prerequisites must be met for each student/lesson combination.
- **Physical resource availability** - Aircraft and simulators are assumed to be operational throughout their availability periods.
- **Additional staff requirements** - Aircraft and simulator-based lessons have crew requirements in addition to instructors. These crew requirements are satisfied by taking non-instructional staff.
- **Instructor currency** - Specific types of instructors need to maintain minimum hours per month on flying tasks or simulators.
- **Student failure and unplanned leave** - These stochastic events are currently not included in the model. However, in this particular school, the effect of student failures on throughput is known from past experience to be minimal. Pre-planned leave is incorporated.
- **Weather** - Weather effects were not modeled at this stage, but is being considered for future work. Expert estimate is that weather constitutes a 4% drop in efficiency.

2.3 Model Output

The simulation model's output is mainly delivered to the user by time plots, charts, bar graphs, list boxes and heat charts. For a calculated number of graduates the following outputs are delivered to the user:

- Average number of graduates over simulated years.
- Average graduation time (in months) for each student type over simulated years.
- Percentage of daily utilisation of physical resources, instructors and additional staff.
- Average daily usage of physical resources, instructors and additional staff.
- A monthly report of the amount of currency (in hours) met by individual instructors.
- Lessons' buffer information, showing the cumulative students' waiting time for every lesson.
- Total number of organised sessions over simulated years.
- Total simulation run-time.

3 THE SIMULATION ENVIRONMENT ARCHITECTURE

3.1 Architecture of the Simulation

Our capacity analysis tool for aircrew training school is based on an integrated simulation-optimisation model. Agent based simulation modeling approach was considered for the development of the simulation module (Abar et al. 2017). The simulation is managed by the AnyLogic time engine starting from t_0 to t_{end}. Key entities such as *Student* agent were implemented using AnyLogic objects. AnyLogic objects support utilities such as state charts which makes implementing key objects easier, more reusable, faster and less error-prone. Other entities or processes which do not need such support were coded as pure Java classes. At the heart of the model, an ILP optimisation algorithm is implemented to solve the assignment problem at each time step. The algorithm is initialised with the current state's information during simulation and determines the optimal assignment of students and resources to the requested lessons. A change in the system state is a signal that triggers the optimisation algorithm to assign students in the waiting list. A change in the system state might be brought about by adding new students to the waiting list, releasing resources or starting a new working day. These situations are checked by a cyclic time-event triggered every 15 minutes to prevent premature assignment which is the situation when there are not enough students to efficiently use resources for their requested lessons. Figure 1 shows the conceptual structure of the implemented agents in AnyLogic. The *Admin* agent is the pivotal object in the simulation as it connects the elements of the functional triangle: the DES, the ILP optimisation algorithm and the resource manager. It collects information from all other agents and generates the required input to the ILP optimisation algorithm.

The *Main* agent initialises the model and creates the required data structures for other agents to feed the *Visualiser* agent.

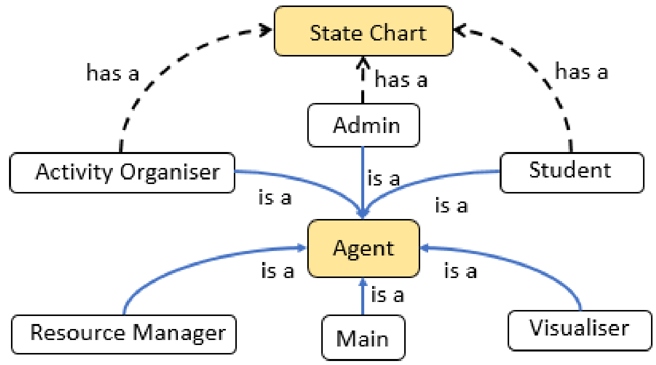

Figure 1: A directed agent graph showing dependencies of key agents implemented in AnyLogic.

Figure 2 shows the state chart of the *Admin* agent that connects the DES to the ILP algorithm. At every 15 minutes the state of the system is checked to determine the demand and possibility of organising new sessions. If the result of this check is positive, the *Admin* agent gathers the required information for the optimisation process. A current list of waiting students and available resources are the two fundamental data structures for the ILP algorithm. The *Admin* agent invokes the ILP optimisation algorithm developed in IBM CPLEX Optimiser. The result of the optimisation algorithm is then sent back to resource allocation which finally results in organising sessions for the requested lessons.

