AURORA - AUTOMATED ROSTERING APPLICATION

Dick te Winkel, KLM Royal Dutch Airlines
Erik Tilanus, KLM Royal Dutch Airlines

1. Introduction

During SUGI 1982 KLM presented the PRONTO system: Duty-roster maintenance. It covered the last segment of the complete rostering process for shift personnel.Much of an airline's staff, such as overhaul technicians and airport ground personnel, work in shifts. An extra complication in creating rosters for them is that there is a varying workload during the day, related to the airline schedule, resulting in a varying number of people per shift. The first step in the duty roster process is therefore to determine the staffing of each shift on each day of the week. This is done in "manpower planning models". Input to these models is:

(i) the airlines' schedule,
(ii) working standards like: "arrival of a DC10 requires X baggage loaders from Y minutes after arrival until Z minutes after arrival" and
(iii) the possible shift start and end times.

The output of these models is a staffing figure for each shift for every day of the week (fig.1), the "covertable". The set of figures from the covertable covers as comprehensive as possible the manpower-needs ("profiles") over the week.

NET NUMBERS OF EMPLOYEES
PER DAY PER SHIFT

MON	TUE	WED	THU	FRI	SAT	SUN	START	END
2	4	4	3	7	6	8	0600	1300
16	16	16	11	14	16	15	0600	1400
17	14	15	15	18	11	14	0700	1500
6	2	2	7	3	6	4	0900	1730
8	13	13	9	10	10	15	1000	1830
6	6	9	5	9	8	4	1330	2230
6	2	4	6	6	4	6	1430	2330
4	3	3	4	3	3	3	1500	2345
3	3	3	3	3	3	3	2345	0600

fig. 1
The "net covertable": required staff on duty by shift and day of week.

Until recently the next phase in the process was manual: transforming the covertable into a roster in the shape of figure 2. This roster indicates the sequence of shifts (by means of their code, which will be explained later) and the days off as they are scheduled from week to week. Each person who will be assigned to the roster starts on a specific week and cycles through the roster, moving down one line every week. The third phase, covered by the PRONTO system, is to assign the names to this roster and to maintain it, i.e. to record and control all requests for changes, such as leave requests, holidays, sickness, staff changes etc. AURORA fills the gap between the manpower planning models and PRONTO. It depends largely on the new SAS/OR facilities. The creation of the duty rosters is done in three steps:

I. Initialization
II. Creation of a Days-Off pattern
III. Allocation of shifts.

WEEK	MON	TUE	WED	THU	FRI	SAT	SUN
1	M2	M2	M2	M1	--	M2	M2
2	E2	E1	--	--	M1	M1	D1
3	D1	--	E1	D3	D3	--	--
4	D1	D1	D1	M2	--	M2	M2
5	E2	E1	--	--	D1	M2	M2
6	--	D1	D1	D1	M2	--	--
7	M1	M1	M1				
				E2	E1	--	--
117	D1	M2	M2	--	E3	E2	E1
118	D3	--	--	E3	E2	E1	D3
119	--	E1	E1	D3	D3	--	--
120	D1	D1	D1	D1	--	E3	E2
121	E2	--	--	E3	E2	E1	D3
122	--	E1	E1	D3	D3	--	--

fig. 2
The intended final result: the roster. Each person is assigned to a specific row for one week and moves to the next row for the next week. Shifts are coded by start/end times:
M# shifts are early shifts
D# shifts are daytime shifts
E# shifts are evening shifts
N# shifts are night shifts.

