DM204 – Autumn 2013

Scheduling, Timetabling and Routing

Lecture 10

Workforce Scheduling

Marco Chiarandini

Department of Mathematics & Computer Science
University of Southern Denmark

Outline

Course Overview

- ✔ Scheduling
  - ✔ Classification
  - ✔ Complexity issues
  - ✔ Single Machine
  - ✖ Parallel Machine
  - ✖ Flow Shop and Job Shop
  - ✖ Resource Constrained Project Scheduling Model
- Timetabling
  - ✖ Sport Timetabling
  - ✔ Workforce Scheduling
    - Reservations and Education
  - ✔ Crew Scheduling
  - ✔ Public Transports
- Vechicle Routing
  - Integer Programming Approaches
  - Construction Heuristics
  - Local Search Algorithms

Outline

Workforce Scheduling
Overview

Shift/Duty: consecutive working hours / sequence of tasks

- Which is their length? What are they made of?
- Which ones are needed?

The two questions may be addressed together or separatedly.
Eg, shift scheduling, crew scheduling, manpower planning

Roster: shift and rest day patterns over a fixed period of time (a week or a month)
Two main approaches:

- coordinate the design of the rosters and the assignment of the shifts to the employees, and solve it as a single problem.
- consider the scheduling of the actual employees only after the rosters are designed, solve two problems in series.

Features to consider: rest periods, days off, preferences, availabilities, skills.

Outline

Shift Scheduling

Creating daily shifts:

- during each period, b_i persons required
- decide working rosters made of m time intervals not necessarily identical
- n different shift patterns (columns of matrix A) each with a cost c

$$\min c^T x$$
$$st \quad Ax \geq b$$
$$x \geq 0 \text{ and integer}$$

$$\min c^T x$$
$$\begin{array}{r} 10am-11pm \\ 11am-12am \\ 12am-1pm \\ 1pm-2pm \\ 2pm-3pm \\ 3pm-4pm \\ 4pm-5pm \end{array} \begin{bmatrix} 1&0&0&0&0&0&0 \\ 1&1&0&0&0&0&0 \\ 1&1&1&0&0&0&0 \\ 1&1&1&1&0&0&0 \\ 1&1&1&1&1&0&0 \\ 0&1&1&1&1&1&0 \\ 0&0&1&1&1&1&1 \end{bmatrix} x \geq \begin{bmatrix} 1 \\ 2 \\ 2 \\ 3 \\ 4 \\ 2 \\ 2 \end{bmatrix}$$
$$x \geq 0 \text{ and integer}$$

(k, m)-cyclic Staffing Problem

Assign persons to an m-period cyclic schedule so that:

- requirements b_i are met
- each person works a shift of k consecutive periods and is free for the other $m - k$ periods. (periods 1 and m are consecutive)

and the cost of the assignment is minimized.

$$\min c^T x$$

$$\text{st} \quad \begin{matrix} \textit{Monday} \\ \textit{Tuesday} \\ \textit{Wednesday} \\ \textit{Thursday} \\ \textit{Friday} \\ \textit{Saturday} \\ \textit{Sunday} \end{matrix} \begin{bmatrix} 1&0&0&1&1&1&1 \\ 1&1&0&0&1&1&1 \\ 1&1&1&0&0&1&1 \\ 1&1&1&1&0&0&1 \\ 1&1&1&1&1&0&0 \\ 0&1&1&1&1&1&0 \\ 0&0&1&1&1&1&1 \end{bmatrix} x \geq \begin{bmatrix} 3 \\ 4 \\ 6 \\ 4 \\ 7 \\ 8 \\ 7 \end{bmatrix} \qquad \text{(IP)}$$

$$x \geq 0 \text{ and integer}$$

Total Unimodular Matrices

Resume'

Recall: Totally Unimodular Matrices

Definition: A matrix A is totally unimodular (TU) if every square submatrix of A has determinant +1, -1 or 0.

Proposition 1: The linear program $\max\{cx : Ax \leq b, x \in \mathbf{R}^m_+\}$ has an integral optimal solution for all integer vectors b for which it has a finite optimal value if A is totally unimodular

Recognizing total unimodularity can be done in polynomial time (see [Schrijver, 1986])

Total Unimodular Matrices

Resume'

Definition

A $(0,1)$–matrix B has the consecutive 1's property if for any column j, $b_{ij} = b_{i'j} = 1$ with $i < i'$ implies $b_{lj} = 1$ for $i < l < i'$.
That is, if there is a permutation of the rows such that the 1's in each column appear consecutively.