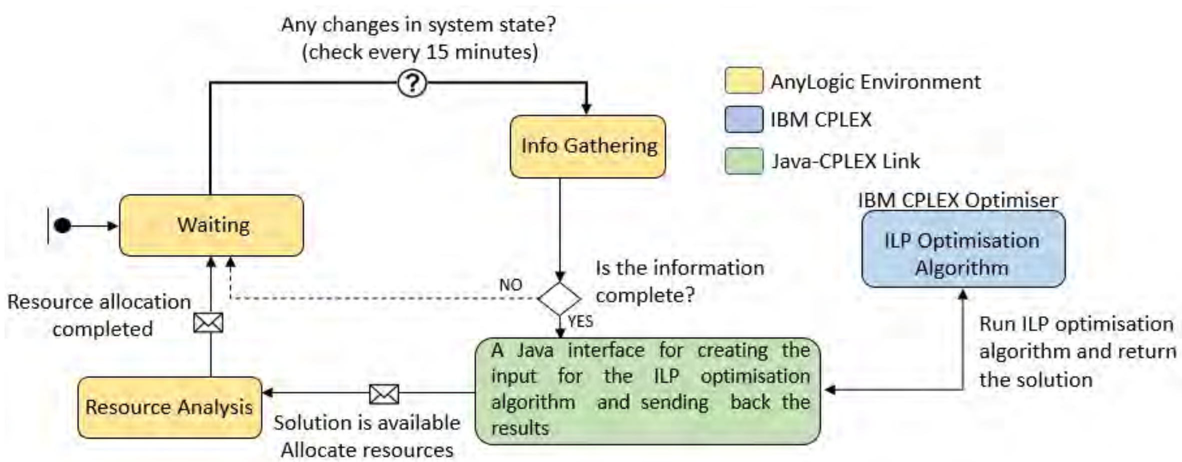

Figure 2: Interactions of the *Admin* agent, Java classes and the IBM CPLEX ILP Solver.

3.2 Formulation of Students' Graduation Priority Measure

Every student must have a value showing their graduation priority to enable the optimisation algorithm to direct its search towards the optimal solution. To have a complete and consistent weighing function over the simulation domain we used fuzzy logic to check, evaluate and propagate students' graduation priority referred to as *student's graduation priority measure*. This measure is the building block of the objective function used during optimisation.

The number of remaining lessons to graduation and the number of days to graduation are two parameters that were taken into account to calculate a student's graduation priority measure. Given $\mathscr{S}$={$s_1, s_2, ..., s_n$} a nonempty set of n waiting students, the graduation priority measure for student $s_j \in \mathscr{S}$ is computed as (1):

$$\sigma_{s_j} = g(N^{LG}_{s_j}, N^{DG}_{s_j}) \tag{1}$$

where g represents the fuzzy inference model, $N^{LG}_{s_j}$ is the number of lessons to graduation and $N^{DG}_{s_j}$ is the number of days to graduation. g is defined by 25 fuzzy rules and triangular membership functions for fuzzification (transforming scalar values into fuzzy values) of inputs and defuzzification (producing quantifiable results in crisp logic) of the output (Liang et al. 2013). The fuzzy rules are defined using the following format:

$$\textbf{IF } N^{LG}_{s_j} \text{ is } A_h \textbf{ AND } N^{DG}_{s_j} \text{ is } B_k \textbf{ THEN } v_{s_j} = F_r \quad \text{where } h,k,r \in \{1,2,3,4,5\} \tag{2}$$

A and B are sets of linguistic values referring to $N^{LG}_{s_j}$ and $N^{DG}_{s_j}$, respectively. A and B refer to the same set of values as indicated in (3):

$$\{\text{Very Small, Small, Medium, Large, Very Large}\} \tag{3}$$

F is a set of output linguistic values determining the graduation priority of students. F takes values from the set described in (4):

$$\{\text{Very Low, Low, Medium, High, Very High}\} \tag{4}$$

The fuzzy operator used for **AND** in (2) is multiplication. The method used for the defuzzification process is arithmetic mean of centers (Liang et al. 2013). Table 1 shows how input and output linguistic values are related to determine the priority of each student. This table implements a logic that strive to minimise the average graduate date by favouring students that are closer to graduation.

