# An Incidence Matrix Approach to Nurse Scheduling

by

Norm Goertzen

A thesis
presented to the University of Manitoba
in partial fulfillment of the
requirements for the degree of
Masters of Science
in
Department of Actuarial and Management Sciences

Winnipeg, Manitoba

# AN INCIDENCE MATRIX APPROACH TO NURSE SCHEDULING

BY

NORMAN GOERTZEN

A thesis submitted to the Faculty of Graduate Studies of the University of Manitoba in partial fulfillment of the requirements of the degree of

MASTER OF SCIENCE

**ABSTRACT**

One of the most time consuming and frustrating chores faced by hospitals is that of nurse scheduling. Hospitals would like an efficient method of scheduling their nurses as to provide sufficient staff coverage. However each hospital, and in fact each hospital ward faces a unique scheduling problem. The minimum number of nurses required on a shift will differ. Collective agreements between the hospitals and the nurses will vary from hospital to hospital. These agreements will impose restrictions on schedules such as maximum work stretch, types of shifts, weekend policy and patterns of rotation.

This thesis presents an algorithm for scheduling nurses under a wide variety of different labour constraints. The resulting nurse schedules will be optimal in terms of either minimizing the number of nurses required or minimizing the maximum surplus of nurses scheduled to work a shift. The algorithm is implemented using BASIC on a microcompututer with only 64K.

**ACKNOWLEDGEMENTS**

There are several individuals whose suggestions and guidance contributed to the value of this thesis. Special thanks and appreciation go to Dr. Earl S. Rosenbloom, who greatly improved the quality, readability, and relevance of my work. Also, I wish to thank the members of my thesis committee, Dr. Suresh K. Bhatt, Dr. S. Amir Bukhari, and Dr. Zowie Wharton, whose comments and recommendations were invaluable.

CONTENTS

## LIST OF TABLES

## LIST OF FIGURES

## Chapter I

## PURPOSE

Many organizations require employees to work shifts. Some method of rotating shifts is frequently used to ensure proper staffing levels, and to provide all employees with a fair share of the undesirable shifts. Typical organizations are: Police forces, ambulance drivers, hospital personnel, and industrial factories. This thesis draws its incentive from the study of nurse scheduling; however, the method used is relevant to any industry that schedules its manpower into shifts.

A description of the nurse scheduling problem could be: "the allocation of nurses to work patterns such that:

1. In each ward there are an adequate number of nurses to perform the necessary duties.
2. The allocation is as efficient as possible. And,
3. Unreasonable work patterns are avoided."

Of these three points, only the first is avoided in this thesis. It is assumed that the hospital administration knows exactly how many nurses are required at each place, on each day, and for each shift. This usually is the case since the coverage requirement is the result of some subjective assessment of need, or is a product of some other form of analysis.

The second point can be interpreted as the requirement that any solution be mathematically optimal. Optimality is a very desirous goal, however the cost in obtaining it are sometimes excessively high. Also, optimality cannot be clearly defined. It normally implies that some indicator is the best it can be—but what indicator should be looked at? Some possible choices are: the least total cost, the fewest number of nurses, the greatest minimum daily coverage, the greatest average daily coverage, the smallest standard deviation of coverages, or perhaps some combination of them all. In general, a sub-optimal, but good[1] solution is acceptable.

The final point focuses on the most complex area with scheduling theory—how can the seemingly infinite number of solutions be adequately, but efficiently, investigated and an answer obtained. Although the collective agreement[2] would describe several work pattern features that must exist, and therefore most combinations can be removed, the number of answers to a typical problem is still enormous. Even relatively small problems can lead to large numbers of answers; enumeration and inspection of all solutions is practically impossible. Previous nurse scheduling approaches have overcome this obstacle by one of several methods: (1) utilization of enormous computer resources, (2) solving for only a specific class of problems, (3) reducing the feasible region through unreasonable or unrealistic restrictions, or (4) utilizing complicated, but suspicious, heuristic

[1] Any solution that is obviously close to the optimal solution, or perhaps even optimal, but its optimality cannot be proven.

[2] Usually between the hospital (employer) and the nurses (employees), there exists a set of rules and regulations which outline acceptable patterns of work for the nurses.

algorithms. Most approaches are also highly problem-specific; if a single problem feature were changed, the solution method would likely have to be scrapped and a different approach taken.

Not surprisingly, the major incentive for this thesis was to overcome these difficulties, and create an implementable system. The overall purpose was to develop a method which would meet, as best as possible, each of the following goals:

1. The approach can be applied to a wide variety of different problems.
2. To simplify schedule generation and implementation, the solution must be cyclic.
3. Some measure of optimality is achieved—or at least closely approached,
4. A fully working scheduling program is implemented on a micro computer.
5. No restriction is introduced, solely to reduce the feasible region size; whereby solution quality is threatened.

## Chapter II

## PROBLEM APPROACH

A variety of Operations Research techniques have been suggested for helping hospitals with scheduling problems. These techniques can be partitioned in to two major categories, Combinatorics and Mathematical Programming. Unfortunately, neither of these techniques have been implemented in very many hospitals.

The Combinatorial approach examines the underlying Mathematical structure of the problem. Examples of this approach can be found in papers by Baker(1)&(2), Smith(7)&(8), Burns(2)&(3), and Orlin(5). In effect, this approach attempts to minimize the number of nurses employed while satisfying both the staffing requirements and the labour constraints imposed by the collective agreement. The minimization is often accomplished by rather elegant combinatorial algorithms. Unfortunately this approach is very limited. It would only be successful in a hospital ward with a very restrictive set of labour constraints.

The Mathematical Programming approach is often much more ambitious. Here every feasible schedule for a particular nurse is generated. Nurses are then assigned a schedule which satisfies the staff coverage requirements and maximizes total nurse satisfaction with the schedule. A nurse satisfaction score is obtained through a questionnaire on individual nurse preferences. Examples and variations on this

approach can be found in papers by Rothstein(6), Warner(9)&(10), and Kostreva, Leszcymski, and Passini(4). Although the resulting quality of the schedules generated is high, this approach has rarely been implemented. In order to obtain a schedule a large scale Mathematical program must be solved for each ward in a hospital. This is usually beyond the computing capacities of most hospitals.

The approach in this thesis could be described as a hybrid of the above two. Although a linear program must be solved in order to obtain the schedules the size of the program is kept small by exploiting the unerlying mathematical structure.

## 2.1 PRESENTED RESEARCH

During the development of the method, several key objectives were kept in mind. The first and foremost, was to reduce the overall problem size as much as justifiably possible. Obviously there are several reasons for this. The larger the problem, the less likely that a computer implementation is possible. Therefore, all other things being equal, the preferred formulation has the fewest variables and/or constraints.

A second objective was to obtain a cyclic solution. There are several benefits derived from this. If the solution can be repeated infinitely, the size of the problem is immediately reduced, while the frequency of solving it are lowered. In addition, some nurses may prefer the repeating, and therefore predictable, schedules.

The first objective led to the idea that the basic unit should be one week. This 7 day pattern of work days is referred as a SCHEDULE throughout this thesis. This in turn, defined the number of nurses, who should work a particular schedule each week, to be the fundamental variable. This limited the number of variables to, at most, 128.[3] Most of these schedules could be avoided, since they had some feature that violated a problem parameter. The observation that some schedules are never used, led directly to the idea that there could exist an elimination procedure which would reduce the number of variables. This procedure was called the STATIC ELIMINATION PROCEDURE, since the schedule were singularly examined.

If nurses are moving into, and out of, the weekly schedules according to some plan, the perhaps, it was imagined, that plan could maintain a constant (and hopefully repeatable) system state. This single thought developed into the concept that an incidence matrix[4] could describe the movements between schedules, and also repeatability. The second objective was thus met. Here, incidence matrices were ables to show how the various schedules could 'connect to each other'. However, not all connections were allowed, since it was possible that a valid schedule would not properly connect with other valid schedules. Schedules that could not connect at all with

---

[3] There are only 2 states for each day; worked (1), or not worked (0). Because there are 7 days per week, there are 2 to the power of 7, possible combinations. Each combination represents one schedule.

[4] Based on an 'every other weekend off policy (1/2)', a 2 dimensional incidence matrix is appropriate. There are obviously other combinations such as: 1/3, 2/4, 2/5, etc. These combinations require the extension to 3, 4 and 5 dimensional incidence matrices. However, this thesis deals only with the '1/2 weekend off' case.

other schedules were therefore unusable. This observation developed into what was called the DYNAMIC ELIMINATION PROCEDURE, since the schedules were now being as part of a continuous series.

With a typical problem, the elimination procedures significantly reduced the number of valid schedules. These remaining schedules were then transformed into a linear program. This linear program was then solved. Almost always, integer solutions were the result. The solutions could then be used to construct a solved set of inter-schedule movements, and then directly implemented. Typically the solution would then be applied to each specific ward within a hospital. A hospital may contain over a hundred wards, thus the linear program would need solving many times.

Most hospitals prefer to organize the nurses into rotating shifts. The main problem with multiple shifts is to find a structure that transfers nurses between shifts, while maintaining adequate daily coverages. The approach taken, was a further application of an incidence matrix. Here the transportation algorithm was utilized to create the nurse movements between shifts. This permitted the overall problem to be divided into several smaller problems, each shift being considered as one problem.

## Chapter III

## DEFINITIONS

The study of nurse scheduling, like most fields, makes use of its own terminology. To help understand this thesis, familiarity with these frequently used terms is recommended. Below are terms general to the study of nurse scheduling; albeit, their definitions may vary somewhat, depending on the user.

SCHEDULE

A SCHEDULE is a 7 day work pattern beginning with a Monday and ending with a Sunday. This schedule exists by itself and is unrelated to the nurses that may work its pattern. For any 7 day week, there are 2 to the 7th, or 128, possible combinations of days worked and days off. A '1' represents a day worked, and a '0' a day off. Therefore, possible schedules range from '0000000' through to '1111111'. The schedule '0101001' represents a working day on Tuesday, Thursday, and Sunday, with all other days off.

WORK STRETCH

A WORK STRETCH is a group of consecutive work days, bounded by days off. Any string of '1's, within any pattern of days, can be considered a Work Stretch, as long as there is at least one '0' at each end.

OFF STRETCH

Just as the Work Stretch is a group of work days, an OFF STRETCH is a group of consecutive off days. Like the Work Stretch, the Off Stretch requires bounding work days to delimit the string of '0's.

DAYS WORKED

DAYS WORKED represents the number of days that a nurse must work within a period of standard length. This number depends on the number of hours per shift. If the shifts are 8 hours long, then the number of working days per week would likely average to 5. Thus, if the period is 2 weeks, then each nurse must work 10 of those 14 days.

SHIFTS

Since most wards must maintain staff 24 hours a day, there are usually several SHIFTS each day. Typically a day is divided into 2 or 3 Shifts. If there are 2 Shifts, each would be 12 hours long; if 3 Shifts, then 8 hours long. However, each Shift typically does not have the same daily requirements as the others; days usually need the most nurses, followed by evenings, and finally nights. Nurses may also change Shifts from time to time. Each time there is a change however, certain rules of intervening days off must be observed.

WEEKENDS OFF

Mostly because the WEEKENDS are perceived to be valuable, nurses tend to prefer as many OFF as possible. They also expect to be provided at steady and regular intervals, say every second, or third week. Since the weekends are highly valued, the number of nurses needed to meet the daily requirements is usually quite dependent on the characteristics of the Weekend Off policy. Throughout this thesis, only the 'every other Weekend Off' policy is examined.

SPLIT WEEKENDS

A SPLIT WEEKEND is any weekend where either; the Saturday is worked while the Sunday is off, or the Saturday is off and Sunday is worked. Many collective agreements disallow Split Weekends.

REQUIREMENTS

Each ward in a hospital, has its own characteristic tasks that must be performed. These tasks vary in quantity and type for different days and different shifts. Because of this fluctuating demand for labor, and the compromises made to provide frequent weekends off, the daily REQUIREMENT is dependent on the day of the week. Typically, the weekdays (Monday through Friday) have the highest Requirements, while while the weekend days are slightly less (usually 50% to 80% of the weekday amount). Also, the various shifts have their own specific Requirements; day shifts usually need the most, with the other shifts needing much less.

## Chapter IV

## SOLUTION SUMMARY

The solution method proposed in this thesis involves several distinct steps. Although each step will be dealt with individually later, here is a a short preview of them. The short discussions which follow, provide a more thorough description of the steps, however they also refer to several variables critical to that step. These variables are defined in the next chapter.

The solution method can be separated into 10 steps. Figure 1 is a flowchart which relates these steps to each other, and references the chapters where each step is discussed in detail. Notice that Step #1 is performed once, and only once. It is an standard representation of all possible schedules. This schedule list was initially generated, and thereafter simply duplicated. Steps #2 through #10 are performed, as required, for each new problem. But, only Steps #6 through #8 need to be performed for each set of daily requirements. Nonetheless, requirements may remain constant for long periods of time.

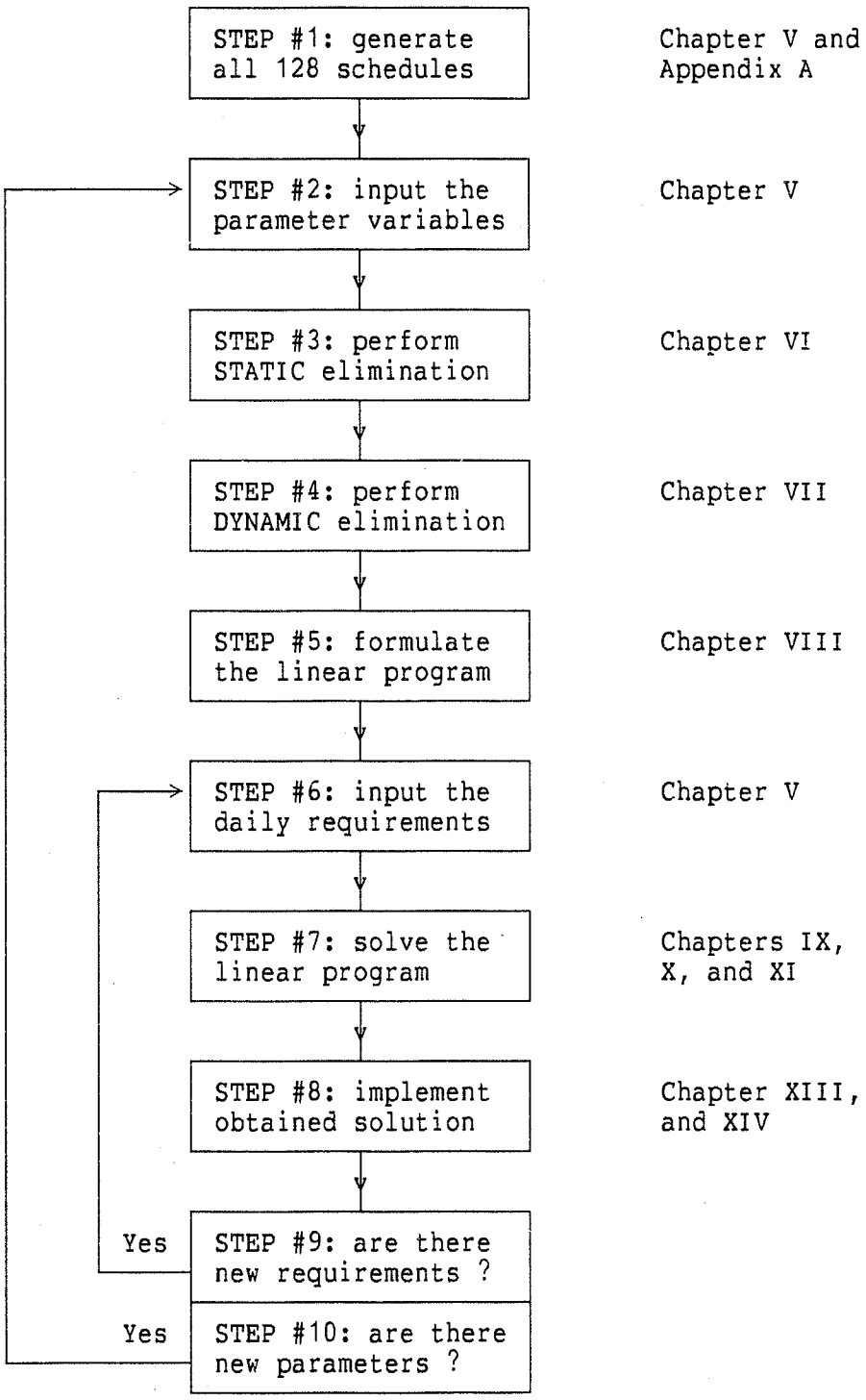


Figure 1: Overall Procedure

1. Generate all 128 schedules
   Since there are 7 days in a week, and there are only 2 states a day could have, there are 128 unique combinations of worked, and not-worked days. Each combination is represented by the following variables: MON, TUE, WED, THR, FRI, SAT, and SUN. Along with these, the schedule number and many other variables are calculated and stored. They are: WSMON, WSSUN, WSLO, WSHI, DWTOT, OSMON, OSLOW, and OSSUN. All these variables are defined in the next chapter. These are used in specific tests by both elimination procedures. The complete list of the 128 unique schedules is given in Appendix A.

