Making good rosters for a 24/7 environment

Han Hoogeveen and Eelko Penninkx
Institute of Information and Computing Sciences
Utrecht University
The Netherlands
j.a.hoogeveen@cs.uu.nl

January 14, 2010

Basic problem 24/7 rostering

- Given set of employees.
- Workers are needed 24/7.
- Desired occupation may vary (per shift and per day).
- The day is divided in three shifts: Morning-Late-Night (small deviations are okay).
- Rosters must follow the order M-L-N ('gezond roosteren').

Objective: find a feasible solution.

Traditional roster: cyclic. Every employee follows the same set of weekrosters.

New trend: individual rosters satisfying personal preferences in the rosters (as much as possible).

Contents of the talk

- Description of example problem (doorkeepers at UMC).
- Solution method.
- Results.
- Possible extensions.

UMC Problem

- 35 employees.
- All employees are qualified for all kinds of work.
- Work in shifts to man several posts 24/7.
- Three shifts per day: Morning-Late-Night.
- Minimum attendance per shift is given, but additional personnel can be hired.
- There are training sessions on Wednesday morning.

Objective: Generate a good roster for a whole year, taking into account roster appreciation, shortage, overstaffing.

Rostering constraints

- There are ‘work’ shifts and stand-by shifts.
- Shifts in a roster must be in the order Morning-Late-Night.
- At most 4 consecutive Night shifts.
- ‘Enough’ time in between Night and Morning shift.
- Approximately 34 working hours per week (contract: 36 hours per week).
- The number of working hours must be ‘reasonably balanced’.

For the problem approach, we ignore the stand-by shifts afterwards (we fill these in later, given the rosters).

Personal appreciation of a roster

- Fixed day off each week (with preference concerning the day).
- Work both Saturday and Sunday, or both free.
- Vacation period off (not too many people at the same time).
- Specific day off (birthdays, etc.).
- Number of consecutive Morning/Late/Night shifts in one run.
- Number of days off after a series of Morning/Late/Night shifts.
- The cost of a feasible roster is scaled from 0 (perfect) to 1 (horrible).

We can guarantee a fixed day off if not too many people choose the same day.

Our approach in a nutshell

You need one year-roster per employee such that

- each employee is happy
- the combination of the chosen rosters is ‘optimal’

The quality of a schedule is computed as the sum of

- Total appreciation
- Total shortage cost (0.1 per employee short per shift)
- Total overstaffing (per employee per shift: 0.0 for Wednesday morning; 0.01 for remaining weekdays Morning/Late; 1000 for Night and weekend)

Basic idea

1. Generate for each employee a number of appreciated rosters that are ‘combinable’ (reduce search space).
2. Pick the rosters that form the best combination.

A similar idea can be used to find a representative team.

- Select players that might make it to the team.
- Build the best team (for example through the ‘computer coach’ program by Gerard Sierksma): the fewer ‘poor’ candidates, the faster.

Selecting the best combination: ILP

Suppose we are given for each employee a set of desirable, useful year-rosters.

- Introduce a binary decision variable for each available roster; selecting the roster corresponds to putting the variable equal to 1.
- Minimize the total cost (appreciation, shortage, overstaffing), such that
  - there are enough employees available in each shift (including shortage and overstaffing).
  - each employee gets one roster,

Question: how do we find this set (call it S) of desirable year-rosters?

Identifying S: column generation

Intuition: the solution of a simplification of the problem will resemble the solution of the real problem.

- Simplify the problem by taking the LP-relaxation (rosters can be chosen with a fractional value).
- Solve the LP-relaxation using column generation.
- Put the set of generated rosters in S.
- Additional trick: After solving the LP-relaxation, determine for each employee the set of 2500 year-rosters with minimum reduced cost and add these to S.

Column generation

- Solve the LP-relaxation starting with a small set of rosters.
- LP-theory: a feasible year-roster outside this set will improve the value of the LP-relaxation only if its reduced cost is negative $\Longrightarrow$
- solve the **pricing problem** of finding the feasible year-roster with minimum reduced cost.
- The LP-relaxation has been solved to optimality if this minimum is ≥ 0.