Table 1: Rules for fuzzy inference engine to compute students' graduation priority measure.

Days to graduation \ Lessons to graduation	Very Small	Small	Medium	Large	Very Large
Very Small	Very High	Very High	High	High	High
Small	Very High	Very High	High	High	High
Medium	Very High	High	Medium	Medium	Medium
Large	Very High	Medium	Medium	Low	Low
Very Large	Very High	Medium	Low	Low	Very Low

For every $s_j \in \mathscr{S}$, $N^{LG}_{s_j}$ and $N^{DG}_{s_j}$ were both normalised by (5) and (6) before the fuzzification process.

$$h(N_{s_j}^{LG}) = \frac{N_{s_j}^{LG} - min(N_{s_j}^{LG})}{max(N_{s_j}^{LG}) - min(N_{s_j}^{LG})} \tag{5}$$

$$h(N_{s_j}^{DG}) = \frac{N_{s_j}^{DG} - min(N_{s_j}^{DG})}{max(N_{s_j}^{DG}) - min(N_{s_j}^{DG})} \tag{6}$$

where $min(N_{s_j}^{LG})$ is the minimum number of remaining lessons to graduation over all students in $\mathscr{S}$, $min(N_{s_j}^{DG})$, $max(N_{s_j}^{LG})$ and $max(N_{s_j}^{DG})$ are defined similarly. Therefore, at every time slot the normalised input values must be re-calculated to represent current priority measures of students in the waiting list. The *Admin* agent is responsible for calculating and setting graduation priority measure for every student. More information about the fuzzy inference engine can be found in (Lalbakhsh et al. 2014).

4 STEPWISE STUDENT-LESSON ASSIGNMENT USING INTEGER LINEAR PROGRAMMING

In this section we present an integer programming model, that is called by the simulator each time students need to be allocated to lessons. The ILP optimally allocates students, resources, instructors and the required additional staff to lessons.

We define the following terms to be used by the ILP model.

- $\mathscr{L}$ is the set of all lessons, with $\mathscr{L}_c$ the set of all cohort lessons, and $\mathscr{L}_n = \mathscr{L} \setminus \mathscr{L}_c$ the set of all other lessons;
- $\mathscr{R}$ is the set of all resource types;
- $\mathscr{S}$ is the set of all students;
- $\mathscr{T}$ is the set of all instructors;
- σ_s is the graduation priority measure for student s;
- ρ_i is the currency score for instructor i;
- A is a $|\mathscr{L}| \times |\mathscr{S}|$ binary matrix with $a_{\ell,s} = 1$ indicating student s is eligible to do Lesson ℓ, and $a_{\ell,s} = 0$ otherwise;
- B is a $|\mathscr{L}| \times |\mathscr{T}|$ binary matrix with $b_{\ell,i} = 1$ indicating instructor i is eligible to teach Lesson ℓ, and $b_{\ell,i} = 0$ otherwise;
- N_ℓ is the class size limit of Lesson ℓ;
- m_ℓ is the number of instructors required in Lesson ℓ;
- $K_{\ell,r}$ is the number of resources of type r required in Lesson ℓ;
- Q_r is the number of available resources of type r;
- M is a sufficiently large number.

We also define the following decision variables:

- x_ℓ, for $\ell \in \mathscr{L}$ is a binary decision variable where $x_\ell = 1$ indicates lesson ℓ is run, and $x_\ell = 0$ otherwise;
- $y_{\ell,s}$, for $\ell \in \mathscr{L}$ and $s \in \mathscr{S}$, is a binary decision variable with $y_{\ell,s} = 1$ indicating Student s is assigned to do lesson ℓ and $y_{\ell,s} = 0$ otherwise;
- $z_{\ell,i}$, for $\ell \in \mathscr{L}$ and $i \in \mathscr{T}$, is a binary decision variable where $z_{\ell,i} = 1$ indicates instructor i will teach lesson ℓ and $z_{\ell,i} = 0$ otherwise.