2. Step I: Initialization

The manpower planning provides the net staffing figures, the "net covertable". However, before making a roster, a gross covertable has to be calculated to accommodate contingencies. Contingencies can vary from department to department, but commonly recurrent are: Public Holidays, Vacation, Sickness, Days Off in Lieu, and Training. The gross covertable is calculated as follows: since the

normal work week in The Netherlands has five workdays, there are 261 workdays p.a., not taking into account any contingencies. If for a specific department the number of workdays lost to contingencies equals 52, the attendance-factor is (261-52)/261 = .8 for this department. This means that of a typical workweek .8 * 5 = 4 days are effective workdays. Multiplying the numbers of the net cover table by 1.25 (= 1/.8) will give the gross covertable. The attendance-factor has to be predicted, using observed figures for contingencies in the past. In the rostering model discussed here, both the net covertable and the attendance-factor are entered into the model by SAS/FSP. Each shift becomes a type-indication according to the labour legislation on start and end times: e.g. "D1" is the first shift of the group of day-shifts, "N1" of the night-shifts. This type-indication will be used throughout the duty roster. The calculated gross covertable is also displayed on the screen and can be modified by the user. The roster has to be made, based on the five workdays a week principle, with the figures of the gross covertable. Labour-legislation and company collective agreements may necessitate making adjustments in the covertable figures. In The Netherlands, one of the most important items in the legislation for shift labour is that each duty roster must have a Sunday off at least every three weeks. Another item is that the average number of working hours per week must not exceed 40.

ADJUSTED GROSS NUMBERS OF EMPLOYEES PER DAY PER SHIFT

MON	TUE	WED	THU	FRI	SAT	SUN	START	END	CODE
3	5	5	4	8	7	9	600	1300	M1
17	17	17	12	15	17	16	600	1400	M2
33	29	30	30	29	17	15	700	1500	D1
7	3	3	8	4	7	5	900	1730	D2
9	14	14	10	11	11	16	1000	1830	D3
7	7	10	6	10	9	5	1330	2230	E1
7	3	5	7	7	5	7	1430	2330	E2
5	4	4	5	4	4	4	1500	2345	E3
4	4	4	4	4	4	4	2345	600	N1

NUMBER OF EMPLOYEES NEEDED: 122
NUMBERS OF DAYS-OFF:
30 36 30 36 30 41 41
TOTAL NUMBER OF SHIFTS: 610
TOTAL NUMBER OF WORKING HOURS: 4663.5 (EXCL. BREAKS)
AVERAGE WORKING HOURS PER WEEK: 38.23

fig. 3
Calculated gross figure of required staff ("gross covertable") by shift and day of week, with key figures.

If the covertable doesn't allow a Sunday off every three weeks, additional staff may be required, since Sunday staff on-duty may not exceed twice the staff off-duty.
The rostering model will make these adjustments for the user according to his priority rules. At this stage, the number of employees needed to roster the shifts will be calculated by taking the sum of all shifts in the (adjusted) covertable and dividing this sum by 5 (= number of workdays a week). The difference between the staff on-duty from the covertable and the total number of staff as calculated here provides the number of people off-duty on every day of the week. (fig.3).

3. Step II: Days-off pattern

The above mentioned rule that any duty roster must have at least one Sunday off every three weeks puts severe limitation on the freedom to create a days-off pattern. Most days-off techniques that are suggested in the literature can't be applied because of the "Dutch Sundays-off rule". This made it necessary to look for another technique: using linear programming. Because of the "Sunday-off rule" each feasible duty roster must be composed of units ("building blocks") of one week, two weeks or three weeks, each having one Sunday off (the last Sunday in the building block). This property can be used in the creation of a days-off schedule. Since a unit of three weeks should have 6 days off including one Sunday, a building block with no Wednesday-off will look like that in figure 4.

MON TUE WED THU FRI SAT SUN

1	1	1	0	0	1	1
1	0	1	1	1	1	1
0	1	1	1	1	0	0

fig. 4
The "building block": a group of rows in the roster with only one (the last) Sunday off duty. In the Netherlands only building blocks of 1, 2 or 3 weeks are permissible. (1 means on duty, 0 means day off)

In this building block there are two duty days for every day of the week except Wednesday, with three duty days. The user may reallocate the days off within this building block as long as the Sunday-off stays on the bottom line and there are two duty days on every day of the week with the exception of three duty days on Wednesday. In the same way building blocks can be created with no Monday off etc. The building blocks of two weeks with one Sunday off and a selection of

three other days off are made in a similar way as are one week building blocks with the Sunday and one other day off. The building blocks have in common that the Sunday-off is always on the bottom line and therefore the blocks can be linked to any other building block in any sequence. The result will always contain the requisite distribution of days-on and days-off. Thus the roster becomes a combination of building blocks where:

a. the total number of weeks is equal to the total staff

b. the number of days off corresponds to the figures in the covertable.

