IMPROVED SHIFT GENERATION FOR NEW ZEALAND CUSTOMS

David Geary

Engineering Science Department

Engineering School

Auckland University

New Zealand

1998

dgea001@esv1.auckland.ac.nz

Abstract

For many years, the University of Auckland has been involved with the staffing problems faced by NZ Customs at Auckland International Airport. To date, the shift generation phase of this work has been implemented using heuristics that attempt to capture many of the non-linearities associated with Customs quality measure. This project develops improved linear programming based tools to generate these shifts, and shows that this approach leads to improved quality rosters for Customs staff.

1 Introduction

Customs has a unique staff rostering problem. The staff demand at any one time is determined by the number and payload of aircraft arriving and departing throughout the day. This leads to an erratic schedule of staff requirements, and hence an interesting problem of staff rostering. Because we are dealing with so many staff, it would be grossly inefficient and expensive to haphazardly allocate staff rosters. It would also be far too time consuming and inaccurate to manually create each roster for each day. By preparing an automatic roster optimisation program, shifts can easily be produced and altered as the need arises.

The main objective of such an optimisation program is to minimise any surplus staff, whilst meeting minimum expected requirements. "Surplus staff" refers to excess allocated staff over and above the expected demand for any period in the roster.

One of the characteristic attributes of the Customs rostering problem is that there are multiple skill levels of staff. The more trained employees can do a wider range of jobs than the new or temporary staff. For example, work at the departure gates is a relatively unskilled job, and can be done by new employees. Work at the arrivals gates requires more training, and is done by more skilled workers. These trained workers can also do departures work as required.

As a result of the complexities of the problem, the task of optimising Customs staff rosters was given to the Auckland University Engineering Science Department. The first version was produced some years ago. It met the client's needs of a 'black-box' approach, and successfully minimised the financial costs of employing staff for the company. Since then, however, the client has requested a second version, which the author is currently working on. This 'improved' version is required to not only minimise financial cost to the company, but also place a certain emphasis on creating 'people friendly' rosters. These 'people friendly' rosters would allow staff to work longer, and more consistent shifts.

2 A-Matrix Construction

The project discussed in this paper works around a Set-Partitioning optimisation program (SETPAR) written by Dr. David Ryan. The first task of the project was to be able to create a right-hand-side vector (RHS) and an A-matrix that are compatible with the SETPAR program. The RHS vector is read in from a downloaded staff requirement file from Customs, and becomes the first string of data for SETPAR to read. The A-matrix is then constructed by listing every possible feasible shift for every available staff member. These shifts can vary in length, and start-time. Depending on the wishes of the client, staff can begin shifts on the quarter-hour, half-hour, on the hour, or any other product of a quarter hour. Lengths of shifts can also be controlled in a similar fashion. These constraints on possible shifts greatly reduce the number of columns in the A-matrix, hence dramatically reducing computer processing time.

By the time the file is ready for SETPAR to read, the A-matrix will contain every possible shift for every single available staff member within a set of constraints chosen by the user. The A-matrix is stored in a file as a series of rows, where each row represents a column of the A-matrix. One of the first values of each row stores the cost of the respective column / shift, and the following values represent the positions of the ones in that column. Thus all the possible shifts are stored with their corresponding costs.

3 Adjusting Costs To Improve Staff Contentedness

The overall goal of this project is to be able to create staff rosters that not only minimise straight financial cost to the company, but that also suit the needs of the staff as well as possible. There is an ongoing list of alterations and cost adjustments that could be made to roster optimisation programs to keep staff satisfied. Some of these may be minor and trivial, but others may have major influence on the staff involved.

3.1 Rewarding Longer Shifts

The most trivial of those looked at so far is to reduce relative costs for longer shifts. This is because staff prefer longer shifts where they can maximise their earnings for each trip they need to make out to the airport.

The trade-off of increasing shift length is increased cost to the employer in the form of increased paid hours. By adjusting a continuous input parameter, the user can choose between shorter shifts, or longer and possibly more expensive shifts.