2. Input the parameter variables
   The problem specific parameters include: WSMIN, WSMAX, OSMIN, DWMIN, DWMAX, SHIFT, and SPLIT. These define the problem environment of the whole hospital, since they are controlled by the collective agreement. Each variable will be formally defined later.

3. Perform STATIC elimination
   Since many of the 128 schedules will violate some parameter, this step removes them from further consideration. The ones remaining after this step, are stored for use by STEP #4.

4. Perform DYNAMIC elimination
   Construct an incidence matrix which relates each of the remaining schedules with themselves. Test the acceptability and non-acceptability, of movements between schedules. Since some schedules can never 'connect' properly with any other schedule, they can also be removed. Another product of this step is the incidence matrix of all remaining schedules, describing how these schedules connect with each other.

5. Formulate the linear program
   The incidence matrix is then converted into a matrix of variables, where the variables represent the exact number of nurses who move from one particular schedule to another. However, the values of these variables are unknown at this stage. If the solution is to repeatable, then the sum of movements out of a schedule, must equal the number of nurses moving into that schedule.

6. Input the daily requirements
   The ward specific, shift specific, and time specific requirements are: RMON, RTUE, RWED, RTHR, RFRI, RSAT, and RSUN.

7. Solve the linear program
   Since the solution represents the number of nurses in each schedule, the solution method must provide an integer solution. The values obtained are substituted back into the incidence matrix—replacing the variables. This matrix is referred to as the 'solution matrix'. If there are multiple shifts, this step is repeated—once for each shift. Each solution is then

incorporated into a single larger matrix. This matrix is further manipulated using transportation algorithm iterations.

8. Implement obtained solution
The obtained solution matrix defines the exact number of nurses who should be in each schedule, and exactly how they must move at the end of each week. Some solution matrices describe the movements very explicitly; others require careful interpretation. In either case, the solution matrix must be translated into a sequence of schedules for each nurse.

Do not proceed to the next step, until either the requirements or the parameters change.

9. Are there new requirements?
Frequently, the ward workload may change. When this occurs, the linear program found in STEP #5 must be re-solved, replaced with the new daily requirements. That is, go back to STEP #6.

10. Are there new parameters?
Infrequently, the problem parameters may change. This occurs when the collective agreement is altered, or other significant changes are desired. That is, go back to STEP #2.

## Chapter V

## ALGORITHM VARIABLES

Since the method utilizes some commonly referenced characteristics in nurse scheduling, the definitions presented here are, hopefully, intuitive.

These variables, and the definitions they imply, are used both in the thesis, and as variable names in the computer programs. The variables can be classified into 3 distinct categories: PARAMETERS, REQUIREMENTS, and DATABASE VARIABLES.

The PARAMETER variables describe the work pattern limitations and are usually specified by the collective agreement, and, to some degree, the discretion of the scheduler. The REQUIREMENT variables describe the daily coverages needed by the ward's workload. The DATABASE variables provide information about each schedule, and are required by the static and dynamic elimination procedures.

1. PARAMETERS

   a) WSMIN = the MINimum Work Stretch of consecutive days, a nurse is required to work. Work Stretches can equal this value, but they cannot fall short of it. Typical values are: 1 to 3.

   b) WSMAX = the MAXimum Work Stretch of consecutive days, a nurse is required to work. Work Stretches can equal this value, but they cannot exceed it. Typical values are: 5 to 9.

   c) OSMIN = the MINimum Off Stretch of consecutive days, a nurse is required to have off. Typical values are: 1 to 3.

d) DWMIN = the MINimum number of required Days Worked per period. Typical values are: 6 and 10. (depending on the number of shifts)

e) DWMAX = the MAXimum number of required Days Worked per period. Typical values are: 6 and 10.

f) SHIFT = the number of SHIFTs in each 24 hour day. Typical values are: 1 to 3.

g) SPLIT = are split weekends allowed? If split weekends are allowed, then SPLIT=1; if they are not, then SPLIT=0.

h) WEOFF = the number of WEekends OFF per period. When PERIOD=2, WEOFF is almost always set to 1. Other typical combinations of, (WEOFF/PERIOD) are: 1/3, 2/3, 1/4, 2/4, 2/5, 3/5, etc.

i) PERIOD = the number of weeks used as the scheduling duration which delimits the number of weekends off. This was previously referred to as the 'period of standard length'. It is also used by DWMIN and DWMAX.

2. REQUIREMENTS

a) RMON = the Required number of nurses needed to perform the work on a MONday.

b) RTUE = the Required number of nurses needed to perform the work on a TUEsday.

c) RWED = the Required number of nurses needed to perform the work on a WEDnesday.

d) RTHR = the Required number of nurses needed to perform the work on a THuRsday.

e) RFRI = the Required number of nurses needed to perform the work on a FRIday.

f) RSAT = the Required number of nurses needed to perform the work on a SATurday.

g) RSUN = the Required number of nurses needed to perform the work on a SUNday.

h) NURSES (or N) = the total workforce of nurses available for scheduling. This variable is only used in the formulation presented in chapter XI. There it supplements the daily requirement variables.

3. DATABASE VARIABLES (listed in Appendix A)

a) S = a unique Schedule identification number. Frequently, this thesis will refer to a specific schedule as S(i), meaning the (i)th schedule listed in the database.

b) MON = is MONday a workday? If Monday is a workday, then MON=1; otherwise, MON=0.

c) TUE = is TUEsday a workday? If Tuesday is a workday, then TUE=1; otherwise, TUE=0.

d) WED = is WEDnesday a workday? If Wednesday is a workday, then WED=1; otherwise, WED=0.

e) THR = is THuRsday a workday? If Thursday is a workday, then THR=1; otherwise, THR=0.

f) FRI = is FRIday a workday? If Friday is a workday, then FRI=1; otherwise, FRI=0.

g) SAT = is SATurday a workday? If Saturday is a workday, then SAT=1; otherwise, SAT=0.

h) SUN = is SUNday a workday? If Sunday is a workday, then SUN=1; otherwise, SUN=0.

i) WSMON = the length of the Work Stretch starting at MONday and counting forward. If Monday is off, then WSMON=0.

j) WSSUN = the length of the Work Stretch starting at SUNday and counting backward. If Sunday is off, then WSSUN=0.

k) WSLO = the LOwest (shortest) Work Stretch within the week. Work stretches that include either Monday or Sunday set WSLO=7. This is done so that the current schedule is not removed for falling short of WSMIN.

l) WSHI = the HIghest (longest) Work Stretch within the week. Here it does not matter if WSHI counts a Monday, Sunday, or neither.

m) DWTOT = the TOTal number of Days Worked in the week.

n) OSMON = the Stretch of days Off starting at MONday and counting forward. If Monday is worked, then OSMON=0.

o) OSLOW = the LOWest Stretch of Off days within the week. For a stretch of days off to be counted, there must be bounding work days at both ends. Schedules without both bounding work days make OSLOW=7. Again, this is so that the current schedule is not removed for falling short of OSMIN.

p) OSSUN = the Stretch of days Off starting at SUNday and counting backward. If Sunday is worked, then OSSUN=0.

## Chapter VI

## STATIC ELIMINATION

Static elimination is the first step towards reducing the problem size. If any of the original 128 schedules could be removed from consideration—then the problem will be obviously simpler. But exactly how these schedules can be removed, is discussed in this chapter.

Here, and throughout the thesis, a particular sample problem will be examined repeatedly. Figure 2 shows this problem, by setting each

| | | |
|---|---|---|
| WSMIN=2 | WSMAX=8 | RMON=6 |
| OSMIN=2 | | RTUE=6 |
| DWMIN=10 | DWMAX=10 | RWED=6 |
| SHIFT=1 or 3, in chapter XIII | | RTHR=6 |
| SPLIT=0 | | RFRI=6 |
| WEOFF=1 | | RSAT=4 |
| PERIOD=2 | | RSUN=4 |

Figure 2: Parameters and Requirements for the Sample Problem

problem parameter to a specific value.

Any feasible solution to this particular problem consists of a set of acceptable schedules, since most of the original 128 violate several parameters. In fact, all but the most severely unrestricted

problems have fewer than 64 valid schedules. There are several ways in which a schedule can violate a parameter. It is important to note, that many schedules violate more than one of the parameters; even one violation is sufficient to justify elimination.

The first parameter, WSMIN, can effect many removals. In this example, any schedule which includes a pattern of '010' is invalid, since each work stretch must cover at least 2 days. A pattern of '010' describes, a day off, a day worked, and a day off again. Any schedule that has this pattern is therefore invalid. There are 63 such schedules. If WSMIN were instead, less than 2, no schedules could be removed, since even the shortest work stretch would be acceptable. In general, the higher WSMIN is set, the more schedules that will be invalid, and thus removed.

If WSMAX were set to 6 (unlike this example), then '1111111' is invalid since it has 7 consecutive days. Obviously, if WSMAX is any value greater than 6, it cannot exclude any schedule. In general, the lower WSMAX is set to, the more schedules will be invalid.

Variable OSMIN behaves exactly like WSMIN, except that off-days are examined instead of work days. In the example, OSMIN is set to 2, thus any schedule with a pattern of '101' somewhere, is invalid. Here 63 schedules are invalidated by this variable. Again, if it were less than 2, it becomes ineffectual. The larger the value, the more schedules are invalidated.

A less obvious means of elimination can occur with the variable DWMIN. Since WEOFF is set to 1 and PERIOD is set to 2 here, the total

number of work days (in this case 10) must be allocated into 14 days. Since the most that can be worked in a single week is 7, the alternate week must contain at least 10 - 7, or 3, work days. Thus, any schedule with only 0, 1, or 2 working days in it, is invalid. For other combinations of PERIOD and WEOFF, similar calculations can be performed, however the usefulness of this test diminishes as PERIOD gets larger, or WSMIN gets smaller. In the example problem, 29 schedules are deemed invalid by this test.

Finally, the variable SPLIT can be an important remover of schedules. If, as is the case here, SPLIT is set to 0 (which implies that split weekends are not permitted), many schedules are invalid. Since any schedule ending in a '01' or '10' denote a split weekend, exactly half of the 128 schedules are invalid. If SPLIT is set to 1, then it cannot cause removal of any schedule.

These 5 distinct ways in which a schedule can be removed, constitute the 'STATIC ELIMINATION PROCEDURE'. All schedules are checked against each of the above tests; if any are deemed invalid by at least one test, then that schedule is eliminated. The ones that remain, will be kept and later examined by the dynamic elimination procedure. No other schedule-removing tests are possible at this stage.

In this example, static elimination forced out 109 invalid schedules, and left 19 good ones. These 19 schedules, and their related database variables, are given in table 1.

TABLE 1

Schedules after STATIC elimination

| S | MON | TUE | WED | THR | FRI | SAT | SUN | WSMON | WSSUN | WSLO | WSHI | DWTOT | OSMON | OSLOW | OSSUN |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 8 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 3 | 0 | 7 | 3 | 3 | 0 | 7 | 4 |
| 15 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 3 | 3 | 3 | 1 | 7 | 3 |
| 16 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 4 | 0 | 7 | 4 | 4 | 0 | 7 | 3 |
| 26 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 2 | 2 | 3 | 0 | 2 | 2 |
| 29 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 3 | 3 | 3 | 2 | 7 | 2 |
| 31 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 4 | 4 | 4 | 1 | 7 | 2 |
| 32 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 5 | 0 | 7 | 5 | 5 | 0 | 7 | 2 |
| 98 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 2 | 7 | 2 | 3 | 0 | 4 | 0 |
| 100 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 2 | 2 | 7 | 2 | 4 | 0 | 3 | 0 |
| 103 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 2 | 2 | 2 | 4 | 1 | 2 | 0 |
| 104 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 3 | 2 | 7 | 3 | 5 | 0 | 2 | 0 |
| 113 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 3 | 7 | 3 | 3 | 4 | 7 | 0 |
| 114 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 3 | 7 | 3 | 4 | 0 | 3 | 0 |
| 116 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 2 | 3 | 7 | 3 | 5 | 0 | 2 | 0 |
| 121 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 4 | 7 | 4 | 4 | 3 | 7 | 0 |
| 122 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 4 | 7 | 4 | 5 | 0 | 2 | 0 |
| 125 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 5 | 7 | 5 | 5 | 2 | 7 | 0 |
| 127 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 6 | 7 | 6 | 6 | 1 | 7 | 0 |
| 128 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 7 | 7 | 7 | 7 | 7 | 0 | 7 | 0 |

## 6.1 ALGORITHM

Below is the STATIC ELIMINATION algorithm.

1. Generate a list of the full 128 schedules. Examine the first schedule in the list.

2. Examine the pattern of days worked, and days off. If the pattern fails at least one of the proceeding tests, then remove that schedule from the list. If the pattern tests true for all conditions, then retain that schedule.

   a) WSLO ≥ WSMIN
   The shortest consecutive series of work days (which must include bounding work days) must be greater than, or equal to, the minimum required work stretch.

   b) WSHI ≥ WSMAX
   The longest consecutive series of work days must be greater than, or equal to, the maximum required work stretch.

   c) OSLOW ≥ OSMIN
   The shortest consecutive series of days off (which must include bounding work days) must equal, or exceed, the minimum allowed work stretch.

   d) DWTOT ≥ DWMIN - 7
   The total number of work days in the week must be greater than, or equal to, the minimum required days worked, less 7. If PERIOD is greater than 2, this test should be omitted.

   e) S < 33 or S > 96 while SPLIT=0
   If SPLIT=0, then split weekends are not allowed. If split weekends are disallowed, then the schedule numbers must be less than 33, or greater than 96. Schedules that have a split weekend, are schedule numbers 33 to 96, inclusive.

3. If the last schedule examined is the final one in the list, then stop. If it was not, then go to the next schedule in the list and continue at Step #2.

## Chapter VII

## DYNAMIC ELIMINATION

Since the basic unit is the schedule, once a nurse begins working one particular schedule, it must be continued until the end of the week. The following week, the nurse will work another schedule, and then another, and so on. But, the transition from the first schedule to the next, constitutes a 2 week pattern. This 14 day pattern must also conform to all the relevant problem parameters. Parameters WSMIN, WSMAX, and DOMIN must not be violated at the transition point (days 7 and 8). In addition the pattern must satisfy the other parameters: DWMIN, DWMAX, WEOFF, and PERIOD. The dynamic elimination procedure checks for both conditions. A total of 6 distinct tests are performed on each possible combination of one schedule paired[5] with another. If the pattern violates even one of the parameters, it becomes a pair of schedules that together no nurse may work.

Unlike the static elimination procedure, schedules are not removed because of a mere violation. Just because schedule X, when connected with schedule Y, violates a parameter, does not imply that either must be removed. It may be that schedule X, when connected with another schedule Z, does not violate a parameter; similarly, it may be that

[5] The terms 'connect', 'concatenate', 'move between', 'paired up with', and 'followed by', all refer to the same concept. That is, 2 schedules can be discussed with the above words when, some nurse may, or does, work the pattern of days of the first schedule, followed by the second. The order of the schedules is important, since they are worked in sequence.

schedule Z, when connected with schedule Y, does not violate a parameter either.

Suppose a schedule passed the static elimination procedure without being removed. Suppose also, that within the set of all the other schedules which passed, this particular schedule cannot, without violating a parameter, connect with any of them. This situation describes the basis for the second phase of elimination, since if a schedule cannot legitimately connect with other schedules, it can never be used in a solution. Under these conditions, it could safely be removed. This is referred to as the 'DYNAMIC ELIMINATION PROCEDURE'.