This is an almost classical LP problem, which can be easily solved by PROC LP. Each type of building block is a variable in the LP problem. The solution of the PROC LP gives the types of building blocks and their numbers that will satisfy the constraints. The selected building blocks will be linked together in such a way that:
(i) the blocks with two Sundays-on, one Sunday-off will be spread evenly over the roster; the same applies for those with one Sunday-on, one Sunday-off and those with the Sunday-off, and
(ii) the blocks with no Monday-off will be spread evenly over the roster; the same applies for those with no Tuesday-off, ..., no Saturday-off. The resulting dataset is reformatted and displayed using PROC FSEDIT (fig.5).

WEEK	MON	TUE	WED	THU	FRI	SAT	SUN
1	1	1	1	1	0	1	1
2	1	1	0	0	1	1	1
3	1	0	1	1	1	0	0
4	1	1	1	1	0	1	1
5	1	1	0	0	1	1	1
6	0	1	1	1	1	0	0
7	1	1	1	0			
					1	1	1
118	1	0	0	1	1	1	1
119	0	1	1	1	1	0	0
120	1	1	1	1	0	1	1
121	1	0	0	1	1	1	1
122	0	1	1	1	1	0	0

fig. 5
The "day-off pattern", result of step II. 0 - day off
1 - duty day

At this point in the process it is possible to apply amendments to the day-off pattern as desired, as long as the total number of staff off duty for each day of the week is unchanged. After verification the now created days-off pattern becomes input for step III: Shift Allocation.

4. Step III: Allocation of shifts

Would you consider allocation of shifts in the roster to be a network flow optimization problem? Probably not. Yet it can be described as such. Let us have a look at the situation at this stage in the process: we have a days-off pattern that has exactly the number of duty days on each day of the week that we find in the covertable of fig. 3. Our problem is to allocate the shifts to the duty days in such a way that the resulting roster will be acceptable to labour-legislation and attractive to the personnel for whom the roster is created. Feasibility according to labour-legislation states that between the end of a shift and the start of the next shift there must a ten hour rest-period. Attractiveness for the employees may mean a forward rotation (after a day-off, work will start with an morning shift, followed by a day shift the next day and an evening shift on the following day etc.) or a backward rotation, where time between shifts is kept to a (legal) minimum. This problem might have been solved with linear programming. However, attempts to do so failed: PROC LP did not provide integer solutions. Searching for another way led to PROC NETFLOW. To use PROC NETFLOW the roster has to be considered as a network, each node representing a type of shift on a specific day of the week. In fact, each shift has two nodes, one left and one right.

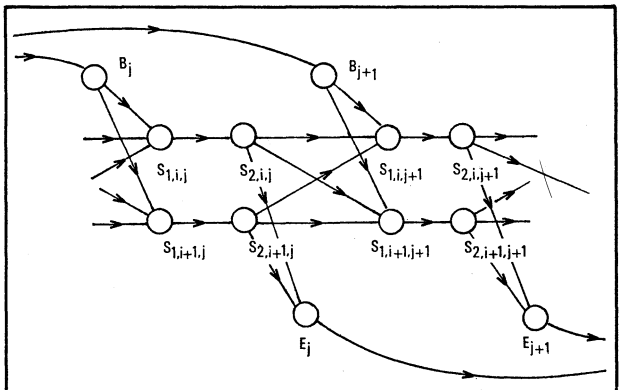

fig. 6
The roster presented as a network: a worker "flows" from a shift on Monday to a shift on Tuesday. A day off, preceding a duty-day acts as source in the network, a day-off after duty-days is the drain.