3.2 Incentive for Similar Start Times for One Day

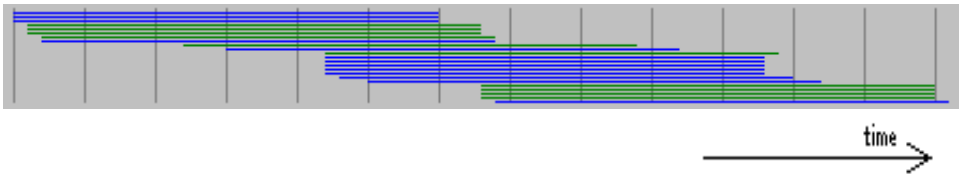

Figure 1. An initial optimisation gives a roster of staff starting at ten times during the day.

The next improvement to be made was to have staff starting their shifts at similar times. This is to avoid staff constantly trickling in at all hours of the day, making it hard to keep a check on who is on time, and who is late for work. The department prefers to have as many staff starting their shifts at the same time as possible. The method by which this is achieved is to optimise a day's roster twice. The first run is done with normal cost adjustments. This will produce an output file which contains the optimised roster so far (see Figure 1 – each horizontal line represents the start time, duration, and end time of each staff member employed for that day). For the second run, the A-matrix is constructed again, where shifts with similar start times to those in the SETPAR output file (created from the first run) have their costs lowered. If there are a large number of shifts in the first roster with a particular start time, then any shift in the second A-matrix with that start time will have a significant reduced cost. For example, if the first run indicates that ten staff should all start at 9:45 am, then the second A-matrix construction will reduce the costs of all shifts starting at 9:45 am by a factor to the power of ten. This gives an incentive for other staff to also start their shifts at 9:45 am. The effect of this can be seen in Figure 2, where there are only seven start times, as opposed to ten for the same problem solved in Figure 1.

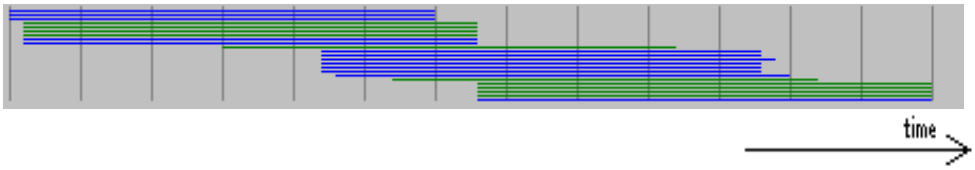

Figure 2. A second run of the program with adjusted costs giving seven starting times.

Again, there is a trade-off between increased staff satisfaction, and cost to the employer. There is a parameter that the user can adjust to add weight to the importance of similar start times, which is quite likely to increase total paid hours, and hence total cost of the solution.

3.3 Incentive for Similar Start Times From Day to Day

The next step after giving staff incentive to cluster their start times for one day, is to give incentive to start at an approximately similar time as the previous, or following day. The idea behind this is to give staff a more consistent roster for the week. The theory for a one day comparison is similar to the theory discussed in 3.2. In this case a reduced cost is applied in the A-matrix for shifts that start at similar times as those from the previous (or following) day's roster.

From this stage, it is a relatively simple, yet expensive task to optimise the whole week's roster (or however long the entire roster may be) such that each staff member's start time for each day is relatively consistent. This is done by doing a comparison first

between the first two days, then one of the first three days, then one of the 2nd, 3rd, and 4th days, and so on until the final two days. This whole process is then repeated as many times as required to 'smooth' the solution.

4 Working With Various Levels Of Staff Ability

An interesting aspect to this type of staff rostering is that different staff have had different levels of training and experience. Hence they are divided into varying levels of ability. For example, trained staff may be able to do arrivals or departures, but less-trained staff may only be able to do departures. Hence less-trained staff are set to do departures, and in the case of a shortage may be covered by surplus trained staff, who are primarily set to do arrivals. The theory is discussed in greater detail in M.Smith's Thesis. The model is of the form:

- number of trained staff must be greater than or equal to the required staff for arrivals
- number of trained staff plus number of less-trained staff must be greater than or equal to the total required staff for arrivals and departures.

This creates an interesting A-matrix structure. For an A-matrix that would normally look like:

period 1	0	1	1
period 2	0	0	1

where each column represents a staff member's possible shift; either not working, working first period only, or working both periods.