Another way to describe this concept is as follows: Imagine that each schedule which passed through static elimination, is a node. If schedule X followed by schedule Y is valid, then draw an arrow from node X to node Y. Do this for all possible pairs of nodes. If the connection is valid, then indicate it with an arrow; if the connection violates at least one parameter, then do nothing. When complete, many nodes will have both arrows in, and arrows out. There will probably be other nodes which have: only arrows in, only arrows out, or no arrows whatsoever; any one of these 3 conditions represents a schedule that cannot be used—and therefore is immediately eliminated.

Although it may appear strange that a schedule, having passed the static elimination test, must now be removed, this is quite normal. For instance, in the previous example, S(113)='0000111' was considered valid after static elimination. Note, that since it has only 3

working days in it, whatever other schedules it connects with, it must contain the remaining 7 working days to make a total of 10 days worked in 2 weeks. Obviously that other schedule must be S(128)='1111111', but it cannot since both weekends are then worked. Schedule S(113) is clearly not usable.

To simplify the detection of invalid schedules, it is convenient to construct an incidence matrix; each element of the matrix is a flag indicating the acceptability, or non-acceptability, of movement. If element (i,j)[6] is equal to 'Y', this implies that schedule (i) connected with schedule (j), is a valid connection; if the element is instead equal to '-', the connection is invalid. Although ones and zeros are typically used when working with incidence matrices, a 'Y' is used in place of a '1', and a '-' instead of a '0'. This is to avoid interpreting the '1' as the number of nurses who make the movement, as opposed to mere acceptability of movement. In other words, if element (i,j) is set to 'Y', this is analogous to stating that a nurse may work the (i)th schedule one week, and the (j)th schedule the next, without violating a parameter.

From the example, there were 19 schedules after static elimination. Figure 3 is the resulting 19x19 incidence matrix. Note that out of 361 possible connections, only 13 did not violate a parameter.

---

[6] The 'i' represents the row number of the matrix, and the 'j' represents the column. These numbers should not be confused with actual schedules numbers, since the range of 'i', and 'j', will be the number of schedules remaining after STATIC elimination—not necessarily 128.

| | S8 | S15 | S16 | S26 | S29 | S31 | S39 | S128 | S100 | S104 | S113 | S132 | S140 | S146 | S161 | S212 | S225 | S227 | S228 | MTWTFSS |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| S(8) | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | Y | 1110000 |
| S(15) | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | Y | 0111000 |
| S(16) | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | Y | - | 1111000 |
| S(26) | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | Y | 1001100 |
| S(29) | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | Y | 0011100 |
| S(31) | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | Y | - | 0111100 |
| S(32) | - | - | - | - | - | - | - | - | - | - | - | Y | - | - | Y | - | - | Y | - | 1111100 |
| S(98) | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | 1000011 |
| S(100) | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | 1100011 |
| S(103) | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | 0110011 |
| S(104) | - | - | - | - | - | - | Y | - | - | - | - | - | - | - | - | - | - | - | - | 1110011 |
| S(113) | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | 0000111 |
| S(114) | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | 1000111 |
| S(116) | - | - | - | - | - | - | Y | - | - | - | - | - | - | - | - | - | - | - | - | 1100111 |
| S(121) | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | 0001111 |
| S(122) | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | 1001111 |
| S(125) | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | 0011111 |
| S(127) | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | 0111111 |
| S(128) | - | - | - | Y | Y | - | - | - | - | - | - | - | - | - | - | - | - | - | - | 1111111 |

Figure 3: Incidence Matrix (initial)

Note that many of the rows do not contain at least one 'Y'; this implies that no schedule can follow that row's schedule. The rows that have no 'Y's are: S(98), S(100), S(103), S(113), S(114), S(121), S(122), S(125), and S(127). Each of these rows, or schedules, can be removed from all future consideration. Its corresponding column can also be removed at the same time.

The columns that do not contain at least one 'Y' relate to those schedules that can never be entered, since no other schedule can precede it. These columns are: S(8), S(15), S(16), S(31), S(98),

S(100), S(103), S(113), S(114), S(121), and S(122). Again, since these schedules are unusable, both the columns and the corresponding rows can be removed.

After all useless rows and columns are removed, the incidence

| | S 2 6 | S 2 9 | S 3 2 | S 1 4 | S 1 6 | S 1 8 | MTWTFSS |
|---|---|---|---|---|---|---|---|
| S(26) | - | - | - | - | - | Y | 1001100 |
| S(29) | - | - | - | - | - | Y | 0011100 |
| S(32) | - | - | - | Y | Y | - | 1111100 |
| S(104) | - | - | Y | - | - | - | 1110011 |
| S(116) | - | - | Y | - | - | - | 1100111 |
| S(128) | Y | Y | - | - | - | - | 1111111 |

Figure 4: Incidence Matrix (final)

matrix is considerably smaller (see figure 4). Originally the number of possibly schedules went from 128 down to 19; now the 19 were reduced further, down to only 6. It should be stressed that these 6 schedules would be the only 6 possibles in any solution of this particular problem—independent of the approach taken. This suggests that a solution produced by another author's method, would necessarily utilize some, or all, of these 6 schedules. This implies also, that these other approaches attempt to solve a much larger problem than is actually necessary.

## 7.1 ALGORITHM

Below is the DYNAMIC ELIMINATION ALGORITHM.

1. Construct a matrix of all MxM possible combinations of schedules paired with themselves. Each pair represents the potential movement from the first schedule to the next, and produces a 'pair pattern'. The first week of the pair will always correspond to the row, and the second to the column. Thus the '(1)' and '(2)' suffixes refer to the first and second half of the pair. Go to the first pair pattern, (row 1)-(column 1).

2. Examine this pair pattern. If it tests false for at least one of the following conditions (a-f), then insert a '-' into the corresponding matrix position. If, on the other hand, it tests true for all conditions, then insert a 'Y' into the same location.[7]

   a) WSSUN(1) + WSMON(2) ≤ WSMAX
   The sum of the consecutive work days from the end of the first week, and the consecutive work days from the start of the second week, should be less than, or equal to, the maximum required work stretch.

   b) WSSUN(1) + WSMON(2) ≥ WSMIN
   Similarly, the same sum, from (a) above, should be greater than, or equal to, the minimum required work stretch.

   c) OSSUN(1) + OSMON(2) ≥ OSMIN
   The sum of the consecutive days off from the end of the first week, and the consecutive days off from the start of the second week, should be greater than, or equal to, the minimum required days off.

   d) DWTOT(1) + DWTOT(2) ≤ DWMAX
   The total work days in the first week, plus the total work days in the second week, should be less than, or equal to, the maximum required work days per period.

   e) DWTOT(1) + DWTOT(2) ≥ DWMIN
   Similarly, this same sum, from (d) above, should be greater than, or equal to, the minimum required work days per period.

---

[7] To reduce computer operations, if a row is found to contain all zeros, any subsequent elements in its corresponding column do not get tested later, since, the row-column pair will eventually be removed anyway.

f) (S(1)<33 & S(2)>32) OR (S(1)>32 & S(2)<33)
Either the first week has the weekend worked, while the second does not, OR the second week has the weekend worked, while the first does not. If the schedule number is greater than 32, the weekend is worked; otherwise, the weekend is off. Note that this test is applicable only when WEOFF=1 and PERIOD=2.

3. If there are no more pairs to check, then continue at Step #4. Otherwise, choose the next pair pattern and return to Step #2.

4. Examine each row of the filled matrix. If a row does not contain at least one 'Y', then cross out that row, and its corresponding column; thereby removing that schedule from the possible list of solutions.

5. Examine each column of the filled matrix. If a column does not contain at least one 'Y', then cross out that column, and its corresponding row; thereby removing that schedule from the possible list of solutions.

6. Repeat Steps #4 and #5 until all rows and columns contain at least one '1'. Redraw the matrix, omitting the crossed out rows and columns.

## Chapter VIII

## INCIDENCE, VARIABLE, AND SOLUTION MATRICES

Although the incidence matrix shows which movements are allowable, it does not, by itself, describe how many nurses should be in each schedule, nor the movements at each week's end. The incidence matrix simply states which movements do not violate the parameters.

If every 'Y' in the incidence matrix were replaced with a unique variable, that variable could represent the actual number of nurses who move between the two associated schedules. For instance, if the 'Y' in the first column, were instead a variable t1, the value of t1 would represent the number of nurses who complete S(16) at the end of one week, and begin the next week with S(127). Thus if each 'Y' were replaced with a unique variable, the incidence matrix given in figure 5 would be the result.[8] The resulting matrix is referred to as the 'Variable Matrix'.

Each variable, in the variable matrix, represents the number of nurses who move between the schedule of its row, and the schedule of its column. Since the total number of nurses who leave a particular schedule is the sum of all movements out of that schedule, the number of nurses residing in each schedule is equal to the sum of that row's variable. And, the number of nurses who enter a particular schedule, is equal to the sum of the variables in that schedule's column. In

[8] The naming of the 't' variables is strictly arbitrary.

Incidence Matrix

| | S 26 | S 29 | S 32 | S 104 | S 116 | S 128 |
|---|---|---|---|---|---|---|
| S(26) | - | - | - | - | - | Y |
| S(29) | - | - | - | - | - | Y |
| S(32) | - | - | - | Y | Y | - |
| S(104) | - | - | Y | - | - | - |
| S(116) | - | - | Y | - | - | - |
| S(128) | Y | Y | - | - | - | - |

Variable Matrix

| | S 26 | S 29 | S 32 | S 104 | S 116 | S 128 | MTWTFSS |
|---|---|---|---|---|---|---|---|
| S(26) | - | - | - | - | - | t1 | 1001100 |
| S(29) | - | - | - | - | - | t2 | 0011100 |
| S(32) | - | - | - | t3 | t4 | - | 1111100 |
| S(104) | - | - | t5 | - | - | - | 1110011 |
| S(116) | - | - | t6 | - | - | - | 1100111 |
| S(128) | t7 | t8 | - | - | - | - | 1111111 |

Figure 5: Incidence Matrix and Variable Matrix

the example, the number of nurses who are just leaving, and just entering each schedule, is as follows:

| Schedule | # Leaving | # Entering |
|---|---|---|
| S(26) | t1 | t7 |
| S(29) | t2 | t8 |
| S(32) | t3 + t4 | t5 + t6 |
| S(104) | t5 | t3 |
| S(116) | t6 | t4 |
| S(128) | t7 + t8 | t1 + t2 |

Since the variable matrix describes the movements at the end of one week, there must be one variable matrix representing the end of each, and every, week. Each time, the number of nurses who enter any schedule, must be equal to the number of nurses who later leave that schedule. To avoid the large quantity of individual variable

matrices, and also to fulfil one of the original objectives (that of having a repeatable solution), only one variable matrix is constructed. A solved variable matrix is thus a solution matrix for all weeks. In other words, let the solution to the 't' variables be the solution used at the end of all weeks. This leads directly to the fact that, the number of nurses leaving each schedule, must equal the number entering. The following equations are the result:

| Schedule | # Leaving = # Entering |
|---|---|
| S(26) | t1 = t7 |
| S(29) | t2 = t8 |
| S(32) | t3 + t4 = t5 + t6 |
| S(104) | t5 = t3 |
| S(116) | t6 = t4 |
| S(128) | t7 + t8 = t1 + t2 |

Since the number who leave is equal to the number entering, the sum of the variables in each row, must equal the sum of the corresponding column. Therefore, there will always be a constant number of nurses in each schedule.

Once the variables t1 through t8 take on values, the variable matrix is called a 'solution matrix'. However, the process used to find the best values is described in the next chapter. Figure 6 shows a variable matrix and a related solution matrix (using some hypothetical values). Thus there are now 3 types of matrices:

1. The incidence matrix which displays allowable movements between schedules.
2. The variable matrix which displays the unknown movements. And,

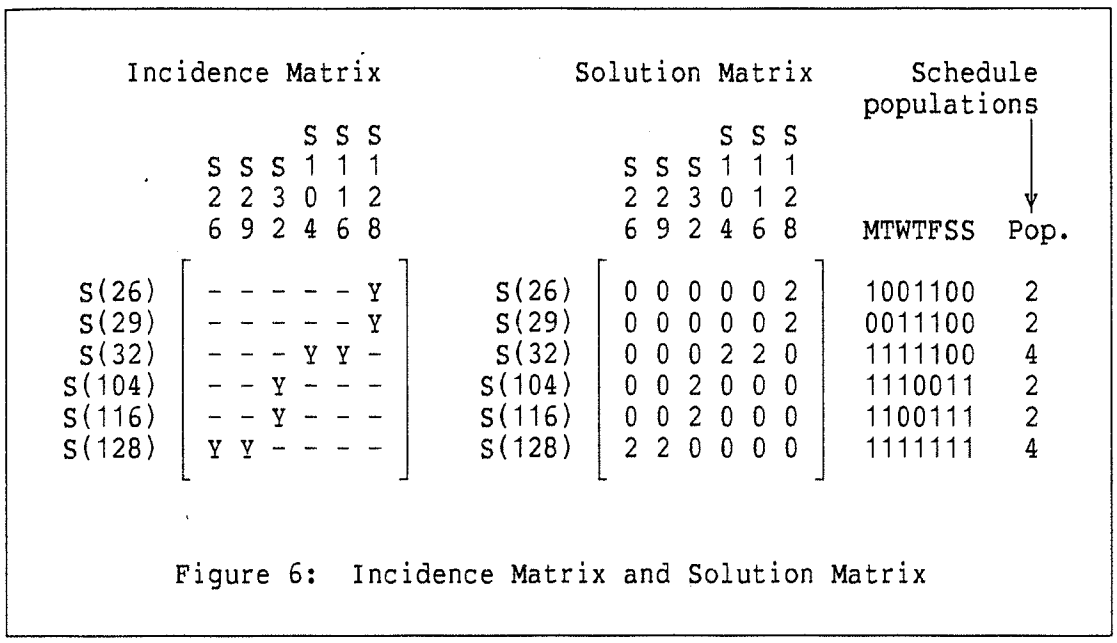


Figure 6: Incidence Matrix and Solution Matrix

3. The solution matrix that shows exactly how many nurses will move, along with the schedule populations.

Although the means of finding this final matrix have not yet been provided, its features can be examining here. From the solution matrix, it is visible that there will always be 2 nurses working S(26), S(29), S(104), and S(116); while 4 nurses work S(32) and S(128). Using this information, the number of nurses who work each day of the week can be easily calculated. For example, to determine the coverage for a Monday, do the following:

> First note that the only schedules which provide a working day on a Monday are: S(26), S(32), S(104), S(116), and S(128). Since there are: 2, 4, 2, 2, and 4 nurses working these schedules respectively, there will be a total of 14 nurses on a Monday.

Similar calculations can be executed for the rest of the week. The obtained coverage for each, and every, week will be:

| Day | Mon | Tue | Wed | Thr | Fri | Sat | Sun |
|---|---|---|---|---|---|---|---|
| Coverage | 14 | 12 | 14 | 12 | 12 | 8 | 8 |

At the end of each and every week, the following movements take place:

the 2 nurses from S(26), begin S(128)
the 2 nurses from S(29), begin S(128)
the 4 nurses from S(32), split up and begin S(104) & S(116)
the 2 nurses from S(104), begin S(32)
the 2 nurses from S(116), begin S(32)
the 4 nurses from S(128), split up and begin S(26) & S(29)

Now, the problem is, "how are these 't' variables determined, or, how can the solution matrix be found?" Several possible linear program approaches have been examined. Each one utilized the incidence matrix, the schedule patterns, and the problem requirements, to generate an optimal set of 't' values. Only three methods are examined, although it is likely there may exist many other formulations that use the incidence matrix perspective.

Whatever solution method used, it must only produce solutions with a certain characteristic. That is, all row sums must equal their respective column sums. Within this restriction, the solution method can perform according to almost any objective function. Without this restriction, individual solutions would then be required for the end of each week, since the solution matrix would not be repeatable. This would be very costly to accomplish, and fortunately is not necessary.