The objective (7) below is a weighted function of student graduation priority measure and instructor currency score, where a student's graduation priority measure is obtained from the fuzzy inference in subsection 3.2. The objective of the ILP optimisation algorithm is to maximize this objective function. As indicated in (7) the objective value is computed using the summation of students' priority measure and instructors' currency score, for all waiting students and available instructors, respectively. Due to the large value of M in (7), the impact of students' priority measure is much bigger that the instructors' currency score. In our simulation model, where more than one instructor is available to be assigned to a lesson, the instructor with the lowest currency score is selected for the lesson. The following are the constraints that define the ILP students-lesson assignment problem.

Constraint (8) is the Inexact Cover Constraint which ensures that a student will do no more than one lesson that he/she is eligible to do. Constraint (9) ensures that a student can only do a lesson if the lesson is run and that he/she is eligible to do the lesson. Constraint (10) makes sure that that if a lesson is run, then there must be at least one student taking it. Constraint (11) ensures that the number of students in a lesson does not exceed the size of the class. Constraint (12) makes sure that the resources required will not exceed the number of resources available. Constraint (13) ensures that if a cohort class is run, then all students in the cohort class must do the lesson. Constraint (14) makes sure that each instructor will not be allocated to more than one lesson at a time. Constraint (15) ensures that if a lesson is not run, then no instructors will be allocated to it. Constraint (16) makes sure that if a lesson is run, then the right number of instructors will be allocated to it.

$$\max z = M \sum_{\ell\in\mathscr{L}} \sum_{s\in\mathscr{S}} \sigma_s \, y_{\ell,s} + \sum_{\ell\in\mathscr{L}} \sum_{i\in\mathscr{T}} \rho_i \, z_{\ell,i} \tag{7}$$

$$\sum_{\ell\in\mathscr{L}} a_{\ell,s} \, y_{\ell,s} \le 1, \qquad \forall s \in \mathscr{S} \tag{8}$$

$$y_{\ell,s} \le a_{\ell,s} \, x_\ell, \qquad \forall \ell \in \mathscr{L},\ s \in \mathscr{S} \tag{9}$$

$$x_\ell \le \sum_{s\in\mathscr{S}} a_{\ell,s} \, y_{\ell,s}, \qquad \ell \in \mathscr{L} \tag{10}$$

$$\sum_{s\in\mathscr{S}} y_{\ell,s} \le N_\ell \, x_\ell, \qquad \ell \in \mathscr{L} \tag{11}$$

$$\sum_{\ell\in\mathscr{L}_c} K_{\ell,r} \, x_\ell + \sum_{\ell\in\mathscr{L}_n} \sum_{s\in\mathscr{S}} K_{\ell,r} \, y_{\ell,s} \le Q_r, \qquad r \in \mathscr{R} \tag{12}$$

$$\sum_{s\in\mathscr{S}} a_{\ell,s} \, y_{\ell,s} = \sum_{s\in\mathscr{S}} a_{\ell,s} \, x_\ell, \qquad \ell \in \mathscr{L}_c \tag{13}$$

$$\sum_{\ell\in\mathscr{L}} b_{\ell,i} \, z_{\ell,i} \le 1, \qquad \forall i \in \mathscr{T} \tag{14}$$

$$z_{\ell,i} \le b_{\ell,i} \, x_l, \qquad \forall \ell \in \mathscr{L},\ i \in \mathscr{T} \tag{15}$$

$$\sum_{i\in\mathscr{T}} z_{\ell,i} = m_\ell \, x_\ell, \qquad \forall \ell \in \mathscr{L} \tag{16}$$

5 SIMULATION SCENARIO AND RESULTS

5.1 Scenario

In this section a sample scenario is explained and simulated. This simulation scenario was run on a Dell Precision Tower 3420 with an Intel Core i7-6700 CPU and 32 GB of RAM. We used the AnyLogic™ 8 Professional simulation software as the container of the whole project. Java classes and interfaces were implemented in NetBeans 8.0.2 using JDK 1.8. The JFuzzyLogic library was used to define fuzzy rules and fuzzy membership functions (Cingolani and AlcalÁ-Fdez 2013) and the IBM CPLEX ILOG Optimiser Java library was used to implement and run the ILP optimisation algorithm.