We now have a 'multi-tasking' A-matrix:

		trained		less-trained		
period 1	0	1	1	1	1	departures
period 1	0	1	1	0	0	departures + arrivals
period 2	0	0	1	0	1	departures
period 2	0	0	1	0	0	departures + arrivals

Figure 3. A-matrix structure for multi-tasking model.

where the first column represents the staff not member working. The second partition, which consists of two columns, represents a trained staff member's submatrix. This submatrix has all the possible shifts for a trained staff. As can be seen in the submatrix, these staff can do 'departures' and 'departures or arrivals'. That is to say that an increase of one trained staff member will increase (by one) the staff available for both the arrivals & arrivals + departures constraints. The second partition corresponds to a less-trained staff submatrix. This submatrix illustrates that less-trained staff can only contribute to the departures demand.

Note that 'less-trained staff' are not trained to do arrivals.

4.1 A Problem Arises With the Multi-tasking Model

After some initial study and experimentation with this model, the author found a problem with it. This problem arises when the staff demand is not met by the roster.

Although this may often not be allowed to be the case, it is a possibility that must be examined. As described above, the model used is:

no. of trained staff	>=	arrivals demand
no. of trained staff + no. of less-trained staff	>=	arrivals + departures demand

The problem arises when arrivals is understaffed. According to this model, we can then have more less-trained staff than required for departures alone (which is all that less-trained staff can actually do). This means there is a surplus of less-trained staff that can not work in that period, but could have been rostered for a different period where they could have been useful. Because the above model does not register this case as an 'overcover' (more staff than necessary), it does not allocate an overcover cost, and sees this period's roster as filling in a required demand. An example roster which displays this problem is shown in Figure 4, where available staff is a limiting factor. In this example, a very large 'overcover cost' was assigned, which effectively means there should be no surplus staff at any time.

Figure 4 is an example output from the author's staff rostering program. The hollow vertical bars represent the required staff at any one period of time during the day. The solid bars within these demand requirements represent the staff allocated by the program to be working at each of those respective periods. Slacks and surpluses are easily recognised as hollow, unfilled sections, or solid bars sticking out from the histogram respectively.

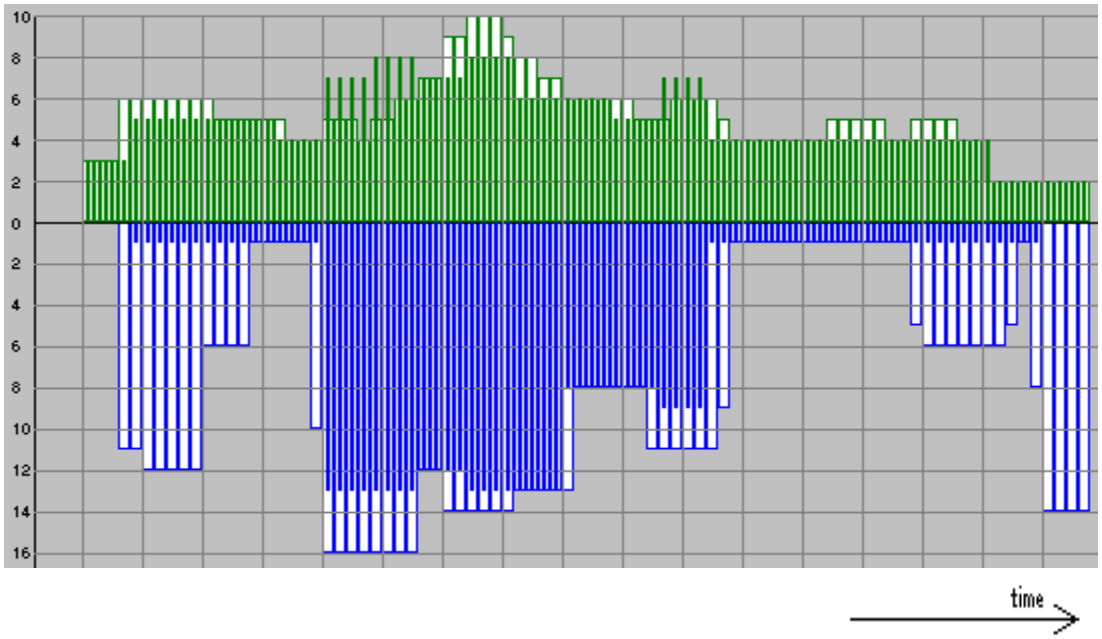

Figure 4. Example of multi-tasking model showing problems when understaffing occurs.