In many ways, the variable matrix now behaves similarly to a Markov process. The schedule populations could be analogous to steady state probabilities, and the variable matrix likened to the Markov transition matrix. The major deviation is the need to discuss movements using integer values, rather than fractional probabilities. Also, concepts such as first passage time do not readily relate, since actual movements from one schedule to another are based upon strict deterministic rules—not chance. However, the different states (schedules) do exhibit a strict period behaviour. See Chapter XII for a discussion of the periodic behaviour of solutions.

## Chapter IX

## SIMPLEX SOLUTION (TYPE 1)

Once the following information is available, the formulation presented in this chapter can provide the answer to the values of the 't' variables. This answer is the actual solution matrix. The required information is:

1. The variable matrix.
2. The corresponding weekly schedules. And,
3. The required daily coverages.

Since the 't' variables represent the number of nurses who move through each valid cell, the number of nurses in any one schedule is the sum of the variables in the row. Thus the sum of all 't' variables equals the total workforce of nurses. One objective function could be the minimization of this sum.

The first set of necessary constraints are the restriction that, the actual daily coverages are greater than, or equal to, the required daily coverages. To calculate the actual daily coverage for any specific day of the week, simply find the sum of any variables that make up all schedules which works that specific day. This sum must meet the required daily coverage for that same day. Since there are 7 actual and required coverages, one for each day of the week, there will clearly be 7 constraints of this type.

A second set of constraint equations are the 'steady state', or balance restrictions. These basically specify that the number of nurses who move out of each schedule, must equal the number who enter. Again, the number who leave each schedule, is the sum of the variables in the corresponding row; the number who enter each schedule, is the sum of the variables in the corresponding column. Therefore, the number of equations will equal the number of remaining schedules (that is, the dimension of the variable matrix).

A final restriction is that all variables should be greater than, or equal to, zero. And, the values of all variables must be whole numbers—fractional nurses are not permitted. This is obvious, since the problem is a real-world application. The described linear program is constructed below, using the same example problem. The objective function used here is the minimization of the sum of the 't' values:

$$\text{Min } t1 + t2 + t3 + t4 + t5 + t6 + t7 + t8 = \text{NURSES}$$

Ensure that the obtained coverages are greater than, or equal to, the required daily coverages:

$$\begin{aligned}
t1 \quad &+ t3 + t4 + t5 + t6 + t7 + t8 \geq 6 \\
&t3 + t4 + t5 + t6 + t7 + t8 \geq 6 \\
t2 \; &+ t3 + t4 + t5 \quad + t7 + t8 \geq 6 \\
t1 + t2 \; &+ t3 + t4 \quad + t7 + t8 \geq 6 \\
t1 + t2 \; &+ t3 + t4 \quad + t6 + t7 + t8 \geq 6 \\
&t5 + t6 + t7 + t8 \geq 4 \\
&t5 + t6 + t7 + t8 \geq 4
\end{aligned}$$

Make sure that the number of nurses in each schedule remains constant. That is, the number who leave, subtract the number who enter, must equal zero:

$$
\begin{aligned}
t1 - t7 &= 0 \\
t2 - t8 &= 0 \\
t3 + t4 - t5 - t6 &= 0 \\
t5 - t3 &= 0 \\
t6 - t4 &= 0 \\
t7 + t8 - t1 - t2 &= 0
\end{aligned}
$$

Finally, all variables must be whole numbers:

$$\text{all } t \geq 0 \text{ and integer}$$

Together this brings the final LP to the formulation given in figure 7.

This solution to this LP was obtained after only 8 Simplex iterations. The best non-zero values are: t2=2, t4=2, t6=2, and t8=2. And, a total of 8 nurses are needed to meet the requirements. Note that because t1, t3, t5, and t7 are zero, schedules S(26) and S(128) are never utilized.

Although it is certainly possible to obtain a non-integer solution, in almost all the cases examined the optimal solution was integer. Obviously, if a non-integer solution was obtained, additional steps would be required to locate the best integer solution. This

$$
\begin{aligned}
&\text{Min } t1 + t2 + t3 + t4 + t5 + t6 + t7 + t8 \\
&\text{s.t.} \\
&\quad t1 + t3 + t4 + t5 + t6 + t7 + t8 \geq 6 \\
&\quad t3 + t4 + t5 + t6 + t7 + t8 \geq 6 \\
&\quad t2 + t3 + t4 + t5 + t7 + t8 \geq 6 \\
&\quad t1 + t2 + t3 + t4 + t7 + t8 \geq 6 \\
&\quad t1 + t2 + t3 + t4 + t6 + t7 + t8 \geq 6 \\
&\quad t5 + t6 + t7 + t8 \geq 4 \\
&\quad t5 + t6 + t7 + t8 \geq 4 \\
&\quad t1 - t7 = 0 \\
&\quad t2 - t8 = 0 \\
&\quad t3 + t4 - t5 - t6 = 0 \\
&\quad -t3 + t5 = 0 \\
&\quad -t4 + t6 = 0 \\
&\quad -t1 - t2 + t7 + t8 = 0 \\
&\text{all } t \geq 0 \text{ and integer}
\end{aligned}
$$

Figure 7: Type 1 Linear Formulation (example)

particular linear program did however produce an integer solution, and the answer is incorporated into the following solution matrix.

| | S 2 6 | S 2 9 | S 3 2 | S 1 4 | S 1 6 | S 1 8 | MTWTFSS | Pop. |
|---|---|---|---|---|---|---|---|---|
| S(26) | 0 | 0 | 0 | 0 | 0 | 0 | 1001100 | 2 |
| S(29) | 0 | 0 | 0 | 0 | 0 | 2 | 0011100 | 0 |
| S(32) | 0 | 0 | 0 | 0 | 2 | 0 | 1111100 | 2 |
| S(104) | 0 | 0 | 0 | 0 | 0 | 0 | 1110011 | 0 |
| S(116) | 0 | 0 | 2 | 0 | 0 | 0 | 1100111 | 2 |
| S(128) | 0 | 2 | 0 | 0 | 0 | 0 | 1111111 | 2 |

## 9.1 RELATED MATRIX ALGEBRA

Although the algebra presented is reasonably straightforward, the manipulations must be transferable into a form understood by computer. The steps taken to construct the LP, can be performed using matrix algebra--a preferable representation for computers. This complete procedure is repeated here, in matrix form.

Let the incidence matrix be called INCID; note that the 'Y's are now replaced with the numerals '1' through '8'. This is done so that later, each variable can be uniquely identified. In fact, these numerals relate directly to the 't' variable numbers. This matrix will always be square. Let the matrix of relevant schedules be called WEEK. This matrix is obtained directly from the appending list of schedules, which appeared beside each of the previously displayed incidence matrices. This matrix will always have 7 columns. Let the column matrix of 't' variables be called 'T'. Here, each entry represents the number of nurses who move through the incidence matrix element associated with that entry. Let the matrix of minimum daily requirements be called 'R'. This matrix will always be a 7 by 1 matrix; where the top element is the minimum daily requirement for a Monday. Similarly for remainder of the week. Let the matrix of actual daily coverages be called 'A'. Each element represents the actual number of nurses scheduled for each day of the week. Again, this matrix will always have 7 rows and 1 column.

The problem matrices:

$$\text{INCID} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 3 & 4 & 0 \\ 0 & 0 & 5 & 0 & 0 & 0 \\ 0 & 0 & 6 & 0 & 0 & 0 \\ 7 & 8 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad \text{WEEK} = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

$$\text{T} = \begin{bmatrix} t1 \\ t2 \\ t3 \\ t4 \\ t5 \\ t6 \\ t7 \\ t8 \end{bmatrix} \qquad \text{R} = \begin{bmatrix} \text{RMON} \\ \text{RTUE} \\ \text{RWED} \\ \text{RTHR} \\ \text{RFRI} \\ \text{RSAT} \\ \text{RSUN} \end{bmatrix} \qquad \text{A} = \begin{bmatrix} \text{AMON} \\ \text{ATUE} \\ \text{AWED} \\ \text{ATHR} \\ \text{AFRI} \\ \text{ASAT} \\ \text{ASUN} \end{bmatrix}$$

Let EXTR and EXTC represent the 'EXTension of Rows matrix' and the 'EXTension of Columns matrix'. The definition would be: "If INCID(i,j)=k, then EXTR(i,k)=1 and EXTC(k,j)=1 (provided k is not equal to zero); otherwise, all elements in EXTR and EXTC are zero."

$$EXTR = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \qquad EXTC = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$EXT = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 1 & -1 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \times \begin{bmatrix} t1 \\ t2 \\ t3 \\ t4 \\ t5 \\ t6 \\ t7 \\ t8 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

The relevant objective function is actually the sum of the T vector values. This is represented here by the expression: $\Sigma$ T. The two sets of constraint equations are; (1) that the actual daily coverage is greater that, or equal to, the required coverages, and (2) that the row sums must equal the column sums. The actual daily coverage is the vector A, and the required coverage vector is R; therefore $A \geq R$ must hold true. EXTR • T represents a vector of nurses who work each schedule in the current week, and EXTC' • T represents a vector of nurses who will work each schedule in the next week. These vectors should be equal to each other; or EXTR • T = EXTC' • T, or ( EXTR - EXTC' ) • T = $\underline{0}$. Letting EXT = EXTR - EXTC', produces EXT • T = $\underline{0}$. The matrix LP is therefore:

$$\text{Min } \Sigma\ T$$

s.t.

$A \geq R$ (actual ≥ required coverage)

$EXT \bullet T = 0$ (row sum - column sum = 0)

all $T \geq 0$ and integer

However, the values of the actual coverages can also be obtained through the multiplication of the three matrices; WEEK', EXTR, and T. That is• A = WEEK' • EXTR • T, or in the example:

$$\begin{bmatrix} \text{AMON} \\ \text{ATUE} \\ \text{AWED} \\ \text{ATHR} \\ \text{AFRI} \\ \text{ASAT} \\ \text{ASUN} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \times \begin{bmatrix} t1 \\ t2 \\ t3 \\ t4 \\ t5 \\ t6 \\ t7 \\ t8 \end{bmatrix}$$

Once A is replaced, the final linear program, in matrix variable terms, is given in figure 8.

$$\text{Min } \Sigma\, T$$

s.t.

$$\text{WEEK}' \bullet \text{EXTR} \bullet T \geq R$$

$$\text{EXT} \bullet T = 0$$

all $T \geq 0$ and integer

Figure 8: Type 1 Linear Formulation

## Chapter X

## SIMPLEX SOLUTION (TYPE 2)

A specific modification of the previous formulation, can drastically reduce the number of variables. However to do this, certain preliminary steps must be taken. These steps require many calculations themselves—in fact, more overall calculations than are ultimately saved. The only justification for this modification is, that the calculations need to be performed only once. Thus, if several problems need solving, and they only differ in their daily requirements, this formulation can save many hand, or computer, operations. The approach recognizes that the balance restrictions create a great deal of inter-dependence between variables. Once the dependent variables are removed, the number of variables and equations are both reduced.

In the previous example, the sum of the first row is t1, and the sum of the first column is t7. If schedule populations are to be balanced, then t1 must always equal t7. The variable t1 is therefore dependent on t7, and the relationship is, t1 = t7. Thus, where ever t1 occurs in the LP, t7 could be used instead.

Gaussian reduction, performed on the balance equations, would reveal all such dependencies. The number of equations manipulated by Gaussian elimination is exactly the dimension of the incidence matrix (in the previous example the number is 6). Using the previous example the following dependencies were obtained:

| Before Gaussian Reduction | After |
|---|---|
| $t1 - t7 = 0$ | $t1 = t7$ |
| $t2 - t8 = 0$ | $t2 = t8$ |
| $t3 + t4 - t5 - t6 = 0$ | $t3 = t5$ |
| $-t3 + t5 = 0$ | $t4 = t6$ |
| $-t4 + t6 = 0$ | $t5 = t5$ |
| $-t1 - t2 + t7 + t8 = 0$ | $t6 = t6$ |
| | $t7 = t7$ |
| | $t8 = t8$ |

Thus, every occurance of a t1, t2, t3, or t4, could be replaced by a t7, t8, t5, and t6, respectively. These replacements would not alter the problem in any respect. A solution to an LP, with dependent variables replaced, would be identical to the original LP. Incorporating this transformation into the previous examples' LP

$$
\begin{aligned}
&\text{Min } 2t5 + 2t6 + 2t7 + 2t8 \\
&\text{s.t.} \\
&\quad 2t5 + 2t6 + 2t7 + t8 \geq 6 \\
&\quad 2t5 + 2t6 + t7 + t8 \geq 6 \\
&\quad 2t5 + t6 + t7 + 2t8 \geq 6 \\
&\quad t5 + t6 + 2t7 + 2t8 \geq 6 \\
&\quad t5 + 2t6 + 2t7 + 2t8 \geq 6 \\
&\quad t5 + t6 + t7 + t8 \geq 4 \\
&\quad t5 + t6 + t7 + t8 \geq 4 \\
&\quad t5 \geq 0 \\
&\quad t6 \geq 0 \\
&\quad t7 \geq 0 \\
&\quad t8 \geq 0 \\
&\text{all } t \geq 0 \text{ and integer}
\end{aligned}
$$

Figure 9: Type 2 Linear Formulation (example)

produced the formulation given in figure 9.

Note that the LP in the previous chapter had 13 equations and 8 variables, this LP has only 11 equations[9] and 4 variables; although the last 4 constraints can be safely omitted here. This is clearly a significant improvement.

Once again the LP was solved. After only 5 iterations, the solution was found. The solution found was: t6=2 and t8=2, while all other variables were equal to zero. After substituting back, it was determined that: t2=2 and t4=2. This answer is the same one obtained in the previous chapter.

Sometimes during this reduction of variables, a curious anomaly appears. The final LP may contain fewer variables than the actual number of independent variables. This occurs because some independent variables—although they are still independent—drop out of the constraints and do not affect the solved daily coverages. In fact, they only control which variables are non-zero in the final solution matrix. They can be set to any value, as long as all the remaining variables do not go negative. Since multiple solutions are the rule for these LPs, this occurance is largely ignored, and the variables are simply left to values of 0.

---

[9] The 4 additional constraints ensure that the dependent variables remain non-negative. These constraints should, however, be written in terms of the independent variables—hence the number of variables do not increase. To keep $t1 \geq 0$, use $t7 \geq 0$ instead. Note that in this particular case, these extra constraints are completely redundant. Only when the dependent variable relationships contain at least one negative coefficient, do these extra constraints become potentially necessary.

## 10.1 DUAL SIMPLEX APPLICATION

Because of the structure of the LP, Dual Simplex is an appropriate method of solution. It is useful to note, that the Dual Simplex method allows the addition of new constraints to the final tableau; without forcing a complete restarting of the iterations. In the context of these linear programs, the 'all t≥0' constraint set is built into the Simplex method, except for the strictly dependent variables. Although the dependent variables remained positive in the current example, in other examples they may not. It has been observed for some large problems, that the dependent variables (if not indirectly forced to be positive), can become negative. Rather than include these additional (and likely useless) constraints each time an LP of this type is solved, the following steps are suggested instead.