Whether a matrix has the consecutive 1's property can be determined in polynomial time [D. R. Fulkerson and O. A. Gross; Incidence matrices and interval graphs. 1965 Pacific J. Math. 15(3) 835-855.]

A matrix with consecutive 1's property is called an interval matrix

Proposition: Consecutive 1's matrices are TUM.

What about this matrix?

$$\begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Definition A $(0,1)$-matrix B has the circular 1's property for columns (resp. for rows) if the rows of B can be permuted so that the 1's appear consecutively in each column, that is, appear in a circularly consecutive fashion.

The circular 1's property for rows does not imply circular 1's property for columns.

Whether a matrix has the circular 1's property for columns (resp. rows) can be determined in $O(m^2 n)$ time [A. Tucker, Matrix characterizations of circular-arc graphs. (1971) Pacific J. Math. 39(2) 535-545]

Integer programs where the constraint matrix A have the circular 1's property for **columns** can be solved efficiently as follows:

Step 1 Solve the linear relaxation of (IP) to obtain $x'_1, \ldots, x'_n$. If $x'_1, \ldots, x'_n$ are integer, then it is optimal for (IP) and STOP. Otherwise go to Step 2.

Step 2 Form two linear programs LP1 and LP2 from the relaxation of the original problem by adding respectively the constraints

$$x_1 + \ldots + x_n = \lfloor x'_1 + \ldots + x'_n \rfloor \qquad \text{(LP1)}$$

and

$$x_1 + \ldots + x_n = \lceil x'_1 + \ldots + x'_n \rceil \qquad \text{(LP2)}$$

From LP1 and LP2 an integral solution certainly arises (P)

Cyclic Staffing with Overtime

- Hourly requirements b_i
- Basic work shift 8 hours
- Overtime of up to additional 8 hours possible

```
minimize        cx

subject to
```

$$
\begin{array}{c|c}
07 & 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\quad 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\quad 0\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1 \\
08 & 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\quad 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\quad 0\ 0\ 1\ 1\ 1\ 1\ 1\ 1\ 1 \\
09 & 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\quad 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\quad 0\ 0\ 0\ 1\ 1\ 1\ 1\ 1\ 1 \\
10 & 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\quad 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\quad 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 1 \\
11 & 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\quad 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\quad 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1 \\
12 & 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\quad 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\quad 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1 \\
13 & 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\quad 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\quad 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1 \\
14 & 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\quad 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\quad 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1 \\
15 & 0\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\quad 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\quad 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0 \\
16 & 0\ 0\ 1\ 1\ 1\ 1\ 1\ 1\ 1\quad 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\quad 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0 \\
17 & 0\ 0\ 0\ 1\ 1\ 1\ 1\ 1\ 1\quad 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\quad 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0 \\
18 & 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 1\quad 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\quad 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0 \\
19 & 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\quad 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\quad 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0 \\
20 & 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\quad 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\quad 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0 \\
21 & 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\quad 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\quad 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0 \\
22 & 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\quad 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\quad 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0 \\
23 & 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\quad 0\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\quad 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1 \\
24 & 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\quad 0\ 0\ 1\ 1\ 1\ 1\ 1\ 1\ 1\quad 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1 \\
01 & 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\quad 0\ 0\ 0\ 1\ 1\ 1\ 1\ 1\ 1\quad 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1 \\
02 & 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\quad 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 1\quad 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1 \\
03 & 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\quad 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\quad 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1 \\
04 & 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\quad 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\quad 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1 \\
05 & 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\quad 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\quad 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1 \\
06 & 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\quad 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\quad 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1
\end{array}
\qquad x \geq b
$$

$x \geq 0$ and integer.

Days-Off Scheduling

- Guarantee two days-off each week, including every other weekend.