It is apparent in this example that during two sections of the roster the number of less-trained staff (bars above axis) exceed the departures demand (histogram above axis). This case highlights a problem with the model, as one of the parameters in this example stressed that no surplus staff should be used. The situation, however, is not recognised by the model as a surplus of staff, as the extra less-trained staff are helping satisfy the constraint: "no. of trained staff + no. of less-trained staff >= arrivals + departures demand".

To remedy this problem, the author experimented with an alternative model:

no. of less-trained staff >= departures demand
no. of trained staff + no. of less-trained staff >= arrivals + departures demand

This alternative model overcomes the above-mentioned problem. If there is a shortage of staff for a period, less-trained staff will *not* be allocated to cover for a shortage in staff for arrivals. The example understaffed scenario shown in Figure 4 is optimised by the alternative model, and the resulting roster is shown in Figure 5, again with large overcover costs which effectively disallow any surplus staff.

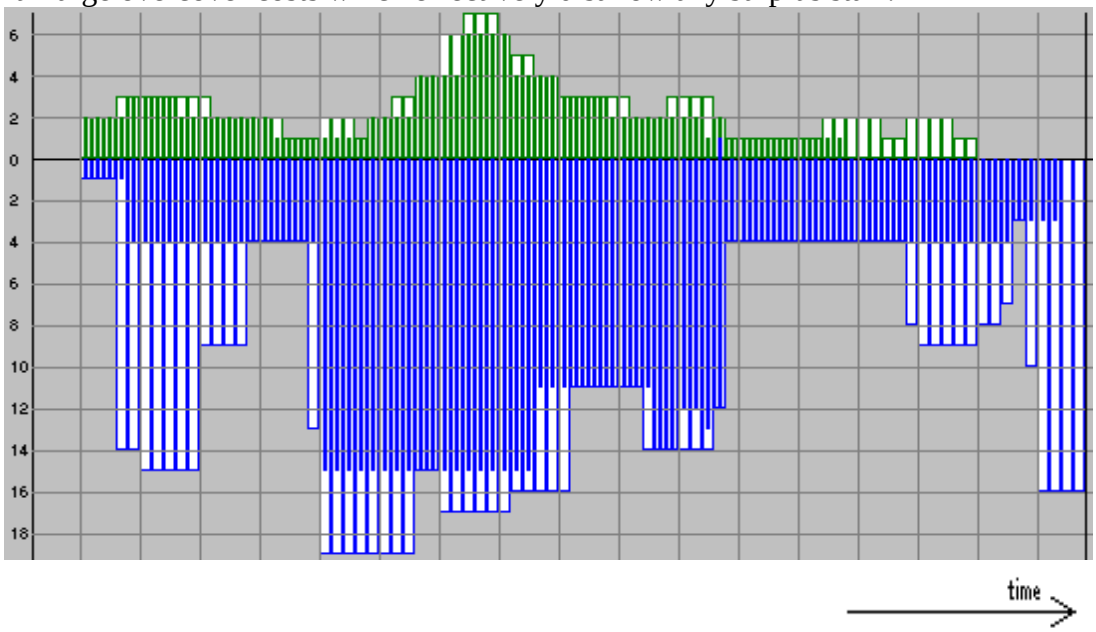

Figure 5. The alternative model handles this understaffed situation slightly better.

This, however, does not solve our problem, but merely creates a new one. The alternative model breaks down in some when staff requirements are met or exceeded. This occurs because less-trained staff are now allowed to fill up both constraints; for departures, and for the total of arrivals & departures. This problem is illustrated in Figure 6.