Using only the constraint 'all independent t≥0', solve the LP. If, after substituting back to derive the dependent variables, all are positive, then the answer is perfectly acceptable. If, on the other hand, some variables are negative, then a new constraint must be added to the final tableau. The constraint added is the restriction that the variable that was negative, must be greater than, or equal to, zero. Since the LP is completely in terms of the independent variables, this new constraint must be rewritten using the independent variables. Below is the smallest problem where this phenomena was observed.

```
WSMIN=2        WSMAX=5          RMON=6
OSMIN=1                         RTUE=6
DWMIN=10       DWMAX=10         RWED=6
SPLIT=0                         RTHR=6
WEOFF=1                         RFRI=6
PERIOD=2                        RSAT=4
                                RSUN=4
```

Variable Matrix

| | S28 | S30 | S31 | S112 | S120 | S124 | MTWTFSS |
|---|---|---|---|---|---|---|---|
| S(28) | 0 | 0 | 0 | t1 | t2 | t3 | 1101100 |
| S(30) | 0 | 0 | 0 | t4 | t5 | t6 | 1011100 |
| S(31) | 0 | 0 | 0 | t7 | t8 | t9 | 0111100 |
| S(112) | t10 | t11 | t12 | 0 | 0 | 0 | 1111011 |
| S(120) | t13 | t14 | t15 | 0 | 0 | 0 | 1110111 |
| S(124) | 0 | t16 | t17 | 0 | 0 | 0 | 1101111 |

After formulating the normal LP, Gaussian elimination was performed on the balance equations. The dependencies were noted, and the LP was completely rewritten in terms of only 8 independent variables. Since the balance equations were used to identify the independent variables, the balance constraints can now be omitted. Although the dependent variables are neither directly, nor indirectly, restricted to be non-negative, solve the LP anyway. This result of these manipulations is the following LP:

```
Min 2t10 + 2t11 + 2t12 + 2t13 + 2t14 + 2t15 + 2t16 + 2t17
s.t.
    2t10 + 2t11  + t12 + 2t13 + 2t14  + t15 + 2t16  + t17 ≥ 6
    2t10  + t11 + 2t12 + 2t13  + t14 + 2t15  + t16 + 2t17 ≥ 6
     t10 + 2t11 + 2t12  + t13 + 2t14 + 2t15  + t16  + t17 ≥ 6
    2t10 + 2t11 + 2t12  + t13  + t14  + t15 + 2t16 + 2t17 ≥ 6
     t10  + t11  + t12 + 2t13 + 2t14 + 2t15 + 2t16 + 2t17 ≥ 6
     t10  + t11  + t12  + t13  + t14  + t15  + t16  + t17 ≥ 4
     t10  + t11  + t12  + t13  + t14  + t15  + t16  + t17 ≥ 4
all independent t≥0 and integer (revised LP)
```

The final tableau using this revised LP is given here. Note that the normally 'right hand side' column is displayed at the left. Also, the 'objective function row' appears at top, instead of the bottom. These variations of the normal Simplex tableau, will assist the subsequent addition of more constraints and variables. Done later. Finally, the basic variables are identified by a '<1>' in their respective columns.

| RHS | t 5 | t 6 | t 8 | t 9 | t 10 | t 11 | t 12 | t 13 | t 14 | t 15 | t 16 | t 17 | s 1 | s 2 | s 3 | s 4 | s 5 | s 6 | s 7 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | −2 |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | <1> | 1 | 0 | 0 | 1 | 0 | 0 | 0 | −2 |
| 0 | 0 | 0 | 0 | 0 | −1 | 0 | 0 | −1 | 0 | 0 | 0 | 0 | 1 | <1> | 0 | 0 | 0 | 0 | −3 |
| 0 | 0 | 0 | 0 | 0 | 0 | −1 | 0 | 0 | 0 | <1> | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | −5 |
| 2 | 0 | 0 | 0 | 0 | 0 | 1 | <1> | 0 | 0 | 0 | 0 | 0 | 0 | 0 | −1 | −1 | 0 | 0 | 3 |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | <1> | 0 | −5 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | <1> | −1 |
| 2 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | <1> | 0 | 0 | −1 | −1 | 0 | −1 | 0 | 0 | 0 | −3 |

The current basic solution is: t5=0, t6=0, t8=0, t9=0, t10=0, t11=0, t12=2, t13=0, t14=2, t15=0, t16=0, and t17=0. The non-zero independents variables are: t12=2 and t14=2. Note that although all the variables are positive or zero, when the dependent variables are

calculated, one of them is negative. These dependent variables are calculated from the independent variables. Gaussian reduction produced the equations which described these functional relationships. The equations are:

```
t1 =  t5 +t6 +t8 +t9 +t10                    -t14 -t15 -t16 -t17
t2 = -t5     -t8                      +t13 +t14 +t15
t3 =     -t6     -t9                                +t16 +t17
t4 = -t5 -t6                 +t11          +t14      +t16
t7 =          -t8 -t9              +t12          +t15      +t17

since t12=2 and t14=2, then: t1=-2, t2=2, t3=2, t4=2, and t7=2.
```

Note that the variable 't1' is negative. To fix this solution so that t1 is positive, first obtain the general equation (including a slack variable), which is equivalent to t1. Insert this equation at the bottom of the final tableau. This equation is:

```
- t5 - t6 - t8 - t9 - t10 + t14 + t15 + t16 + t17 - s8 = 0
```

The new tableau, with the equation added, is:

| RHS | t 5 | t 6 | t 8 | t 9 | t 10 | t 11 | t 12 | t 13 | t 14 | t 15 | t 16 | t 17 | s 1 | s 2 | s 3 | s 4 | s 5 | s 6 | s 7 | s 8 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | −2 | 0 |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | <1> | 1 | 0 | 0 | 1 | 0 | 0 | 0 | −2 | 0 |
| 0 | 0 | 0 | 0 | 0 | −1 | 0 | 0 | −1 | 0 | 0 | 0 | 0 | 1 | <1> | 0 | 0 | 0 | 0 | −3 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | −1 | 0 | 0 | 0 | <1> | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | −5 | 0 |
| 2 | 0 | 0 | 0 | 0 | 0 | 1 | <1> | 0 | 0 | 0 | 0 | 0 | 0 | 0 | −1 | −1 | 0 | 0 | 3 | 0 |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | <1> | 0 | −5 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | <1> | −1 | 0 |
| 2 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | <1> | 0 | 0 | −1 | −1 | 0 | −1 | 0 | 0 | 0 | −3 | 0 |
| 0 | −1 | −1 | −1 | −1 | −1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

Since the identity structure is lost, the first several iterations must remove the non-zero elements from the identity columns. With one further iteration the variable t5 enters. The solution obtained is optimal since the top row has remained non-negative. It is also feasible since there are no negatives in the first column. The final tableau is:

| RHS | t 5 | t 6 | t 8 | t 9 | t 10 | t 11 | t 12 | t 13 | t 14 | t 15 | t 16 | t 17 | s 1 | s 2 | s 3 | s 4 | s 5 | s 6 | s 7 | s 8 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | −2 | 0 |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | <1> | 1 | 0 | 0 | 1 | 0 | 0 | 0 | −2 | 0 |
| 0 | 0 | 0 | 0 | 0 | −1 | 0 | 0 | −1 | 0 | 0 | 0 | 0 | 1 | <1> | 0 | 0 | 0 | 0 | −3 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | −1 | 0 | 0 | 0 | <1> | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | −5 | 0 |
| 2 | 0 | 0 | 0 | 0 | 0 | 1 | <1> | 0 | 0 | 0 | 0 | 0 | 0 | 0 | −1 | −1 | 0 | 0 | 3 | 0 |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | <1> | 0 | −5 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | <1> | −1 | 0 |
| 2 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | <1> | 0 | 0 | −1 | −1 | 0 | −1 | 0 | 0 | 0 | −3 | 0 |
| 2 | <1> | 1 | 1 | 1 | 2 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 4 | −1 |

The basic solution is: t5=2, t6=0, t8=0, t9=0, t10=0, t11=0, t12=2, t13=0, t14=2, t15=0, t16=0, and t17=0. These independent variables

produce only non-negative dependent values. The final solution matrix is:

| | S 28 | S 30 | S 31 | S 112 | S 120 | S 124 | MTWTFSS | Pop. |
|---|---|---|---|---|---|---|---|---|
| S(28) | 0 | 0 | 0 | 0 | 0 | 0 | 1101100 | 0 |
| S(30) | 0 | 0 | 0 | 0 | 2 | 0 | 1011100 | 2 |
| S(31) | 0 | 0 | 0 | 2 | 0 | 0 | 0111100 | 2 |
| S(112) | 0 | 0 | 2 | 0 | 0 | 0 | 1111011 | 2 |
| S(120) | 0 | 2 | 0 | 0 | 0 | 0 | 1110111 | 2 |
| S(124) | 0 | 0 | 0 | 0 | 0 | 0 | 1101111 | 0 |

Instead of solving an LP with 17 variables and 13 constraints, this shortcut has solved an LP with only 8 variables and 7 constraints. The first formulation needed 19 iterations to obtain the optimal solution, while the second needed only 10 (including the 4 extra iterations).

## Chapter XI

## SIMPLEX SOLUTION (TYPE 3)

Til now, it has been assumed that all scheduling problems have had certain similar characteristics: That is, that the number of nurses were not fixed, and the daily requirements need only be satisfied with the fewest possible. Although this situation may be the case in many hospitals, it may also be that the number of nurses cannot, under any circumstances, be altered. It would be difficult to imagine that a union would concede that a mathematically optimal solution is important enough to condone layoffs.

Another problem to solve may be the maximization of daily coverages, while maintaining a fixed number of nurses. Certainly this would avoid solutions that advocate workforce changes. However, a problem arises when this becomes the sole objective function. Since there are 7 daily requirements per week, how should the objective function be stated? If the average were used, a major problem may occur; the daily surplus[10] values could fluctuate wildly, even though the average itself may be very high. It is generally felt that fluctuating coverages are undesirable, since the real, somewhat random, workforce demands, may cause actual shortages. Also, a large surplus of nurses may promote lower relative productivity.

---

[10] The actual daily coverage, subtract the required daily coverage.

Another approach makes use of the surplus values themselves. For any given solution, there will be 7 surplus values; one for each day of the week. As long as the surplus values are all non-negative, the daily requirements are met. If the actual coverages do fluctuate excessively, the surplus values will also indicate this, since the maximum of the surplus values, here labelled 's', would tend to be much larger than the average. As long as 's' is minimized, the distribution of the nurses throughout the week will be as even as possible.

The variable 's' can be expressed as the smallest value whereby each of the following relationships hold true:

$$
\begin{aligned}
AMON - RMON &\leq s \\
ATUE - RTUE &\leq s \\
AWED - RWED &\leq s \\
ATHR - RTHR &\leq s \\
AFRI - RFRI &\leq s \\
ASAT - RSAT &\leq s \\
ASUN - RSUN &\leq s
\end{aligned}
$$

This is identical to stating that 's' is the maximum of the 7 surplus variables. The following LP extends from the previous chapter's formulation. Instead of minimizing the sum of the nurses, the objective function minimizes the maximum surplus variable 's'. There are exactly 8 nurses, and the daily requirements are the same as before. Note, that 7 more constraints have been added. These constraints relate the actual and required coverages, to the new variable 's'.

$$
\begin{aligned}
&\text{Min } s \\
&\text{s.t.} \\
&\qquad 2t5 + 2t6 + 2t7 + 2t8 = 8 \\
&\qquad 2t5 + 2t6 + 2t7 + t8 \geq 6 \\
&\qquad 2t5 + 2t6 + t7 + t8 \geq 6 \\
&\qquad 2t5 + t6 + t7 + 2t8 \geq 6 \\
&\qquad t5 + t6 + 2t7 + 2t8 \geq 6 \\
&\qquad t5 + 2t6 + 2t7 + 2t8 \geq 6 \\
&\qquad t5 + t6 + t7 + t8 \geq 4 \\
&\qquad t5 + t6 + t7 + t8 \geq 4 \\
&-s + 2t5 + 2t6 + 2t7 + t8 \leq 6 \\
&-s + 2t5 + 2t6 + t7 + t8 \leq 6 \\
&-s + 2t5 + t6 + 2t7 + 2t8 \leq 6 \\
&-s + t5 + t6 + 2t7 + 2t8 \leq 6 \\
&-s + t5 + 2t6 + t7 + 2t8 \leq 6 \\
&-s + t5 + t6 + t7 + t8 \leq 4 \\
&-s + t5 + t6 + t7 + t8 \leq 4 \\
&\text{all } t,\ s \geq 0 \text{ and integer}
\end{aligned}
$$

Below is the solution obtained by the type 2 formulation. The type 1 formulation could just as easily have been used here; the only key modification is the addition of the 's' variable. Note that the required coverages are met exactly, except for Friday, where there is a surplus of 2 nurses.

TYPE 2 SOLUTION:

t1=0 t2=2 t3=0 t4=2 t5=0 t6=2 t7=0 t8=2 s=2

| M | T | W | T | F | S | S | |
|---|---|---|---|---|---|---|---|
| 6 | 6 | 6 | 6 | 8 | 4 | 4 | actual |
| 6 | 6 | 6 | 6 | 6 | 4 | 4 | required |
| 0 | 0 | 0 | 0 | 2 | 0 | 0 | surplus |

Here is the solution obtained by the new, type 3, formulation. It was found after only 8 Simplex iterations. Unlike before, the largest surplus value is 1. Instead of having one day with a surplus of 2, this solution provides two days, each having a surplus of only 1. The net effect is that the actual daily coverage is now more evenly distributed.

TYPE 3 SOLUTION:

t1=1 t2=1 t3=1 t4=1 t5=1 t6=1 t7=1 t8=1 s=1

| M | T | W | T | F | S | S | |
|---|---|---|---|---|---|---|---|
| 7 | 6 | 6 | 6 | 7 | 4 | 4 | actual |
| 6 | 6 | 6 | 6 | 6 | 4 | 4 | required |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | surplus |

## Chapter XII

## PERIODIC WORK PATTERNS

Once the final solution matrix is found, all nurse movements are determined. Since the movements are restricted to a finite set of schedules, it is expected that eventually the series will repeat. In fact, from the viewpoint of the individual nurse, the schedules always repeat every so many weeks.

However, the period of this repetition is not always straightforward and easily observed. For example, the previous

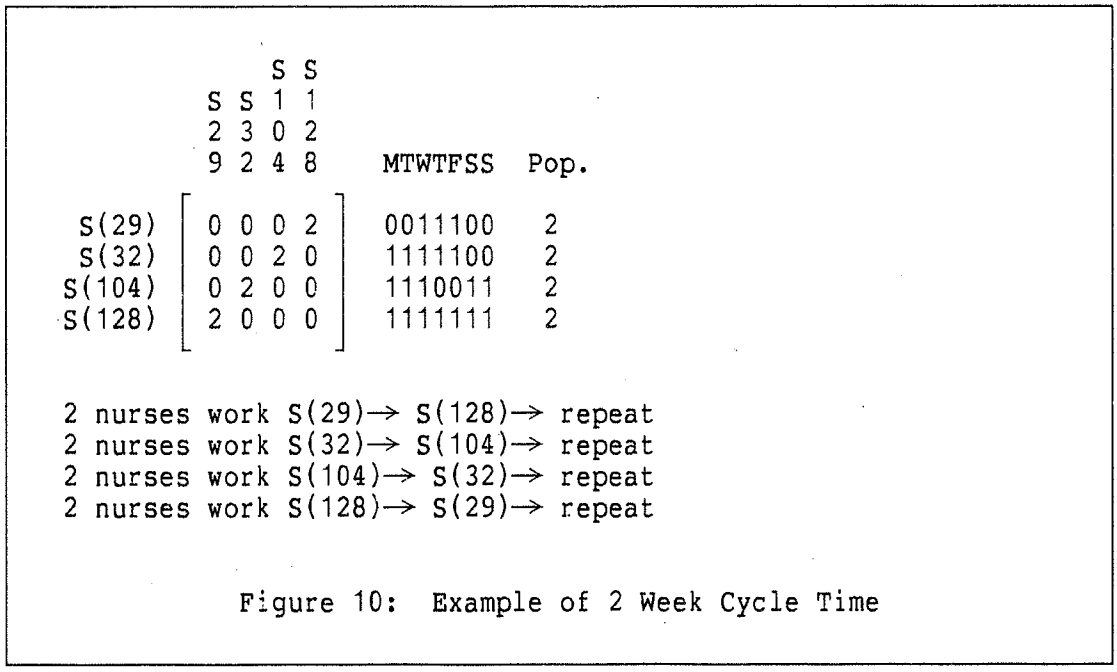


Figure 10: Example of 2 Week Cycle Time

solution matrix, given here in figure 10, displays the actual patterns of schedules that the nurses must work. Note that each nurse works 2 distinct schedules, then repeats them. Thus each of the 8 nurses experiences a 2 week cycle time. Since each nurse only works one of two different schedules, the two schedules are worked in alternating sequence.