The simulation for the sample scenario starts from 1 Jan. 2017 to 1 Jan. 2022. As mentioned in subsection 2.2, no stochastic events were considered in this version of the model, so the simulation was run only once. Since the major objective of the experiment is capacity analysis, we will focus on resource usage and bottleneck analysis rather than the optimisation algorithm itself. The scenario has two different types of students e.g. *Student Type 1* and *Student Type 2* with two different syllabi and also different expectation durations e.g twelve months for Type 1 and six months for Type 2. Fourteen students are injected into the simulation model each year. Commencement dates for both student types are shown in Table 2 from 2017 to 2022. Sessions can generally be organised on weekdays from 8am to midnight throughout the year. No sessions can be organised on weekends and public holidays according to the Australian calendar. Six different types of physical resources were considered, including two aircraft and five different types of simulators. The simulation scenario also involves three different types of additional staff. The numbers and the availability of the physical resources and additional staff are presented in Table 3. Twelve instructors from four different instructor types were also involved in the simulation scenario with the availability ratio of 60% and 40 hours of weekly work load. Instructors need to have 10 hours currency per month, except Type 4 instructors. Table 4 shows the type and specialisation of every instructor.

Table 2: Student yearly intakes for the simulation scenario (2017-2022).

Intake date	Student type	Number of intakes
20 Jan (6:00 am)	Student Type 1	4
23 Jan (6:00 am)	Student Type 2	3
5 June (6:00 am)	Student Type 1	4
5 June (6:00 am)	Student Type 2	3

Table 3: Availability of physical resources and additional staff for the simulation scenario.

Physical resource				Additional staff		
Resource type	Count	From (hrs)	To (hrs)	Staff type	Count	Daily availability (hrs)
Aircraft	2	10:00	23:00	Staff Type 1	5	8.5
Simulator Type 1	1	0900	23:00	Staff Type 2	5	8.5
Simulator Type 2	1	0900	23:00	Staff Type 3	5	8.5
Simulator Type 3	5	0900	23:00			
Simulator Type 4	5	0900	23:00			
Simulator Type 5	1	0900	23:00			

Table 4: Intructors' type and specialisation considered for the simulation scenario.

Instructor label	Type	Specialisation	Instructor label	Type	Specialisation
Instructor 1	Type 1	A, B	Instructor 7	Type 2	B, D
Instructor 2	Type 1	B	Instructor 8	Type 2	A, B, C
Instructor 3	Type 1	A,B	Instructor 9	Type 3	—
Instructor 4	Type 1	—	Instructor 10	Type 3	—
Instructor 5	Type 1	—	Instructor 11	Type 3	—
Instructor 6	Type 2	B,C,D	Instructor 12	Type 4	—

5.2 Simulation Results

The complete simulation runtime for the scenario described in subsection 5.1 was 21 minutes. For the five years simulation scenario 20,018 sessions were organised. Figure 3 shows the user interface designed in AnyLogic to display the students' graduation information. The status of each student is shown with rectangles placed on the student information list box. Red rectangles represent delayed graduation, blue rectangles show on-time graduation, green rectangles show early graduation and finally white rectangles refer to the students who have not graduated yet. As shown in Figure 3, 32 out of 40 students of Type 1 and 18 out of 24 students of Type 2 graduated. The minimum and maximum graduation time for Type 1 students are 10 and 13, respectively. For Type 2 students, these figures are 8 and 13. Average graduation time for Type 1 students is 11.75 and for Type 2 students is 10 months.

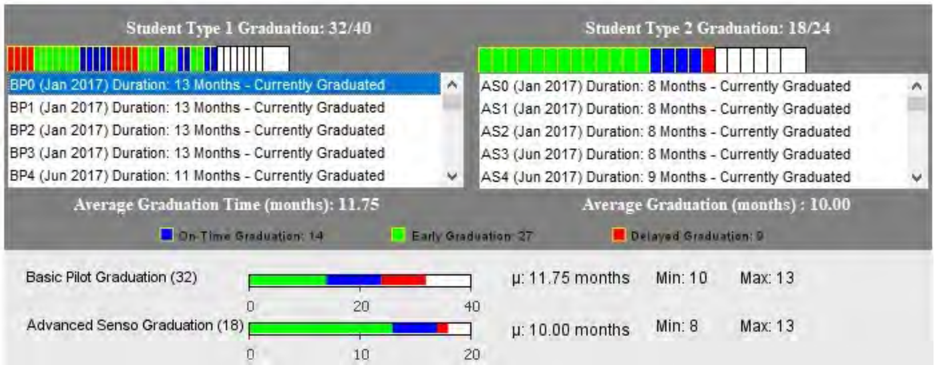

Figure 3: The user interface developed in AnyLogic for showing students' graduation information.