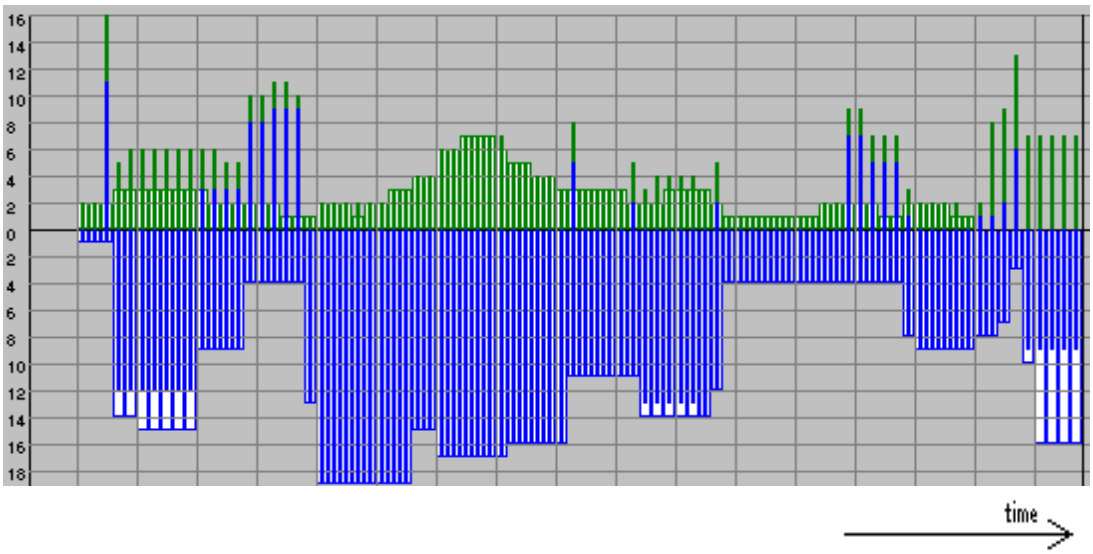

Figure 6. Example of alternative model showing problems when overstaffing occurs.

Figure 6 is the roster created by the alternative model where the parameters are set such that the staff roster meets or exceeds demand at all times. It is clear that there is a problem with this model as not all demand of arrivals is being met by A-staff. The alternative model 'thinks' it is filling in these gaps with surplus less-trained staff, when in actual fact less-trained staff can not do arrivals. Hence this alternative model is *not* ideal for situations where demand in all periods is required to be met or exceeded.

As expected, the original multi-tasking model was found to handle such an overstaffing scenario quite satisfactorily (see Figure 7).

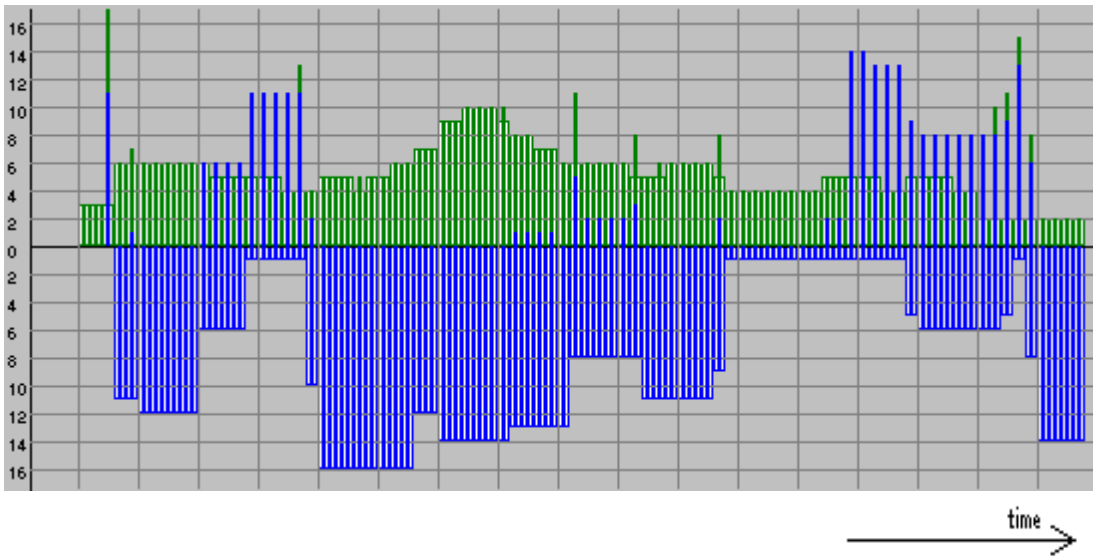

Figure 7. Original multi-tasking model shows no problems with overstaffing.