If, however, the solution matrix were slightly modified[11] slightly, the cycle times change significantly. This altered solution matrix is

| | S2 | S3 | S01 | S02 | MTWTFSS | Pop. |
|---|---|---|---|---|---|---|
| S(29) | 0 | 0 | 2 | 0 | 0011100 | 2 |
| S(32) | 0 | 0 | 0 | 2 | 1111100 | 2 |
| S(104) | 0 | 2 | 0 | 0 | 1110011 | 2 |
| S(128) | 2 | 0 | 0 | 0 | 1111111 | 2 |

2 nurses work S(29)→ S(104)→ S(32)→ S(128)→ repeat
2 nurses work S(32)→ S(128)→ S(29)→ S(104)→ repeat
2 nurses work S(104)→ S(32)→ S(128)→ S(29)→ repeat
2 nurses work S(128)→ S(29)→ S(104)→ S(32)→ repeat

Figure 11: Example of 4 Week Cycle Time

given in figure 11. Note that this example uses the same schedules, has the same schedule populations, and yet the pattern of schedules worked is very different. Now, each nurse works all 4 schedules, and

[11] This is not a legitimate procedure. It is only done here to display a certain feature.

then rotates between them. Thus, the cycle time for each nurse is now 4 weeks. The nurses still receive every other weekend off, yet they use a greater number of unique schedules, and use each one less often.

Although there are indications that some careful manipulations could prevent this discrepancy, it is not considered here. Firstly because there are no observable problems with having unusual cycle times. Each nurse experiences simplifying cyclic schedules all the same. Secondly, the manipulations needed to correct this characteristic, may force out many (perhaps all) of the optimal solutions. Solution quality is much more important than correcting this curiosity.

In more complicated problems, there may even be several different cycle times for different groups of nurses, all within the same solution matrix. For example, half the nurse may experience a 2 week cycle time, a quarter may experience a 4 week cycle time, and the final quarter, an 8 week cycle time.

## Chapter XIII

## MULTIPLE SHIFTS

An important facet of the nurse scheduling problem is the accommodation of multiple shifts. Typically there are either 2 or 3 shifts per 24 hour day. This chapter will consider only the more difficult, 3 shifts per day problem. The techniques used, easily apply directly to the 2 shifts per day problem.

Assume that a hospital has 3 shifts and adheres to the parameters used in the sample problem. However, the hospital's daily requirements for each of the 3 shifts are as follows:

| | Mon | Tue | Wed | Thr | Fri | Sat | Sun |
|---|---|---|---|---|---|---|---|
| D=Day | 12 | 12 | 12 | 12 | 12 | 8 | 8 |
| E=Evening | 6 | 6 | 6 | 6 | 6 | 5 | 5 |
| N=Night | 4 | 4 | 4 | 4 | 4 | 4 | 4 |

To solve the complete problem, it is necessary to first solve 3 separate problems—the Day, Evening, and Night shift problems. Using the type 2 solution described in Chapter X, the corresponding solution matrices were obtained for each shift. Note that the solution matrices are smaller than the original incidence matrix. This is because, several of the 't' variables were found to be zero, causing schedules S(26) and S(104) to be unnecessary. To save room within the

figures, assume the following label substitutions: S(29)="1", S(32)="2", S(116)="3", and S(128)="4". The solution matrices for each

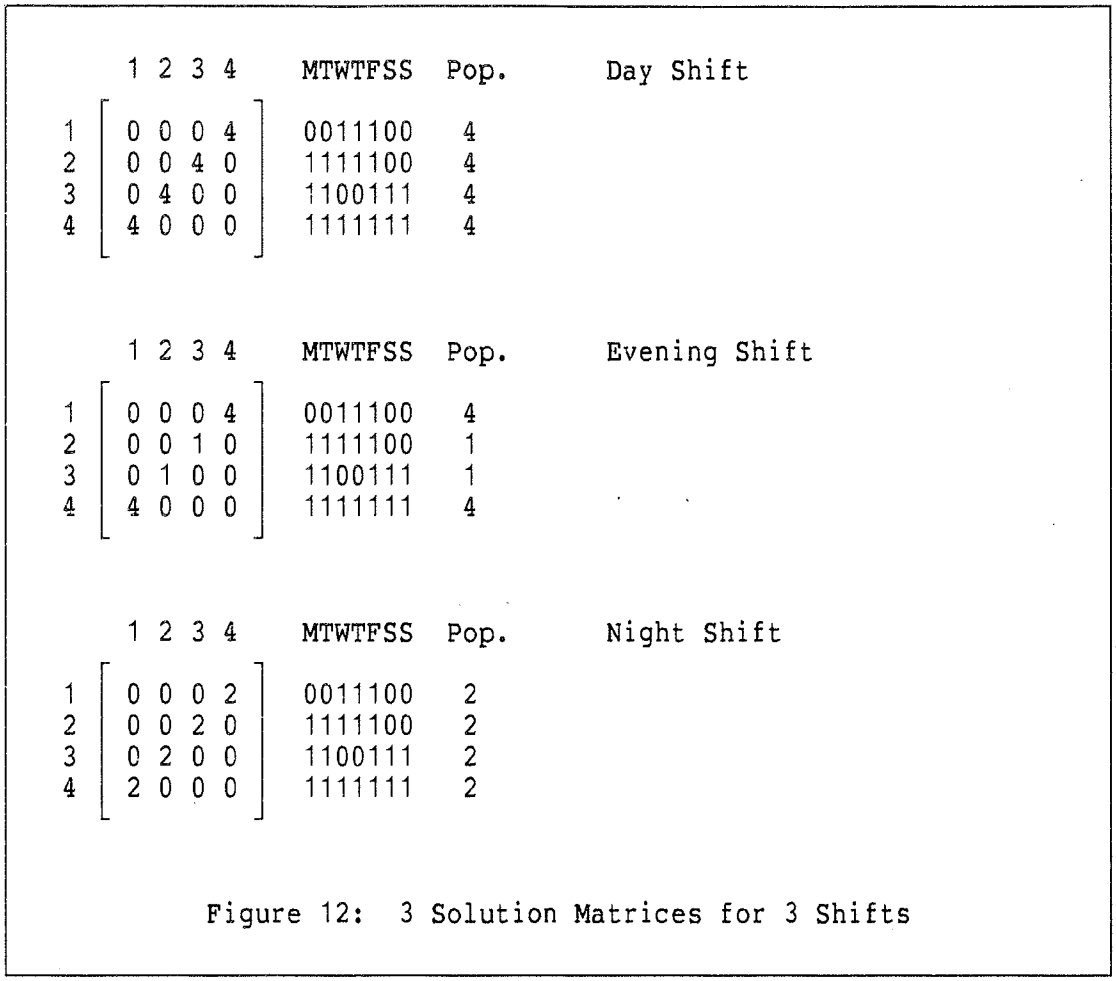


Figure 12: 3 Solution Matrices for 3 Shifts

shift, are given in figure 12.

Although this solution provides an adequate number of nurses for each shift, it does not, so far, allow nurses to move between shifts. For the moment, assume that this is an acceptable condition. To describe the complete solution, these 3 matrices can be combined into

a single, larger, solution matrix. This larger matrix is constructed by arranging the 3 matrices diagonally, and filling the excess area with zeros. This larger matrix is given in figure 13.

When discussing multiple shifts, the single shift solution matrices are referred to as 'sub-matrices'. In the current case, 9 such sub-matrices exist. They are: D-D, D-E, D-N, E-D, E-E, E-N, N-D, N-E, and N-N (for example, D-N describes any movement from a day schedule to a night schedule). Previously only the D-D, E-E, and N-N cases were considered; these movements are called 'intra-shift movements'. The 6 remaining sub-matrices D-E, D-N, E-D, E-N, N-D, and N-E relate to the 'inter-shift movements'.

To simplify matrix labelling, let the 'D', 'E', and 'N' prefixes denote the Day, Evening, and Night shift version of each particular schedule. Also, let the schedule represent which shift it is, by using the 'D', 'E', or 'N' in place of each '1' in the 7 day pattern. For example, Day shift S(29)='0011100', is the same as D1='··DDD··'.

This larger solution matrix is a correct description of the previous solution—provided, movements between shifts are not needed. However, the accommodation of shift movements is not difficult to achieve. First, it is important to realize that the population values are essential to the optimality of any solution. If they change, the solution will likely cease to be optimal. The solution will remain optimal, as long as the row and column sums remain constant.

On the other hand, the matrix elements themselves are not restricted in the same way. They may change as long as the row/column

| | D 1 | D 2 | D 3 | D 4 | E 1 | E 2 | E 3 | E 4 | N 1 | N 2 | N 3 | N 4 | MTWTFSS | Pop. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| D1 | 0 | 0 | 0 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ··DDD·· | 4 |
| D2 | 0 | 0 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | DDDDD·· | 4 |
| D3 | 0 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | DD··DDD | 4 |
| D4 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | DDDDDDD | 4 |
| E1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 4 | 0 | 0 | 0 | 0 | ··EEE·· | 4 |
| E2 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | EEEEE·· | 1 |
| E3 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | EE··EEE | 1 |
| E4 | 0 | 0 | 0 | 0 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | EEEEEEE | 4 |
| N1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | ··NNN·· | 2 |
| N2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | NNNNN·· | 2 |
| N3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | NN··NNN | 2 |
| N4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 4 | 0 | 0 | 0 | NNNNNNN | 2 |

Figure 13: Solution Matrix for 3 Shifts (original)

sums do not become altered. If just an element is changed, then the number of nurses who move through that cell will change also. For the moment, ignore how changes could be performed and examine the feasibility of work patterns. If the element at position (D3,E4) were changed from 0 to 4, then 4 nurses must work the pattern 'DD··DDDEEEEEEE'. Not only does the nurse lose every other weekend off, he/she would also have to change from a Sunday-Day, to a Monday-Night, without a day off in between.

Thus there are 2 distinct classes of violations. The first class includes all violations of the previous parameters. Thus, disregarding shift changes, the pattern of days on and days off must conform to all previous restrictions. Invalid movements in this class

are easy to determine, since they exist in the identical pattern as the '-'s of the single incidence matrices. In other words, any previously disallowed movement is mapped onto the inter-shift sub-matrices, where the invalid movements are noted.

The second class are those which violate some reasonable minimum days-off value between shifts. In fact, this value can be considered a new problem parameter. The question that must be answered is; "when a nurse changes shifts, how many days off should be provided?" Say that it is felt that 2 days off are sufficient. The inter-shift sub-matrix cells can be examined, and a count made of the number of days a nurse would have off between each pair of schedules. If this count is less than 2, then that cell can be excluded from further consideration. Similarly, all other inter-shift cells that do not supply the minimum number of days off, can be removed. Note, only the inter-shift sub-matrices should be checked in this manner; intra-shift work patterns may allow work stretches across weeks. See the insert in figure 14, for the matrix of inter-shift days off.

Let all cells which are beyond consideration, for either class, have the '0' replaced with a '-'. Any cell that does not contain a '-', is hereafter referred to as a 'usable' or 'valid' cell. The previous solution matrix, with disallowed cells, is given in figure 14. Now, any manipulation of the matrix cells is valid, as long as: (1) it maintains the row/column sums, and (2) it only changes the usable, or valid, cells.

| | D 1 | D 2 | D 3 | D 4 | E 1 | E 2 | E 3 | E 4 | N 1 | N 2 | N 3 | N 4 | MTWTFSS | Pop. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| D1 | - | - | - | 4 | - | - | - | 0 | - | - | - | 0 | ··DDD·· | 4 |
| D2 | - | - | 4 | - | - | - | 0 | - | - | - | 0 | - | DDDDD·· | 4 |
| D3 | - | 4 | - | - | - | - | - | - | - | - | - | - | DD··DDD | 4 |
| D4 | 4 | - | - | - | 0 | - | - | - | 0 | - | - | - | DDDDDDD | 4 |
| E1 | - | - | - | 0 | - | - | - | 4 | - | - | - | 0 | ··EEE·· | 4 |
| E2 | - | - | 0 | - | - | - | 1 | - | - | - | 0 | - | EEEEE·· | 1 |
| E3 | - | - | - | - | - | 1 | - | - | - | - | - | - | EE··EEE | 1 |
| E4 | 0 | - | - | - | 4 | - | - | - | 0 | - | - | - | EEEEEEE | 4 |
| N1 | - | - | - | 0 | - | - | - | 0 | - | - | - | 2 | ··NNN·· | 2 |
| N2 | - | - | 0 | - | - | - | 0 | - | - | - | 2 | - | NNNNN·· | 2 |
| N3 | - | - | - | - | - | - | - | - | - | 2 | - | - | NN··NNN | 2 |
| N4 | 0 | - | - | - | 0 | - | - | - | 2 | - | - | - | NNNNNNN | 2 |

| | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | - | - | - | 3 |
| 2 | - | - | 2 | - |
| 3 | - | 0 | - | - |
| 4 | 2 | - | - | - |

Figure 14: Solution Matrix for 3 Shifts (initial tableau)

In fact, the transportation algorithm is ideally suited to performing these manipulations. The transportation cell variables can be represented by the movements of nurses between schedules. The transportation costs can utilize the following 3 values: infinity, some positive constant, and zero. These costs are based directly upon the type of cell. The 3 cell types are:

1. Invalid, or unusable, cells; those with a '-' in them. Let these cells have a cost of infinity, since they should never be used.

2. Valid intra-shift cells; those currently holding a non-zero value. Let these cells have some nominal cost, say 1. Thus solutions which comprise few intra-shift movements are preferred to those which have many.

3. Valid inter-shift cells; those currently holding a zero value. Let these cells have a cost of zero. Solutions which contain many inter-shift movements are therefore preferred.

This approach tends to increase the numbers of movements between shifts. It may be that some other criteria is more desirable in certain situations. Subsequent research could examine other strategies and algorithms relevant to inter-shift nurse movements, and its related work patterns.

In short, the current solution matrix becomes the initial tableau for the transportation method. The cell costs are inserted according to the 3 types discussed. After which, iterations are performed until the cost is minimized. When the optimal tableau is obtained, it is simply converted back into a solution matrix. This solution matrix will still be optimal (because the row/column sums are unchanged), and will now provide both intra-shift and inter-shift nurse movements.

Although the iterations tend to be the degenerate type, and the initial tableau is usually a non-basic solution, the application of the transportation algorithm is otherwise straightforward and produces quick results. The initial tableau is given in figure 15, while the final tableau is in figure 16. Whereas the initial tableau had an overall cost of 34, the final tableau has a cost of 8. But the final tableau, unlike the initial one, provides for the all important, inter-shift nurse movements. The final tableau can be converted back into the solution matrix shown in figure 17.