The arcs in the network represent the possibility for a shift to be followed by a shift on the next day. The cost of this arc is the undesirability of linking these shifts. Fig. 6 shows a detail of the network. In this figure, only two shifts (i and i+1) are shown on two consecutive days (j and j+1). The capacity of arc $s_{1,i,j}$ to $s_{2,i,j}$ is

equal to the number of shifts of type i on day j. The arcs coming in from above, used for first shifts of a rotation (the previous day was a day-off), are "begin"-arcs.The arcs curving downward, used to represent the last shift before a day-off, are the "end"-arcs. To find out what capacity the incoming arc of node B_j should be, we count the number of times in the days-off pattern that day j is a workday while day j-1 is a day-off. In the same way, the capacity of the outgoing arc of node E_j is the number of times that day j is a workday while day j+1 is a day-off. Having set up the network in this way, it is possible with PROC NETFLOW to calculate the set of linkages between shifts that should be selected to make the roster legally feasible and as attractive as possible for the users. However, as PROC NETFLOW only determines linkages and no complete rotations, fitting the linkages from the optimal solution into the days-off pattern, may form a puzzle impossible to solve. This is due to the fact that there is no way to determine whether a linkage is vital to a particular rotation. Therefore the solution of the allocation problem is reached in seven steps, one for each day of the week. First count the number of rotations that begin on Monday and that end on Monday, Tuesday, ..., Sunday. Then use these numbers for the capacities of the outgoing arcs of nodes E_1 to E_7 and the sum of these numbers for the capacity of the incoming arc of node B_1, to calculate a partial solution of the allocation problem. From this partial solution all rotations beginning on Monday can be composed and filled into the days-off pattern. Next all capacities are reduced with the flows found in the solution of the netflow-problem and the same procedure is applied to solve the partial problem for Tuesday. After completing this procedure for Sunday the roster will have been accomplished.

5. Experience with SAS/OR

KLM used a Fortran Linear Program before PROC LP was announced. This enables us to compare. The Fortran LP package is about as fast as PROC LP, but the matrix generation for the Fortran package is much less user-friendly than the generation of the input dataset for PROC LP. However, if the columns of the matrix are generated as observations of a dataset, that dataset has to be transposed later on. PROC TRANSPOSE can do it, but may need a lot of memory space. SAS/OR functions as a genuine SAS product: easy to use and reliable. In many situations however an integer option in PROC LP is a necessity, also in AURORA, since only integer solutions of PROC LP can be used for making the days-off pattern. Fortunately PROC LP has yielded here only integer solutions so far.

6. Experience with AURORA

Both the people that are involved in roster preparation as those working in shifts tend to react with reservation on computerized rostering.
The AURORA approach of creating rosters in three distinctive steps, each of them yielding concrete adjustable results, seems to help people to overcome this reservation. They realize the advantages of AURORA over the manual procedure:

- It is easy to predict the influence of any changes in the flight schedules on the rosters, since the whole procedure of drafting a new roster may be completed in about 15 minutes.
- The method results in a much better balance between available manpower and workload.
- The consequences of any amendments made to the intermediate results can be quickly analysed. Many alternatives may be evaluated in a short period of time.

AURORA is now in Beta-test at 2 departments within KLM. Both small and large rosters have been generated (varying between 28 and 127 rows). The reactions of those departments are in general positive. The flexibility to adjust to changing circumstances as well as the ability to evaluate alternative approaches (e.g. forward and backward rotation) are considered the most important advantages.
The AURORA project is not yet finished. The problem of creating rosters has been solved. This is however only the first step. The next step is a management of change problem: AURORA allows many shift start and end times, much more than was practically feasible in manual rostering procedures. This is a powerful feature,but it requires adaption of the way shift labour is planned.

further information about AURORA may be obtained from:

KLM Royal Dutch Airlines
dept. AMS/XB
p.o. box 7700
1117 ZL Schiphol
Netherlands