It can now be seen that there is a problem with the available models, and a new one must be created. A hypothetical model which has not yet been implemented, but which should solve the above problems is discussed in section 4.3.

4.2 Discussion of Figure 7

It can be noted from Figure 7 that there is a huge 'unnecessary' spike near the beginning of the roster where 17 staff are employed and only 3 required. This occurs because the parameters were set so that all demand should be met or exceeded, but that increments in shift start times, and shift lengths must be multiples of two periods. This parameter was set to speed up computer processing time, and so that staff would only start their shifts on the hour or half-hour, and their shift lengths would be multiples of 30 minutes. Thus the surplus spike is mandatory with this shift under these conditions.

It can also be noted from Figure 7 that there are two sections in the roster where there is a large overstaffing of A-staff for a significant number of periods. These occur because of the arrivals (histogram below the axis) peaks towards either end of the roster. In the parameter file, it was set that all A-staff must work a minimum of 30 periods (7.5 hours) in any one day. Another requirement in the parameter file was that all staff demands must be met. Therefore the two lengthy sections of overstaffing are inevitable in this example, for these particular parameters. Note that the user can set these parameters to suit the problem at hand, or make slight alterations to get rosters with

varying characteristics. By altering these parameters the user can affect the quality of staff rosters, as well as the total hours paid, or cost to the company.

4.3 A Revised Proposed Model

It is clear from the problems discussed in section 4.1 that the slacks and surpluses are not being handled correctly. The current A-matrix structure which SETPAR implements is shown above in Figure 3 with inequality constraints. This A-matrix structure is rewritten in Figure 8 with equality constraints. Note that for simplicity's sake, only the slacks have been shown. In reality, the surplus columns would be the same as the slack columns except negative.

	trained		less-trained		slacks / surpluses				
period 1	0	1	1	1	1	1	0	0	0	departures
period 1	0	1	1	0	0	0	1	0	0	departures + arrivals
period 2	0	0	1	0	1	0	0	1	0	departures
period 2	0	0	1	0	0	0	0	0	1	departures + arrivals

Figure 8. Original A-matrix multi-tasking structure with equality constraints.

There is clearly a problem with the structure outlined in Figure 8, as a slack column can affect the departures + arrivals constraint without affecting the departures constraint. In order to correct this unique problem, the SETPAR program must be adjusted so that slacks (and surpluses) are dealt with as shown in Figure 9. Although this has not yet been implemented or tested, it should work, as a slack which affects the departures + arrivals constraint must also now affect the departures constraint.

	trained		less-trained		slacks / surpluses				
period 1	0	1	1	1	1	1	1	0	0	departures
period 1	0	1	1	0	0	0	1	0	0	departures + arrivals
period 2	0	0	1	0	1	0	0	1	1	departures
period 2	0	0	1	0	0	0	0	0	1	departures + arrivals

Figure 9. Proposed A-matrix multi-tasking structure with equality constraints.

4.4 Three or More Work Types for Three or More Staff Ability Levels

The next hurdle for the author to overcome is the prospect of a larger number of work types, and staff abilities. Indeed it has been requested by the client, NZ Customs, that a model be created whereby three (or even more) levels of staff can work the roster with flexible duties. This model is still in the developmental stage at the time of writing this article. It is expected the model will be a fairly simple extension of the above multi-tasking model. An example of an A-matrix for a 4-level roster is shown below in Figure 10.

	supervisor		secondary		trained		less-trained		
period 1	0	1	1	1	1	1	1	1	1	departures

period 1	0	1	1	1	1	1	1	0	0	arrivals
period 1	0	1	1	1	1	0	0	0	0	secondary
period 1	0	1	1	0	0	0	0	0	0	supervisor
period 2	0	1	1	1	1	1	1	1	1	departures
period 2	0	1	1	1	1	1	1	0	0	arrivals
period 2	0	1	1	1	1	0	0	0	0	secondary
period 2	0	1	1	0	0	0	0	0	0	supervisor

Figure 10. An example 4-level multi-tasking A-matrix.