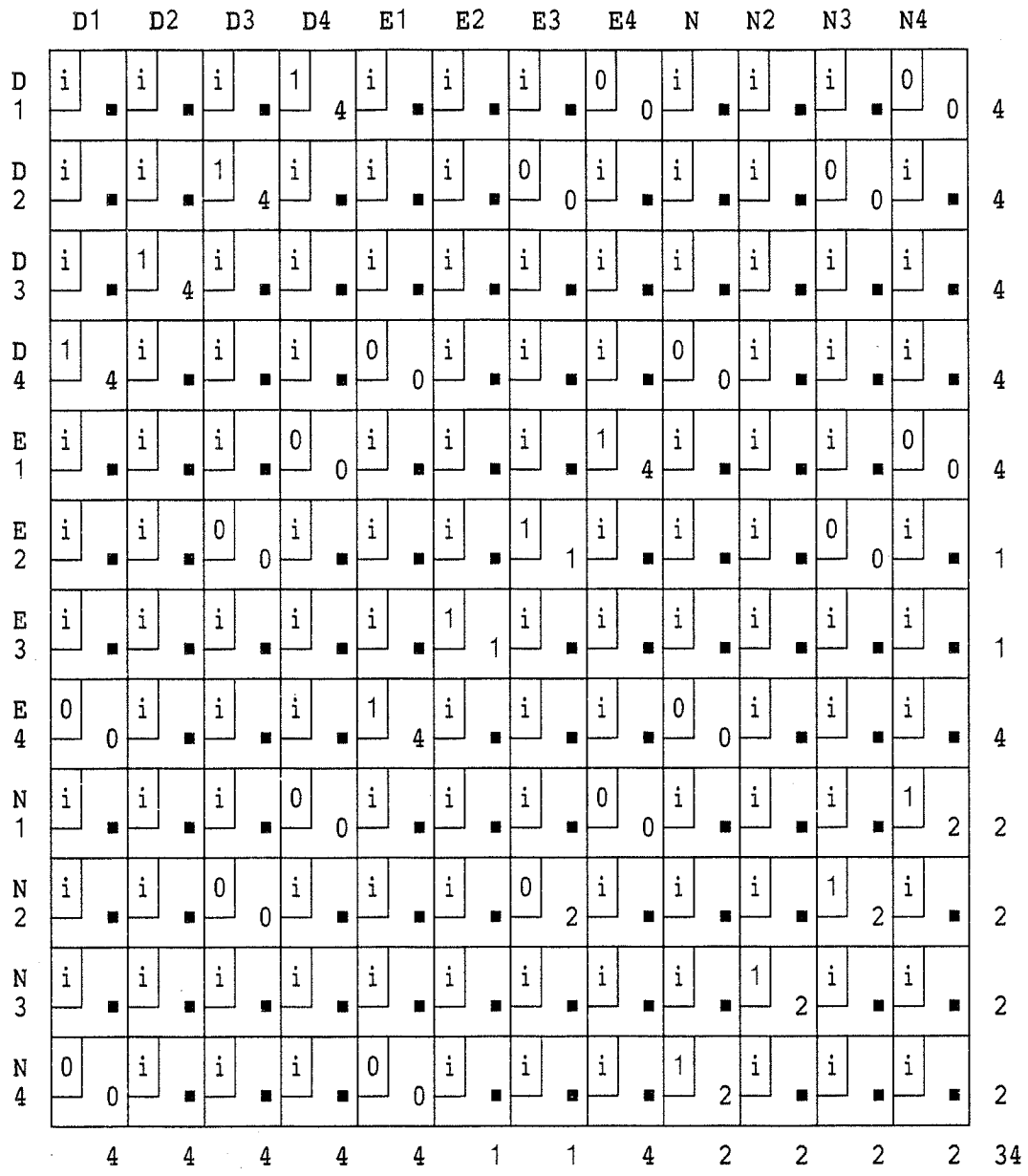


Figure 15: Initial Transportation Tableau

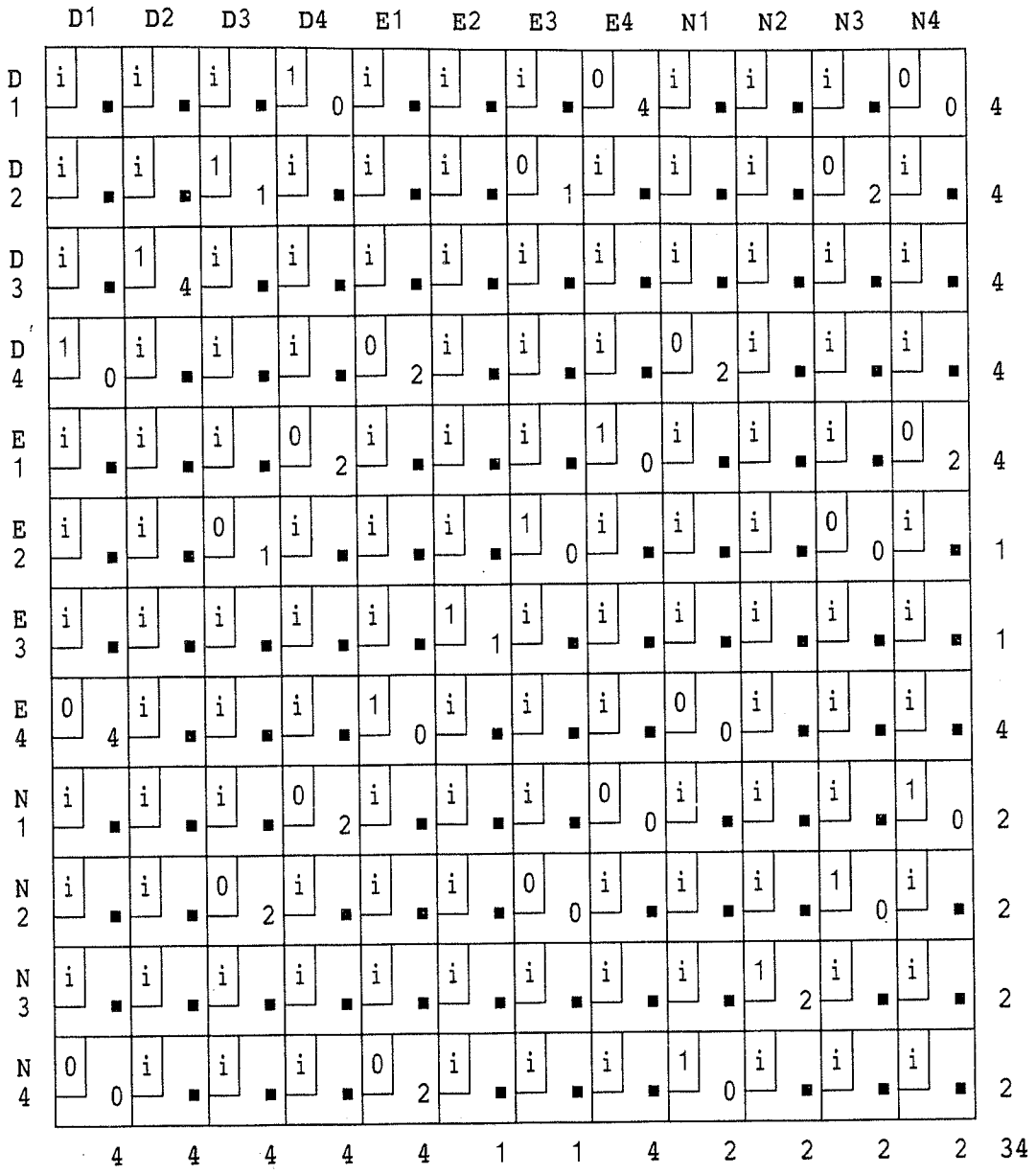


Figure 16: Final Transportation Tableau

| | D 1 | D 2 | D 3 | D 4 | E 1 | E 2 | E 3 | E 4 | N 1 | N 2 | N 3 | N 4 | MTWTFSS | Pop. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| D1 | - | - | - | 0 | - | - | - | 4 | - | - | - | 0 | ·DDD··· | 4 |
| D2 | - | - | 1 | - | - | - | 1 | - | - | - | 2 | - | DDDDD·· | 4 |
| D3 | - | 4 | - | - | - | - | - | - | - | - | - | - | DD··DDD | 4 |
| D4 | 0 | - | - | - | 2 | - | - | - | 2 | - | - | - | DDDDDDD | 4 |
| E1 | - | - | - | 2 | - | - | - | 0 | - | - | - | 2 | ·EEE··· | 4 |
| E2 | - | - | 1 | - | - | - | 0 | - | - | - | 0 | - | EEEEE·· | 1 |
| E3 | - | - | - | - | - | 1 | - | - | - | - | - | - | EE··EEE | 1 |
| E4 | 4 | - | - | - | 0 | - | - | - | 0 | - | - | - | EEEEEEE | 4 |
| N1 | - | - | - | 2 | - | - | - | 0 | - | - | - | 0 | ·NNN··· | 2 |
| N2 | - | - | 2 | - | - | - | 0 | - | - | - | 0 | - | NNNNN·· | 2 |
| N3 | - | - | - | - | - | - | - | - | - | 2 | - | - | NN··NNN | 2 |
| N4 | 0 | - | - | - | 2 | - | - | - | 0 | - | - | - | NNNNNNN | 2 |

Figure 17: Final Solution Matrix

## Chapter XIV

## INDIVIDUAL NURSE MOVEMENTS

In the previous chapter, it was shown how to obtain a solution to a problem with 3 shifts. However, the final tableau contained several rows which had more than one, non-zero elements. Previously, all solution matrices had at most, one non-zero value per row. Although it may appear to be a significant problem, in fact the construction of the individual nurse movements is quite simple.

Before describing the appropriate means of generating individual movements, it is interesting to note how not to apply a solution. Note that of the 4 nurses who work schedule D2, each week 1 nurse moves to D3, 1 moves to E3, while the remaining 2 move to N3. One of the 4 nurses could, every other week, move from D2 to D3, while the other 3 move onto the Evening and Night shifts. Without some regulation, this one nurse could forever avoid evening and night shifts. Unless this is desired, the other nurses may feel they are being treated unfairly. It would be preferable that some means of ensuring all nurses experience the same distribution of each shift. The most straightforward method to prevent this, would be to rank the nurses within each schedule. Then, the weekly movements could dictate where each nurse moves to, based upon their rank in the group.

To begin, list all the combinations of schedules that require a positive schedule population. For the current example, there are 12

such schedules: D1, D2, D3, D4, E1, E2, E3, E4, N1, N2, N3, and N4. Beside each schedule, write the names of 34 nurses; the total number of nurses required. Be sure to include the appropriate number of names for each schedule.

At the end of the week, execute the nurse movements. Start with the first row in the incidence matrix, and the furthest column to the right. In this particular case, the first chosen cell is located at (D1,E4). Starting a new name column, rewrite the names currently associated with D1 along the row associated with E4. Continue with the next cell, (D2,N3), then (D2,E3), and so on. The order of the cells is directed down the rows, but backwards across the columns. Repeat this rewriting of the names, until all cells have been considered. Whenever the number of nurses in a schedule exceeds the cell number, simply take only as many names as needed. For example, listed here are the first 4 (out of 15) steps: Note that instead of names, the characters '1' through '9' and 'A' through 'Z' are used (the letter 'O' is excluded).

1. Nurses 1, 2, 3, and 4 would be rewritten in the E4 row.
2. Nurses 5 and 6 would be rewritten in the N3 row.
3. Nurse 7 would be rewritten in the E3 row.
4. Nurse 8 would be rewritten in the D3 row. And, so on...

In figure 18 there is a display of the first week's movements, along with a single cell movement for the following week. Table 2 gives a more complete sequence of nurse locations. Note that the placement of the nurses repeats after the first 12 weeks. However,

| | D 1 | D 2 | D 3 | D 4 | E 1 | E 2 | E 3 | E 4 | N 1 | N 2 | N 3 | N 4 | Pop. | W(1) | W(2) | ⟶ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| D1 | - | - | - | 0 | - | - | - | 4 | - | - | - | 0 | 4 | 1,2,3,4 | D,E,F,G | |
| D2 | - | - | 1 | - | - | - | 1 | - | - | - | 2 | - | 4 | 5,6,7,8 | 9,A,B,C | |
| D3 | - | 4 | - | - | - | - | - | - | - | - | - | - | 4 | 9,A,B,C | 8,L,U,V | |
| D4 | 4 | - | - | - | 0 | - | - | - | 0 | - | - | - | 4 | D,E,F,G | J,K,S,T | |
| E1 | - | - | - | 2 | - | - | - | 0 | - | - | - | 2 | 4 | H,I,J,K | N,P,Q,R | |
| E2 | - | - | 1 | - | - | - | 0 | - | - | - | 0 | - | 1 | L | M | |
| E3 | - | - | - | - | - | 1 | - | - | - | - | - | - | 1 | M | 7 | |
| E4 | 0 | - | - | - | 4 | - | - | - | 0 | - | - | - | 4 | N,P,Q,R | 1,2,3,4 | D,E,F,G |
| N1 | - | - | - | 2 | - | - | - | 0 | - | - | - | 0 | 2 | S,T | Y,Z | |
| N2 | - | - | 2 | - | - | - | 0 | - | - | - | 0 | - | 2 | U,V | W,X | |
| N3 | - | - | - | - | - | - | - | - | - | 2 | - | - | 2 | W,X | 5,6 | |
| N4 | 0 | - | - | - | 0 | - | - | - | 2 | - | - | - | 2 | Y,Z | H,I | |

Figure 18: Individual Nurse Locations

the nurses repeat their individual pattern after only 4, or after only 6 weeks (depending on which schedules they work).

TABLE 2

Individual Nurse Movements for 3 Shifts

| Week | D1 | D2 | D3 | D4 | E1 | E2 | E3 | E4 | N1 | N2 | N3 | N4 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1234 | 5678 | 9ABC | DEFG | HIJK | L | M | NPQR | ST | UV | WX | YZ |
| 2 | DEFG | 9ABC | 5LUV | HIST | NPQR | M | 6 | 1234 | YZ | WX | 78 | JK |
| 3 | HIST | 5LUV | 9MWX | NPYZ | 1234 | 6 | A | DEFG | JK | 78 | BC | QR |
| 4 | NPYZ | 9MWX | 5678 | 12JK | DEFG | A | L | HIST | QR | BC | UV | 34 |
| 5 | 12JK | 5678 | 9ABC | DEQR | HIST | L | M | NPYZ | 34 | UV | WX | FG |
| 6 | DEQR | 9ABC | 5LUV | HI34 | NPYZ | M | 6 | 12JK | FG | WX | 78 | ST |
| 7 | HI34 | 5LUV | 9MWX | NPFG | 12JK | 6 | A | DEQR | ST | 78 | BC | YZ |
| 8 | NPFG | 9MWX | 5678 | 12ST | DEQR | A | L | HI34 | YZ | BC | UV | JK |
| 9 | 12ST | 5678 | 9ABC | DEYZ | HI34 | L | M | NPFG | JK | UV | WX | QR |
| 10 | DEYZ | 9ABC | 5LUV | HIJK | NPFG | M | 6 | 12ST | QR | WX | 78 | 34 |
| 11 | HIJK | 5LUV | 9MWX | NPQR | 12ST | 6 | A | DEYZ | 34 | 78 | BC | FG |
| 12 | NPQR | 9MWX | 5678 | 1234 | DEYZ | A | L | HIJK | FG | BC | UV | ST |

repeat

## Chapter XV

## CONCLUSIONS

The proposed method of nurse scheduling does have certain features which may sometimes hinder its implementation. A few of the limitations are caused directly by some fundamental assumption; thus corrective measures are not possible. Other limitations are not of this type, therefore adjustments to some key step may overcome the problem. Subsequent research is therefore needed to discover the adjustments appropriate for each 'fixable' limitation.

As depicted in the flowchart on page 12, the complete scheduling solution requires many distinct steps. Some steps are performed seldomly while others are performed regularly. This grouping of steps (into seldom or regularly performed), is dependent on the approach taken. Since the purpose of the first 5 steps is to generate a incidence matrix, they need not be performed for each scheduling period, nor for each ward. Once the incidence matrix and related LP exist, the remaining steps are used to obtain a solution again and again.

This division of the overall method can create some confusion. For each set of unique parameters, steps 2 through 5 must be performed. These steps can be very time-consuming to perform. For each set of unique requirements, steps 6 through 8 must be performed. Although

this division may reduce total calculations, it does tend to conceptually detach the original problem from the final formulation. The hospital scheduler may experience difficulty in fully understanding the advantages of this convoluted approach.

Although it was previously stated that the 'collective agreement' and scheduler together determine the parameters, some clarification on how they interact is necessary. Given that the collective agreement defines acceptable ranges for each parameter, the scheduler must then establish their exact values. The scheduler must be aware of 2 opposing effects. These effects are; (1) the ultimate size of the incidence matrix and its number of variables, and (2) how close is the best incidence matrix solution to meeting the 'minimal workforce lower bound'. This lower bound was discussed by Baker & Burns (2), and describes the fact that the number of nurses needed cannot be less than the product of PERIOD and the weekend requirement. In the example problem, this lower bound is 2x4, or 8 nurses (PERIOD=2, and RSAT=RSUN=4). Since this is the same total as obtained before, the number of nurses needed could not be reduced using less restrictive parameters.

On the other hand, more restrictive parameters might increase the number of nurses required, since the number and diversity of acceptable schedule patterns may be reduced. This can be demonstrated here: if WSMIN were set to 4, the resulting incidence matrix would only comprise of S(32) and S(104). The best solution with this matrix uses a total of 12 nurses. Obviously, if the utilization of nurses is

to be as efficient as possible, the parameters should be made as unrestrictive as allowable to permit the 'minimal workforce lower bound' to be met.

If this were the only consideration, then one might feel that the bigger the incidence matrix--the better. This is definitely not the case. The larger the matrix, the larger the final LP, and the longer it takes to obtain an answer. The best size of the incidence matrix (and thus the relative restrictiveness of the parameters) is one that is as small as possible, while its solution meets the 'minimal workforce lower bound'. The scheduler must therefore choose those parameters which balance these 2 opposing effects. Only a trial-and-error process can fulfil this goal. Therefore this method is hampered by the need to repeatedly search for the best incidence matrix.