IP with matrix A:

first week	1	1	1	1	1	1	1	1	1	1	1	0
	1	1	1	1	1	1	1	1	1	1	0	0
	1	1	1	1	1	1	1	1	1	0	0	1
	1	1	1	1	1	1	1	1	0	0	1	1
	1	1	1	1	1	1	1	0	0	1	1	1
	0	0	0	0	0	0	0	0	1	1	1	1
	0	0	0	0	0	0	0	1	1	1	1	1
second week	0	1	1	1	1	1	1	1	1	1	1	1
	0	0	1	1	1	1	1	1	1	1	1	1
	1	0	0	1	1	1	1	1	1	1	1	1
	1	1	0	0	1	1	1	1	1	1	1	1
	1	1	1	0	0	1	1	1	1	1	1	1
	1	1	1	1	0	0	0	0	0	0	0	0
	1	1	1	1	1	0	0	0	0	0	0	0

Cyclic Staffing with Part-Time Workers

- Columns of A describe the work-shifts
- Part-time employees can be hired for each time period i at cost c'_i per worker

$$\begin{aligned} \min\ & cx + c'x' \\ st\quad & Ax + Ix' \geq b \\ & x, x' \geq 0 \text{ and integer} \end{aligned}$$

Cyclic Staffing with Linear Penalties for Understaffing and Overstaffing

- demands are not rigid
- a cost c_i' for understaffing and a cost c_i'' for overstaffing
- x' level of understaffing

$$\min cx + c'x' + c''(b - Ax - x')$$

$$st \quad Ax + Ix' \geq b$$

$$x, x' \geq 0 \text{ and integer}$$

Outline

Crew Scheduling
Aka Manpower Planning

Cover a number of job functions using the least possible resources

Constraints difficult to formulate.

- Air-crew scheduling
- Hospital-crew scheduling
- Supermarket-crew scheduling

Model is general that it can handle transportation issues.

$$\begin{aligned} \min\ & c^T x \\ & Ax \geq 1 \\ & x \in \{0,1\} \end{aligned}$$

Set covering/partitioning model

where a_{ij} is 1 iff job i is covered by job schedule (shift) j, and c_j is cost of schedule j.

Crew Scheduling

Pairings problem

Input:

- A set of m flight legs (departure, arrival, duration)
- A set of crews
- A set of n (very large) feasible and permissible combinations of flights legs that a crew can handle (eg, round trips)
- A flight leg i can be part of more than one round trip
- Each round trip j has a cost c_j

Output: A set of round trips of mimimun total cost

Set partitioning problem:

$$
\begin{aligned}
\min\ & c_1x_1 + c_2x_2 + \ldots + c_nx_n \\
& a_{11}x_1 + a_{12}x_2 + \ldots + \ldots a_{1n}x_n = 1 \\
& a_{21}x_1 + a_{22}x_2 + \ldots + \ldots a_{2n}x_n = 1 \\
& \vdots \\
& a_{m1}x_1 + a_{m2}x_2 + \ldots + \ldots a_{mn}x_n = 1 \\
& x_j \in \{0,1\}, \qquad \forall j = 1, \ldots, n
\end{aligned}
$$

Route	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
c_j	8	10	4	4	2	14	10	8	8	10	11	12	6	6	5
	1	0	0	0	0	1	1	1	1	0	0	0	0	0	0
	0	1	0	0	0	1	0	0	0	1	1	1	0	0	0
	0	0	1	0	0	0	1	0	0	1	0	0	1	1	0
	0	0	0	1	0	0	0	1	0	0	1	0	1	0	1
	0	0	0	0	1	0	0	0	1	0	0	1	0	1	1

Set partitioning or set covering??

Often treated as set covering because:

- its linear programming relaxation is numerically more stable and thus easier to solve
- it is trivial to construct a feasible integer solution from a solution to the linear programming relaxation
- it makes it possible to restrict to only rosters of maximal length

Crew Scheduling

Generalization of set partitioning

With a set of p crew members
Generalized set partitioning problem:

$$
\begin{aligned}
\min\ & c_1x_1 + c_2x_2 + \ldots + c_nx_n \\
& a_{11}x_1 + a_{12}x_2 + \quad \ldots \quad + \ldots a_{1n}x_n = b_1 \\
& a_{21}x_1 + a_{22}x_2 + \quad \ldots \quad + \ldots a_{2n}x_n = b_2 \\
& \vdots \\
& a_{m1}x_1 + a_{m2}x_2 + \quad \ldots \quad + \ldots a_{mn}x_n = b_3 \\
& x_1 + x_2 + \ldots x_l \qquad\qquad\qquad\qquad\qquad\quad = 1 \\
& \qquad\qquad\qquad\qquad x_i + x_{i+1} + \ldots x_s \qquad\quad = 1 \\
& \vdots \\
& x_j \in \{0,1\}, \qquad \forall j = 1, \ldots, n
\end{aligned}
$$