Utilisation of physical resources, additional staff and instructors are shown by *Utilisation* and *Frequency* time plots. Figure 4 shows two examples of these plots generated for *Aircraft* and *Simulator Type 2*. The *Utilisation* time plot shows the percentage of daily resource utilisation while the *Frequency* time plot shows average number of daily successful resource allocation tasks over all individual resources of the same type. Table 5 shows detailed utilisation percentages for physical resources, instructors and additional staff.

Instructors' currency information is delivered through bar graphs shown in Figure 5. As shown in this figure, for the sample scenario, almost all instructors met their 10 hour monthly currency except one instructor type 3 which is not strictly getting 10 hours.

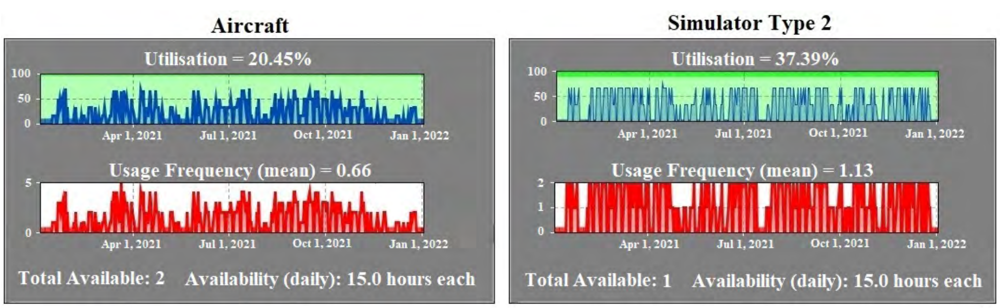

Figure 4: Utilisation and Frequency time plots generated by the *Visualisation* agent in AnyLogic .

Finally, the heat chart shown in Figure 6 shows the situation of buffers in terms of cumulative waiting hours. Buffers can be considered as waiting queues for each lesson. Four color spectrum were defined to show how busy buffers were for each lesson: yellow spectrum for lessons with the shortest waiting time, then green, then blue, and finally the red spectrum for lessons with the longest waiting time. The heat chart is a valuable feature to detect potential bottlenecks. Figure 7 shows a plot for normalised cumulative waiting time presented by the heat chart for the sample scenario for all lessons. We considered lessons with normalised buffer values larger than 0.75 as potential bottlenecks which are shown in red in Figure 7. For the sample scenario, the 13 detected potential bottlenecks are: $\{l_{44}, l_{51}, l_{56}, l_{77}, l_{85}, l_{91}, l_{97}, l_{106}, l_{193}, l_{331}, l_{406}, l_{507}, l_{529}\}$. They are all in the red spectrum. Nine lessons $\{l_{44}, l_{51}, l_{56}, l_{77}, l_{85}, l_{91}, l_{97}, l_{106}, l_{331}\}$ need *Simulator Type 2*, showing that the availability of this resource is too low for organising the required numbers of sessions for waiting students. Similarly, lessons $\{l_{193}, l_{507}, l_{529}\}$ need an *Aircraft*, showing that the number of aircraft may be too low to support the students' progress.

Table 5: Utilisation statistics for physical resources, instructors and additional staff

Resource type	Util.%	Instructor type	Util.%	Staff type	Util.%
Aircraft	20.45%	Instructor Type 1	19.42%	Staff Type 1	23.32%
Simulator Type 1	32.27%	Instructor Type 2	18.67%	Staff Type 2	24.07%
Simulator Type 2	37.39%	Instructor Type 3	18.46%	Staff Type 3	27.90%
Simulator Type 3	0.46%	Instructor Type 4	1.49%		
Simulator Type 4	18.10%				
Simulator Type 5	5.91%				

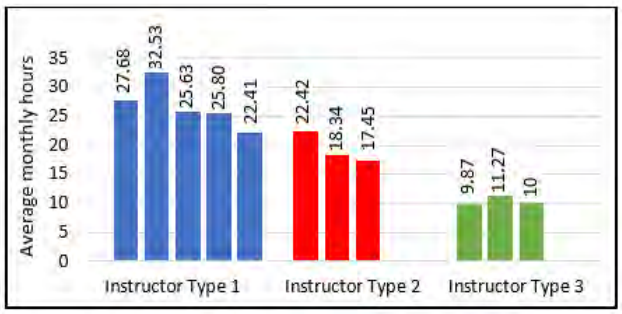

Figure 5: Monthly currency in hours met by instructors (*Type 4* instructors are not included since they do not have currency).