A hypothetical graphical representation of a 3-level roster is shown in Figure 11.

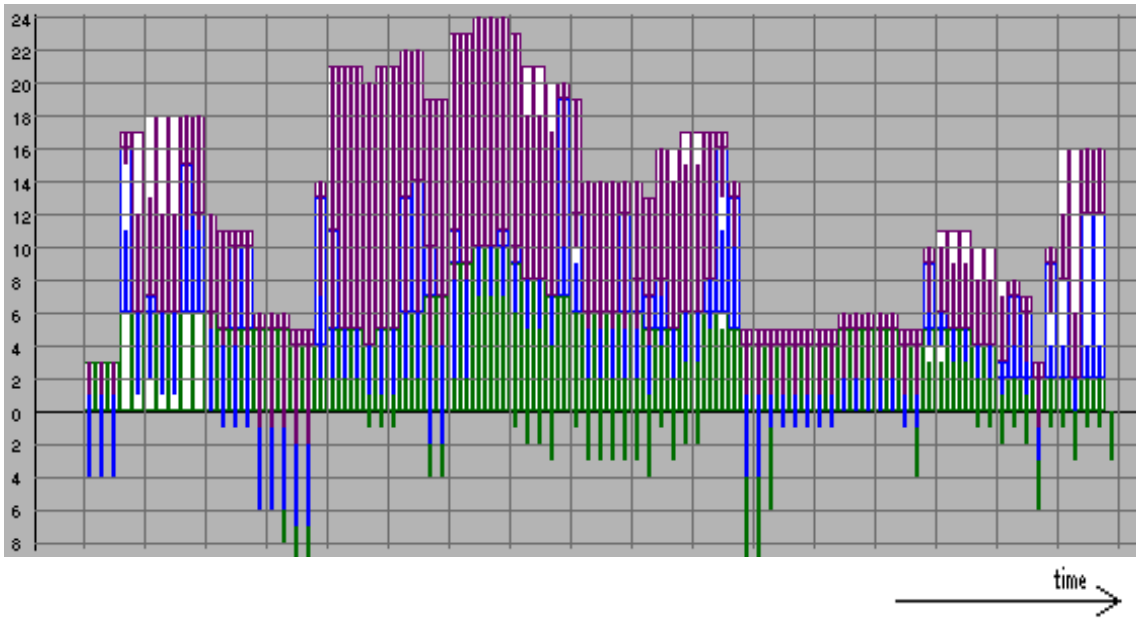

Figure 11. Example of a possible graphical representation for a 3-level multi-tasking roster.

For the case of a 3-level roster, the extra task is called “secondary”, which is done by secondary staff, who can, if need be, work at arrivals or departures.

5 Discussion

Eventually it is expected that the program will be able to generate rosters for any number of staff types for a similar number of job levels. As the project proceeds through its developmental stage, it may well pick up many more new, improved features as desired by the client. One of these possible features that is still in the brainstorming stage is the issue of time taken for staff to transfer work station from task to task. If we require staff to be constantly switching from arrivals to departures and back again, this

perhaps should incur a penalty of some sort, as a disincentive. There are no doubt other such issues yet to be raised and worked on.

It is expected that this project will soon have reached a level whereby it can be run using real data, and compared with rosters created by the current optimisation package. The first experimental run will be done with parameters set to solely minimise hours of labour required. The second and following runs will experiment with varying certain input parameters. The effects these have on the objective financial cost as well as any improvements of the shifts will be studied. Hopefully there will be significant improvements of the shifts generated at little or no additional total hours paid.

Acknowledgments

Project Supervisor: Dr Andrew Mason - Engineering Science Department - University of Auckland.

References

G. Page, *Construction of Optimal Weekly Schedules and Rosters at Dairy Queen International*, Fourth Year Industrial Maths Project, 1997.

M. Smith, *Optimal Nurse Scheduling Using Column Generation*, Masters Thesis, 1996.