Preprocessing for Set Covering/Partitioning

1. if $e_i^T A = 0$ then the ith row can never be satisfied

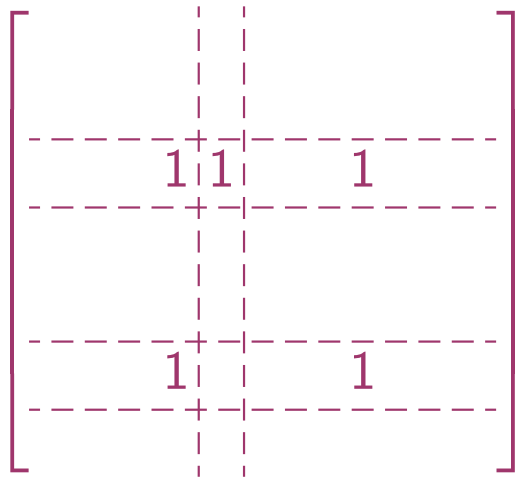
$$\begin{bmatrix} 0\ 0 \ldots 1 \ldots 0 \end{bmatrix} \begin{bmatrix} & & \\ \text{---} & \text{---} & \text{---} \\ & & \\ \text{---} & \text{---} & \text{---} \\ & & \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

2. if $e_i^T A = e_k$ then $x_k = 1$ in every feasible solution

$$\begin{bmatrix} 0\ 0 \ldots 1 \ldots 0 \end{bmatrix} \begin{bmatrix} & | & \\ \text{---} & 1 & \text{---} \\ & | & \\ \text{---} & | & \text{---} \\ & | & \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

In SPP can remove all rows t with $a_{tk} = 1$ and set $x_j = 0$ (and then remove cols) for all cols that cover t

3. if $e_t^T A \geq e_p^T A$ then we can remove row t, row p dominates row t (by covering p we cover t)



In SPP we can remove all cols j:
$a_{tj} = 1, a_{pj} = 0$

4. if $\sum_{j \in S} Ae_j = Ae_k$ and $\sum_{j \in S} c_j \leq c_k$ then we can cover the rows by Ae_k more cheaply with S and set $x_k = 0$
(Note, we cannot remove S if $\sum_{j \in S} c_j \geq c_k$)

ILP Solution

Try to make the constraint matrix such that the polytope that it generates has integral vertices, ie, the solution to the linear relaxation of the set covering problem is integer.

Two ways:

- Generate columns that yield a certain structure in the matrix, ie, balanced matrix
- in branch and bound branch by introducing constraitns that bring the matrix closer to have a certain structure, ie, to be a balance matrix.

Ryan & Foster branching rule

Solving the SPP and SCP integer program

- trivial 1–0 branching leads to a very unbalanced tree in which the 0-branch has little effect

- constraint branching [Ryan, Foster, 1981]
  Identify constraints r_1, r_2 with

$$0 < \sum_{j \in J(r_1, r_2)} x_j < 1$$

  $J(r_1, r_2)$ all columns covering r_1, r_2 simultaneously.
  (there certainly exists one such pair of constraints)
  Branch on:

$$\sum_{j \in J(r_1, r_2)} x_j \leq 0 \qquad \sum_{j \in J(r_1, r_2)} x_j \geq 1$$

Motivation:

A balanced matrix B is an integer matrix that does not contain any submatrix of odd order having row and column sums equal to two (ie, a cycle without chords in the corresponding graph).

$$\begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$$

NO

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

OK

$$\begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

OK

Other insights:

- constraint ordering (petal structure)
- unique subsequence

The remaining fraction must be given by variables that do not cover r_1 and r_2 simultaneously

$$\sum_{j \in J(r_1, r_2)} x_j \leq 0 \text{ 0-branch}$$
$$\sum_{j \in J(r_1, r_2)} x_j \geq 1 \text{ 1-branch}$$