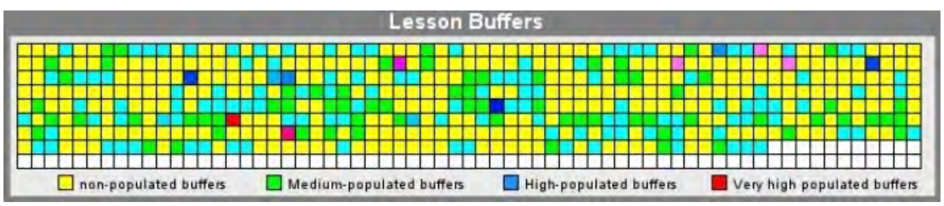

Figure 6: The heat chart showing the situation of buffers for every lesson using four colour spectrum.

6 CONCLUSIONS

Capacity analysis of a complex non-stationary dynamic system with constraining rules, different types of students and resources is a challenging task. This paper presents an integrated simulation-optimisation solution that forceasts the number of annual graduated aircrew, for a given set of availble physical and human resources, constrained by operating policies. Furthermore, the model with its intuitive interface, provides users with ready access to detailed information on the state of each system, for example, resource bottlenecks.

The integrated simulation-optimiser capitalises on the use of AnyLogic to provide the simulation engine with a flexible inter-dependent agent architecture, while the ILP optimiser makes the best possible decisions on allocation of instructors and students to lesson at every time slot. In this work computing the optimal states dynamically, when global optimisation is not possible, has proved to be a useful strategy for capacity analysis over other sub optimal methods. Finally, fuzzy logic is utilised to derive dynamic weightings of student priorities, in a consistent fashion. Heat maps provide a visualistion of lesson buffers and readily available statistics on resource utilisation is displayed on the dashboard. Taken together, this system provides a robust decision making framework for Defence planners.

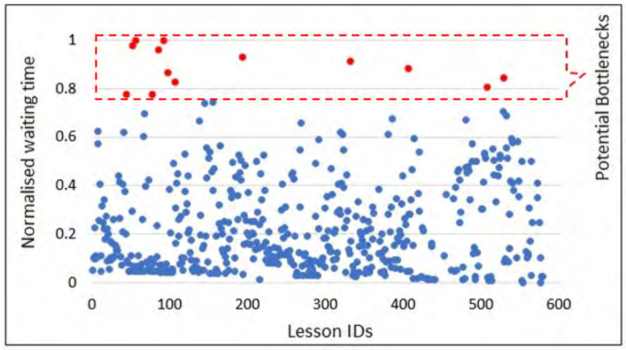

Figure 7: Normalised cumulative waiting hours for all buffers for the total simulation time.

ACKNOWLEDGMENTS

The authors would like to acknowledge Professor Terry Caelli for his insightful comments and editorial feedback.

REFERENCES

Abar, S., G. K. Theodoropoulos, P. Lemarinier, and G. M. OHare. 2017. "Agent Based Modelling and Simulation Tools: A Review of The State-of-Art Software". *Computer Science Review* 24:13–33.

Abril, M., F. Barber, L. Ingolotti, M. A. Salido, P. Tomos, and A. Lova. 2008. "An Assessment of Railway Capacity". *Transportation Research Part E: Logistics and Transportation Review* 44:774–806.

andrei Borshchev 2013. *THe BIg BOok of SImulation MOdeling: MUltimethod MOdeling With ANylogic 6*. 1st ed. Anylogic North America.

Barrer, J., P. Kuzminski, and W. Swedish. 2005. "Analyzing The Runway Capacity of Complex Airports". In *Aiaa 5th Atio and 16th Lighter-Than-Air Sys Tech. and Balloon Systems Conferences*, 7354–7361.