Throughout this thesis a fundamental restriction was made: consider only the case where every other weekend is off. This implies a weekend off ratio of 1/2, the demoninator, 2, is directly related to the 2 dimensional incidence matrix. If instead ever third weekend were off, the ratio would be 1/3, and the 2 dimensional matrix should be replaced with a 3 dimensional matrix. Here the row and column structure must be replaced with 3 planes (one for each pair of 2 directions). As difficult as it may be to envision and work with a 3 dimensional matrix, higher dimensions would also be possible as the ratios's demoninator increases. With 3 dimensions, a manual solution would be extremely difficult to obtain; beyond 3, it would be utterly impossible. Therefore, a computerized algorithm is a necessity.

The singular objective function used by this method is either; (1) the minimization of the total workforce, while meeting each of the 7 daily requirements, or (2) the minimization of the greatest excess of nurses for any one day, while meeting the daily requirement, given a constant nurse population. Both of these objectives manipulate the numbers and schedules of the nurses; they do not consider nurse opinion. Although some attempts have been made at maximizing nurse preferences (9), their efforts tend to result in large unwieldy models that are difficult to solve and implement.

Although the approach described in this thesis does not consider any measure of nurse preference, it can accommodate these preferences is so far as those preferences can be defined by the parameters. As was mentioned earlier in this chapter, the scheduler has some flexibility in choosing the ultimate parameters. If certain parameter values are more favoured by the nurses, the scheduler's choice can be modified to reflect these values. It is therefore necessary to obtain these values according to some nurse-surveyed preference scale. This scale would rank the relative preferences of the various parameter values (throughout a set of reasonable possibilities). For instance, the following table shows the reasonable possibilities along with a hypothetical preference ranking. It is apparent the nurses feel that 'no split weekends' is the most important schedule characteristic, and 'a work stretch of 5' is the second most important. The scheduler can now direct his trial-and-error search for the 'best matrix' utilizing this ranking. Thus in some way, some consideration of nurse preference is incorporated.

TABLE 3

Ranked Parameter Preferences

Possible Parameter values:
WSMIN=1, 2, or 3
WSMAX=5, 6, 7, or 8
OSMIN=1 or 2
SPLIT=0 or 1

| Rank | Parameter value |
|---|---|
| 1 | SPLIT=0 |
| 2 | WSMAX=5 |
| 3 | OSMIN=2 |
| 4 | WSMAX=6 |
| 5 | WSMIN=3 |
| 6 | WSMIN=2 |
| 7 | WSMAX=7 |
| 8 | WSMAX=8 |
| 9 | WSMIN=1 |
| 10 | OSMIN=1 |
| 11 | SPLIT=1 |

Several other scheduling methods consider the special circumstances of holidays. This method mostly does not, except for the following treatment. Since holidays represent days that the nurses deserve extra time off, either; (1) the daily requirement of holidays are lower than normal days, or (2) the requirement remains unchanged. If (1) is true, then it could be viewed that a week with a holiday is exactly like any other, except the requirements are different. If (2) is true, then nothing different can be done to modify the schedule--at least, not using this approach. Any other aspects such as reduced work stretches around holidays, extra shift changes, and other complex movements are beyond a solution with this method.

Another limitation is the inability to concurrently schedule a mixture of different nurse classes. The incidence matrix approach cannot be applied unless the various levels of nurses are distinct and not substitutable for one another. For example, say there is a requirement of 10 RNs and 5 LPNs. The present method must solve the scheduling problem as two distinct problems; one problem for the RNs and another for the LPNs.

However, another special case of this example may specify that that 3 RNs are equivalent to 2 LPNs. Although other methods may provide a solution to this case, the current approach cannot. It is possible, however, that some slightly modified method may permit an adequate solution to this case.

The consideration of part time nurses is similar to the case of varying nurse types. Part time nurses can be viewed in one of 2 manners; (1) they can be used as a 'float work pool' without parameters restrictions, or (2) their individual work patterns must meet some set of parameters. If case (1) is true, then the current method can be applied directly. Simply solve the scheduling problem normally making the best use of all available full time nurses. Then, any days whose requirement is not met can be supplemented with nurses from the part time pool. A part time nurse in this situation could likely work patterns that would violate the normal parameters. If case (2) is true, then this approach cannot be applied in its present form.

Recall that in chapter XIII, a means to combine separate solutions for different shifts was discussed. Although the complete multiple shift problem could have been considered as a single incidence matrix, it was solved individually and later combined. This was done since the single matrix would have produced a significantly larger linear program. Despite the benefits of its smaller LPs, the technique does have one failing; the resulting solution may contain one of the following deficiencies: First, the nurses could be divided into several, mutually exclusive groups, each of which has its own unique combination of shifts worked. This becomes a problem when one group experiences a much larger percentage of preferred shifts.

Second, the changing shift movements may differ greatly from the desired movements. Since the transportation algorithm changes movements strictly on the basis of minimizing the pseudo cost variable, the hypothetical goal of moving a certain number of nurses through a specific path cannot be realized. As long as the transportation algorithm step deals solely with this single cost variable, complete control of the final answer is not possible. Future research might investigate the interaction between the measurable Morkovian characteristics of the tableau and the effect of the transportation iterations.

## Appendix A

## DATASET CONTENTS

| S | MON | TUE | WED | THR | FRI | SAT | SUN | WSMON | WSSUN | WSLO | WSHI | DWTOT | OSMON | OSLOW | OSSUN |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 7 | 0 | 0 | 7 | 7 | 7 |
| 2 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 7 | 1 | 1 | 0 | 7 | 6 |
| 3 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 7 | 5 |
| 4 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 7 | 2 | 2 | 0 | 7 | 5 |
| 5 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 2 | 7 | 4 |
| 6 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 2 | 0 | 1 | 4 |
| 7 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 2 | 2 | 1 | 7 | 4 |
| 8 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 3 | 0 | 7 | 3 | 3 | 0 | 7 | 4 |
| 9 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 3 | 7 | 3 |
| 10 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 2 | 0 | 2 | 3 |
| 11 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 2 | 1 | 1 | 3 |
| 12 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 2 | 0 | 1 | 2 | 3 | 0 | 1 | 3 |
| 13 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 2 | 2 | 2 | 2 | 7 | 3 |
| 14 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 2 | 2 | 3 | 0 | 1 | 3 |
| 15 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 3 | 3 | 3 | 1 | 7 | 3 |
| 16 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 4 | 0 | 7 | 4 | 4 | 0 | 7 | 3 |
| 17 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 4 | 7 | 2 |
| 18 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 2 | 0 | 3 | 2 |
| 19 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 2 | 1 | 2 | 2 |
| 20 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 2 | 0 | 1 | 2 | 3 | 0 | 2 | 2 |
| 21 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 2 | 2 | 1 | 2 |
| 22 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 3 | 0 | 1 | 2 |
| 23 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 2 | 3 | 1 | 1 | 2 |
| 24 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 3 | 0 | 1 | 3 | 4 | 0 | 1 | 2 |
| 25 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 2 | 2 | 2 | 3 | 7 | 2 |
| 26 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 2 | 2 | 3 | 0 | 2 | 2 |
| 27 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 2 | 3 | 1 | 1 | 2 |
| 28 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 2 | 0 | 2 | 2 | 4 | 0 | 1 | 2 |
| 29 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 3 | 3 | 3 | 2 | 7 | 2 |
| 30 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 3 | 3 | 4 | 0 | 1 | 2 |
| 31 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 4 | 4 | 4 | 1 | 7 | 2 |
| 32 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 5 | 0 | 7 | 5 | 5 | 0 | 7 | 2 |
| 33 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 5 | 7 | 1 |
| 34 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 2 | 0 | 4 | 1 |
| 35 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 2 | 1 | 3 | 1 |
| 36 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 2 | 0 | 1 | 2 | 3 | 0 | 3 | 1 |
| 37 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 2 | 2 | 2 | 1 |
| 38 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 3 | 0 | 1 | 1 |
| 39 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 2 | 3 | 1 | 2 | 1 |

| S | MON | TUE | WED | THR | FRI | SAT | SUN | WSMON | WSSUN | WSLO | WSHI | DWTOT | OSMON | OSLOW | OSSUN |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 40 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 3 | 0 | 1 | 3 | 4 | 0 | 2 | 1 |
| 41 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 2 | 3 | 1 | 1 |
| 42 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 3 | 0 | 1 | 1 |
| 43 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 3 | 1 | 1 | 1 |
| 44 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 2 | 0 | 1 | 2 | 4 | 0 | 1 | 1 |
| 45 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 2 | 3 | 2 | 1 | 0 |
| 46 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 2 | 4 | 0 | 1 | 1 |
| 47 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 3 | 4 | 1 | 1 | 1 |
| 48 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 4 | 0 | 1 | 4 | 5 | 0 | 1 | 1 |
| 49 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 2 | 2 | 2 | 4 | 7 | 1 |
| 50 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 2 | 2 | 3 | 0 | 3 | 1 |
| 51 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 2 | 3 | 1 | 2 | 1 |
| 52 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 2 | 0 | 2 | 2 | 4 | 0 | 2 | 1 |
| 53 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 2 | 3 | 2 | 1 | 1 |
| 54 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 2 | 4 | 0 | 1 | 1 |
| 55 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 2 | 2 | 4 | 1 | 1 | 1 |
| 56 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 3 | 0 | 2 | 3 | 5 | 0 | 1 | 1 |
| 57 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 3 | 3 | 3 | 3 | 7 | 1 |
| 58 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 3 | 3 | 4 | 0 | 2 | 1 |
| 59 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 3 | 4 | 1 | 1 | 1 |
| 60 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 2 | 0 | 3 | 3 | 5 | 0 | 1 | 1 |
| 61 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 4 | 4 | 4 | 2 | 7 | 1 |
| 62 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 4 | 4 | 5 | 0 | 1 | 1 |
| 63 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 5 | 5 | 5 | 1 | 7 | 1 |
| 64 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 6 | 0 | 7 | 6 | 6 | 0 | 7 | 1 |
| 65 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 7 | 0 | 1 | 6 | 7 | 0 |
| 66 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 7 | 1 | 2 | 0 | 5 | 0 |
| 67 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 2 | 1 | 4 | 0 |
| 68 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 2 | 1 | 7 | 2 | 3 | 0 | 4 | 0 |
| 69 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 2 | 2 | 3 | 0 |
| 70 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 3 | 0 | 1 | 0 |
| 71 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 2 | 2 | 3 | 1 | 3 | 0 |
| 72 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 3 | 1 | 7 | 3 | 4 | 0 | 3 | 0 |
| 73 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 2 | 3 | 2 | 0 |
| 74 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 3 | 0 | 2 | 0 |
| 75 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 3 | 1 | 1 | 0 |
| 76 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 2 | 1 | 1 | 2 | 4 | 0 | 1 | 0 |
| 77 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 2 | 2 | 3 | 2 | 2 | 0 |
| 78 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 2 | 2 | 4 | 0 | 1 | 0 |
| 79 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 3 | 3 | 4 | 1 | 2 | 0 |
| 80 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 4 | 1 | 7 | 4 | 5 | 0 | 2 | 0 |
| 81 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 2 | 4 | 1 | 0 |
| 82 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 3 | 0 | 1 | 0 |
| 83 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 3 | 1 | 1 | 0 |
| 84 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 2 | 1 | 1 | 2 | 4 | 0 | 1 | 0 |
| 85 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 3 | 2 | 1 | 0 |
| 86 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 4 | 0 | 1 | 0 |
| 87 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 2 | 4 | 1 | 1 | 0 |
| 88 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 3 | 1 | 1 | 3 | 5 | 0 | 1 | 0 |

| S | MON | TUE | WED | THR | FRI | SAT | SUN | WSMON | WSSUN | WSLO | WSHI | DWTOT | OSMON | OSLOW | OSSUN |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 89 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 2 | 2 | 3 | 3 | 1 | 0 |
| 90 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 2 | 2 | 4 | 0 | 1 | 0 |
| 91 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 2 | 4 | 1 | 1 | 0 |
| 92 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 2 | 1 | 2 | 2 | 5 | 0 | 1 | 0 |
| 93 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 3 | 3 | 4 | 2 | 1 | 0 |
| 94 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 3 | 3 | 5 | 0 | 1 | 0 |
| 95 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 4 | 4 | 5 | 1 | 1 | 0 |
| 96 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 5 | 1 | 7 | 5 | 6 | 0 | 1 | 0 |
| 97 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 2 | 7 | 2 | 2 | 5 | 7 | 0 |
| 98 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 2 | 7 | 2 | 3 | 0 | 4 | 0 |
| 99 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 2 | 1 | 2 | 3 | 1 | 3 | 0 |
| 100 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 2 | 2 | 7 | 2 | 4 | 0 | 3 | 0 |
| 101 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 2 | 1 | 2 | 3 | 2 | 2 | 0 |
| 102 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 2 | 1 | 2 | 4 | 0 | 1 | 0 |
| 103 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 2 | 2 | 2 | 4 | 1 | 2 | 0 |
| 104 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 3 | 2 | 7 | 3 | 5 | 0 | 2 | 0 |
| 105 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 2 | 1 | 2 | 3 | 3 | 1 | 0 |
| 106 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 2 | 1 | 2 | 4 | 0 | 1 | 0 |
| 107 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 2 | 1 | 2 | 4 | 1 | 1 | 0 |
| 108 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 2 | 2 | 1 | 2 | 5 | 0 | 1 | 0 |
| 109 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 2 | 2 | 2 | 4 | 2 | 1 | 0 |
| 110 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 2 | 2 | 2 | 5 | 0 | 1 | 0 |
| 111 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 2 | 3 | 3 | 5 | 1 | 1 | 0 |
| 112 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 4 | 2 | 7 | 4 | 6 | 0 | 1 | 0 |
| 113 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 3 | 7 | 3 | 3 | 4 | 7 | 0 |
| 114 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 3 | 7 | 3 | 4 | 0 | 3 | 0 |
| 115 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 3 | 1 | 3 | 4 | 1 | 2 | 0 |
| 116 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 2 | 3 | 7 | 3 | 5 | 0 | 2 | 0 |
| 117 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 3 | 1 | 3 | 4 | 2 | 1 | 0 |
| 118 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 3 | 1 | 3 | 5 | 0 | 1 | 0 |
| 119 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 3 | 2 | 3 | 5 | 1 | 1 | 0 |
| 120 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 3 | 3 | 7 | 3 | 6 | 0 | 1 | 0 |
| 121 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 4 | 7 | 4 | 4 | 3 | 7 | 0 |
| 122 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 4 | 7 | 4 | 5 | 0 | 2 | 0 |
| 123 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 4 | 1 | 4 | 5 | 1 | 1 | 0 |
| 124 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 2 | 4 | 7 | 4 | 6 | 0 | 1 | 0 |
| 125 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 5 | 7 | 5 | 5 | 2 | 7 | 0 |
| 126 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 5 | 7 | 5 | 6 | 0 | 1 | 0 |
| 127 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 6 | 7 | 6 | 6 | 1 | 7 | 0 |
| 128 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 7 | 7 | 7 | 7 | 7 | 0 | 7 | 0 |

## BIBLIOGRAPHY


(1) Baker,K.R. "Scheduling a Full-Time Workforce to meet Cyclic Staffing Requirements" Management Science, Vol.20, No.12, Aug.1974

(2) Baker,K.R. & Burns,R.N. "Staff Scheduling with Day-Off and Workstretch Constraints" AIIE Transactions, Vol.11, No.4, Dec.1979

(3) Burns,R.N. "Manpower Scheduling with Variable Demands and Alternate Weekends Off" INFOR, Vol.16, No.2, Jun.1978

(4) Kostreva, Leszcymski, and Passini.

(5) Orlin,J.B. & Bartholdi,J.J. & Ratliff,D.H. "Cyclic Scheduling via Integer Programs with Circular Ones" Operations Research, Vol.28, No.5, Sep.-Oct.1980

(6) Rothstein.

(7) Smith,L.D. "The Application of an interactive algorithm to Develop Cyclical Rotational Schedules for Nursing Personnel" INFOR, Vol.14, No.1, Feb.1976

(8) Smith,L.D. & Wiggins,A. "A Computer-Based Nurse Scheduling System" Comp.& Oper.Res., Vol.4, pp.195-212 1977

(9) Warner,D.M. "Scheduling Nursing Personnel According to Nursing Preference: A Mathematical Programming Approach" Operations Research, Vol.24, No.5, Sep.-Oct.1976

(10) Warner,D.M. & Prawda,J. "A Mathematical Programming Model for Scheduling Nursing Personnel in a Hospital" Management Science, Vol.19, No.4, Dec.1972, Part I