- 0-branch: constraints r_1 and r_2 must not be covered together
- 1-branch: constraints r_1 and r_2 must be covered together
  In SPP can be imposed by forcing to zero all variables/duties in complementary sets $J(\bar{r}_1, r_2)$, $J(r_1, \bar{r}_2)$ ($\bar{r} \equiv$ constraint not covered)

In practice, select r_1 and r_2 such that $\sum_{j \in J(r_1, r_2)} x_j$ is maximized and descend first in the 1-branches

Further Readings

- D.M. Ryan. The Solution of Massive Generalized Set Partitioning Problems in Aircrew Rostering. The Journal of the Operational Research Society, Palgrave Macmillan Journals on behalf of the Operational Research Society, 1992, 43(5), 459-467

- D.M. Ryan and B.A. Foster. An integer programming approach to scheduling. A. Wren (ed.). Computer Scheduling of Public Transport, North-Holland, Amsterdam, 1981, 269-280

- M. Pinedo, Planning and Scheduling in Manufacturing and Services. Springer Verlag, 2005 (Sec. 12.6)

Lagrangian Relaxation Approach

Minimization problem $\rightsquigarrow$ Lagrangian dual is a maximization problem
Subgradient $\gamma = Ax' - b$

Lagrange relaxing all constraints using $\lambda \geq 0$

$$\begin{aligned} \text{minimize} \quad & \sum_{j=1}^{n} c_j x_j - \sum_{i=1}^{m} \lambda_i \left(\sum_{j=1}^{n} a_{ij} x_j - 1 \right) \\ \text{subject to} \quad & x_j \in \{0,1\} \end{aligned}$$

which can be reduced to

$$\begin{aligned} \text{minimize} \quad & \sum_{j=1}^{n} \left(c_j - \sum_{i=1}^{m} \lambda_i a_{ij} \right) x_j + \sum_{i=1}^{m} \lambda_i \\ \text{subject to} \quad & x_j \in \{0,1\} \end{aligned}$$

Latter problem is easily solved by inspection

- $c_j - \sum_{i=1}^{m} \lambda_i a_{ij} < 0$ then $x_j = 1$
- $c_j - \sum_{i=1}^{m} \lambda_i a_{ij} > 0$ then $x_j = 0$
- $c_j - \sum_{i=1}^{m} \lambda_i a_{ij} = 0$ then $x_j = 1$ or 0

Remaining constraints define the convex hull, hence best choice of λ corresponds to dual variables
Many manpower problems are so large that they cannot be solved by LP solvers

$$
\begin{array}{lrrrrrrrrrrrl}
\text{minimize} & 5x_1 & +4x_2 & +6x_3 & +3x_4 & +7x_5 & +2x_6 & +5x_7 & +3x_8 & +2x_9 & +3x_{10} & & \\
\text{subject to} & x_1 & & +x_3 & & +x_5 & & & +x_8 & & & \geq & 1 \\
& & x_2 & +x_3 & +x_4 & +x_5 & & +x_7 & +x_8 & & +x_{10} & \geq & 1 \\
& & & & x_4 & & +x_6 & & & & & \geq & 1 \\
& x_1 & +x_2 & +x_3 & & & +x_6 & & +x_8 & & +x_{10} & \geq & 1 \\
& x_1 & +x_2 & & & & & +x_7 & +x_8 & +x_9 & & \geq & 1 \\
& & x_2 & +x_3 & & & & & & & +x_{10} & \geq & 1 \\
& & & & & x_5 & & & & & & \geq & 1 \\
& x_1 & +x_2 & & +x_4 & & +x_6 & & +x_8 & & +x_{10} & \geq & 1 \\
& x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10} \in \{0,1\} & & & & & & & & & & &
\end{array}
$$