Cingolani, P., and J. AlcalÁ-Fdez. 2013. "Jfuzzylogic: A Java Library To Design Fuzzy Logic Controllers According To The Standard For Fuzzy Control Programming". *International Journal of Computational Intelligence Systems*:61–75.

Ignaccolo, M. 2003. "A Simulation Model For Airport Capacity and Delay Analysis". *Transportation Planning and Technology* 26:135–170.

Lalbakhsh, P., B. Zaeri, and Y. P. Chen. 2014. "Using Dead Ants To Improve The Robustness and Adaptability of Antnet Routing Algorithm". *Journal of Network and Computer Applications* 44:196–211.

Liang, D., D. Liu, W. Pedrycz, and P. Hu. 2013. "Triangular Fuzzy Decision-Theoretic Rough Sets". *International Journal of Approximation Reasoning* 54:1087–1106.

Lindfeldt 2015. *Railway Capacity Analysis: Methods For Simulation and Evaluation of Timetables, Delays and Infrastructure*. Ph.D. thesis, Department of Transport Science, KTH Royal Institute of Technology, Stockholm.

Özkan, E. D., S. Nas, and N. GÜler. 2016. "Capacity Analysis of Ro-Ro Terminals By Using Simulation Modeling Method". *The Asian Journal of Shipping and Logistics* 32:139–147.

Roberts-Gray, C., P. M. Gingiss, and M. Boerm. 2007. "Evaluating School Capacity To Implement New Programs". *Evaluation and Program Planning* 30:247–257.

Swidar, M., S. Szewranski, and J. K. Kazak. 2018. "Foodshed As An Example of Preliminary Research For Conducting Environmental Carrying Capacity Analysis". *Sustainability* 10:882–904.

Wiggins, V. L., S. K. Engquist, and L. T. Looper. 1992. "Applying Neural Networks To Air Force Personnel Analysis". Technical Report No. 2, Armstrong Laboratory, Human Resources Directorate, Bryan, Texas.

AUTHOR BIOGRAPHIES

POOIA LALBAKHSH is a Research Assistant at RMIT University. He holds a PhD in Computer Science and Computer Engineering from La Trobe University, Melbourne, Australia. Pooia works on decision support systems, swarm intelligence, optimisation algorithms, simulation modelling, data analytics and data mining. His email address is pooia.lalbakhsh@rmit.edu.au.

VICKY MAK-HAU is a senior lecturer at Deakin University. She holds a PhD in Mathematics from The University of Melbourne. She was one of the Chief Investigators in two Australian Research Council Discovery Projects, and is experienced in modelling and solving real-life decision making and optimisation problems such as: Aircraft Rotation, Vehicle Routing, Machine Scheduling, Radiotherapy Treatment Planning, Kidney Matching Optimisation, Day Oncology Ward Patient Scheduling, Hospital Ward Layout Planning, Scheduling, and Workload Allocation Optimisation. Her email address is vicky.mak@deakin.edu.au.

VIVIAN NGUYEN is a Research Scientist within the Defence Science and Technology (DST) Group. Vivian Nguyen holds a Bachelor of Information Technology (Software Engineering) (Hons) from the University of South Australia. She has an extensive background in software engineering and research in the areas of computer-supported collaborative work and smart-room environments, as well as modelling and simulation to support organizational planning and design. Her current research interests include novel algorithms for complex scheduling problems; agent based modelling and multi-paradigm modelling approaches. Her email address is vivian.nguyen@dst.defence.gov.au.

RENÉ SÉGUIN holds a Ph.D. in Operations Research from the University of Montréal. Since 2002, he has been a Defence Scientist in the Centre for Operational Research and Analysis of Defence Research and Development Canada. He is currently Director Air Staff Operational Research and is working on various readiness, force development and force generation issues for the Royal Canadian Air Force. He is an Area Editor of INFOR. His current research interests are resource allocation, scheduling and simulation. His email address is rene.seguin@drdc-rddc.gc.ca.

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 optimisation firm. After immigrating to Australia in 2000, she received her B.Eng. (Honours) 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 and scheduling systems, simulation and modelling, optimisation and modelling uncertainty. Her email address is ana.novak@dst.defence.gov.au.