Lagrange relaxed

$$
\begin{array}{ll}
\text{minimize} & (5-\lambda_1-\lambda_4-\lambda_5-\lambda_8)x_1 \\
& +(4-\lambda_2-\lambda_4-\lambda_5-\lambda_6-\lambda_8)x_2 \\
& +(6-\lambda_1-\lambda_2-\lambda_4-\lambda_6)x_3 \\
& +(3-\lambda_2-\lambda_3-\lambda_8)x_4 \\
& +(7-\lambda_1-\lambda_2-\lambda_7)x_5 \\
& +(2-\lambda_3-\lambda_4-\lambda_8)x_6 \\
& +(5-\lambda_2-\lambda_5)x_7 \\
& +(3-\lambda_1-\lambda_2-\lambda_4-\lambda_5-\lambda_8)x_8 \\
& +(2-\lambda_5)x_9 \\
& +(3-\lambda_2-\lambda_4-\lambda_6-\lambda_8)x_{10} \\
& +\lambda_1+\lambda_2+\lambda_3+\lambda_4+\lambda_5+\lambda_6+\lambda_7+\lambda_8 \\
\text{subject to} & x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10} \in \{0,1\}
\end{array}
$$

Start with $\lambda^{(1)} = (1,1,1,1,1,1,1,1)$
$\underline{z} = 0$, $\bar{z} = 40$ (sum of all costs)

Example

Iteration 1

$$\begin{array}{ll} \text{minimize} & x_1 - x_2 + 2x_3 + 0x_4 + 4x_5 - x_6 + 3x_7 - 2x_8 + x_9 - x_{10} + 8 \\ \text{subject to} & x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10} \in \{0,1\} \end{array}$$

Optimal solution $x' = (0,1,0,1,0,1,0,1,0,1)$, infeasible
$\underline{z} = 3, \overline{z} = 40$

$$\gamma^{(1)} = Ax' - b = (0,3,1,3,1,1,-1,4),$$

$$\theta = 0.19, \lambda^{(2)} = (1.00, 0.42, 0.81, 0.42, 0.81, 0.81, 1.19, 0.22)$$

Iteration 2

$$\begin{array}{ll} \text{minimize} & 2.56x_1 + 1.34x_2 + 3.36x_3 + 1.56x_4 + 4.39x_5 + \\ & 0.56x_6 + 3.78x_7 + 0.14x_8 + 1.19x_9 + 1.14x_{10} + 5.66 \\ \text{subject to} & x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10} \in \{0,1\} \end{array}$$

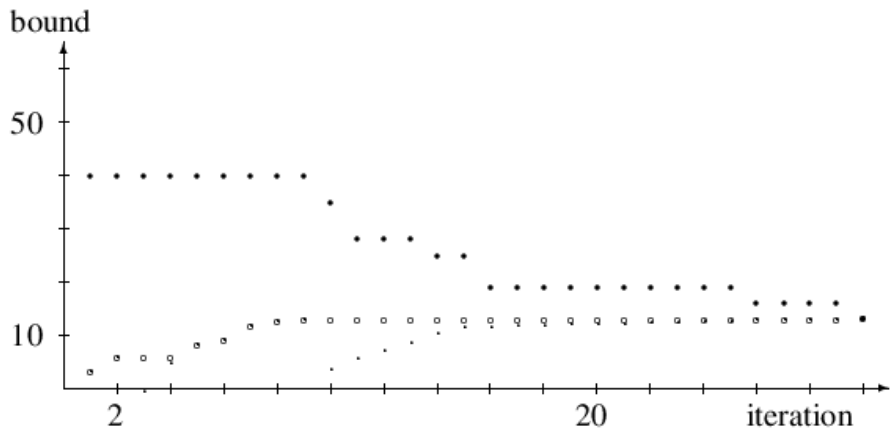
Optimal solution $x' = (0,0,0,0,0,0,0,0,0,0)$, infeasible
$\underline{z} = 5.66, \overline{z} = 40$

$$\gamma^{(1)} = Ax' - b = (-1,-1,-1,-1,-1,-1,-1,-1),$$

$$\theta = 0.86, \lambda^{(3)} = (1.86, 1.27, 1.66, 1.27, 1.66, 1.66, 2.05, 1.08)$$

Example

Development of $\overline{z}$ and $\underline{z}$.



After 30 iterations we have $\underline{z} = \overline{z} = 13$

$$\lambda = (1.85, 0, 2.77, 0, 1.13, 2.99, 6.11, 0)$$

Dual variables

$$y = (0, 0, 2, 0, 1, 3, 7, 0)$$

Outline

Nurse Scheduling
A CP approach

- Hospital: head nurses on duty seven days a week 24 hours a day
- Three 8 hours shifts per day (1: daytime, 2: evening, 3: night)
- In a day each shift must be staffed by a different nurse
- The schedule must be the same every week
- Four nurses are available (A,B,C,D) and must work at least 5 days a week.
- No shift should be staffed by more than two different nurses during the week
- No employee is asked to work different shifts on two consecutive days
- An employee that works shifts 2 and 3 must do so at least two days in a row.

Mainly a feasibility problem

A CP approach

Two solution representations

	Sun	Mon	Tue	Wed	Thu	Fri	Sat
Shift 1	A	B	A	A	A	A	A
Shift 2	C	C	C	B	B	B	B
Shift 3	D	D	D	D	C	C	D

	Sun	Mon	Tue	Wed	Thu	Fri	Sat
Worker A	1	0	1	1	1	1	1
Worker B	0	1	0	2	2	2	2
Worker C	2	2	2	0	3	3	0
Worker D	3	3	3	3	0	0	3

Variables: w_{sd} nurse assigned to shift s on day d
and y_{id} the shift assigned to nurse i on day d

$$w_{sd} \in \{A, B, C, D\} \qquad y_{id} \in \{0, 1, 2, 3\}$$

Three different nurses are scheduled each day

$$\texttt{alldiff}(w_{\cdot d}) \qquad \forall d$$

Every nurse is assigned to at least 5 days of work

$$\texttt{cardinality}(w_{\cdot\cdot} \mid (A, B, C, D), (5, 5, 5, 5), (6, 6, 6, 6))$$

At most two nurses work any given shift

$$\texttt{nvalues}(w_{s\cdot} \mid 1, 2) \qquad \forall s$$

All shifts assigned for each day

$$\texttt{alldiff}(y_{.d}) \qquad \forall d$$

Maximal sequence of consecutive variables that take the same values

$$\texttt{stretch-cycle}(y_{i.} \mid (2,3),(2,2),(6,6),P)$$
$$\forall i,\ P = \{(s,0),(0,s) \mid s = 1,2,3\}$$

Channeling constraints between the two representations:
on any day, the nurse assigned to the shift to which nurse i is assigned must be nurse i (element constraint)

$$w_{y_{id},d} = i \qquad \forall i,d \qquad \texttt{element}(y_{id},(w_{0d},\ldots,w_{3d}),z_{id})$$
$$y_{w_{sd},d} = s \qquad \forall s,d \qquad \texttt{element}(w_{sd},(y_{Ad},\ldots,y_{Dd}),z_{sd})$$

$\texttt{element}(y,x,z)$: z be equal to the yth variable in the list $(x_1 \ldots, x_m)$

The complete CP model

$$\text{Alldiff:} \left\{ \begin{array}{l} (w_{\cdot d}) \\ (y_{\cdot d}) \end{array} \right\}, \text{ all } d$$

$$\text{Cardinality: } (w_{\cdot\cdot} \mid (A,B,C,D),(5,5,5,5),(6,6,6,6))$$

$$\text{Nvalues: } (w_{s\cdot} \mid 1,2), \text{ all } s$$

$$\text{Stretch-cycle: } (y_{i\cdot} \mid (2,3),(2,2),(6,6),P), \text{ all } i$$

$$\text{Linear:} \left\{ \begin{array}{l} w_{y_{id}d} = i, \text{ all } i \\ y_{w_{sd}d} = s, \text{ all } s \end{array} \right\}, \text{ all } d$$

$$\text{Domains:} \left\{ \begin{array}{l} w_{sd} \in \{A,B,C,D\}, \; s = 1,2,3 \\ y_{id} \in \{0,1,2,3\}, \; i = A,B,C,D \end{array} \right\}, \text{ all } d$$

Constraint Propagation:

- alldiff: matching
- nvalues: max flow
- stretch: poly-time dynamic programming
- index expressions $w_{y_{id}d}$ replaced by z and constraint:
  `element`(y, x, z): z be equal to y-th variable in list $x_1, \ldots, x_m$

Search:

- branching by splitting domanins with more than one element
- first fail branching
- symmetry breaking:
  - employees are indistinguishable
  - shifts 2 and 3 are indistinguishable
  - days can be rotated

  Eg: fix A, B, C to work $1, 2, 3$ resp. on sunday

Heuristic Methods

- Local search and metaheuristic methods are used if the problem has large scale.
- Procedures are very similar to what we saw for course timetabling.