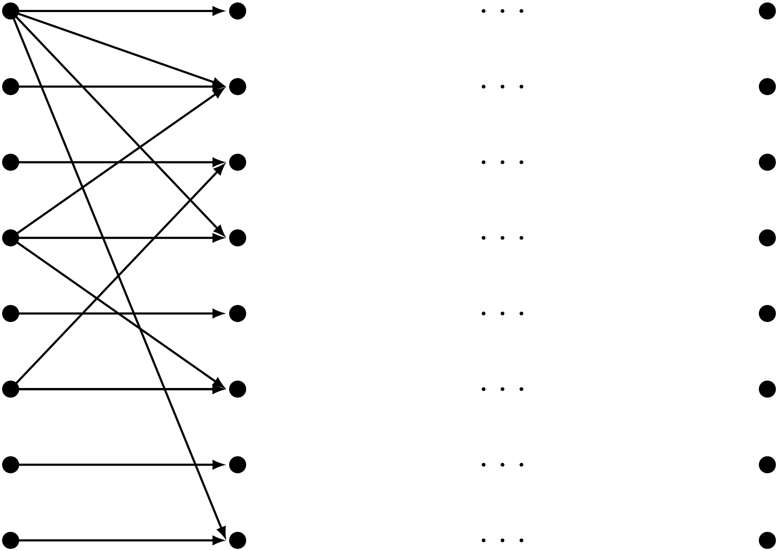
Pricing problem

- The reduced cost of a given roster is equal to the cost of this roster minus the total value of the dual multipliers of the included work shift minus a constant depending on the chosen employee.
- For a given employee, the pricing problem is defined as
  - select the work shifts for this employee that
  - minimize the reduced cost such that
  - the selected work shifts constitute a feasible roster.
- Rostering constraints
  - Shifts in a roster must be in the order Morning-Late-Night.
  - At most 4 consecutive Night shifts.
  - 'Enough' time in between Night and Morning shift.
  - Approximately 34 working hours per week (contract: 36 hours per week).
  - The number of working hours must be 'reasonably balanced'.

Solve the pricing problem

Consider a given employee and his preferences

- Enumerate all feasible four-week rosters
- Eliminate the ones with cost more than 0.5 plus ‘dead ends’
- Use these in a layered-graph, where layer k corresponds to the kth four-week period
- Connect two vertices in successive layers if this gives a feasible eight-week roster
- A path through this layered network corresponds to a feasible year roster



Solving the pricing problem

See to it that the length of a path corresponds to the reduced cost of the corresponding year roster $\Longrightarrow$ Solve the pricing problem as a shortest path problem.

The reduced cost consists of

- the value of the year-roster: this can be expressed as the value of the included four-week periods plus the connecting arcs
- the sum of the dual multipliers of the included shifts: this can be included by adjusting the value of the included four-week periods
- a constant term per employee: easy to handle, since we consider all non-identical employees separately.

Some data

Depending upon the employee:

- Each layer consists of approximately 1000-2000 nodes
- There are approximately 20.000-170.000 arcs between two layers (on average 75.000)

The LP-relaxation can be solved to optimality; this gives a lower bound on the value of the optimal solution.

Solving the resulting ILP

Unfortunately: this ILP is too big to be solved.

Remedy: use a rolling-horizon approach

1. Compute an optimal solution for the LP-relaxation for the first 2 periods (8 weeks)
2. Pick the 2500 year-rosters with minimum reduced cost per employee.
3. Solve the resulting ILP.
4. Fix the first period (four weeks) accordingly.
5. Repeat the procedure for weeks 5 and onward.

Results

- Solving the LPs (CPLEX 9.0) altogether requires approximately 1 hour (but this can be reduced a lot).
- Solving the ILPs (CPLEX 9.0) altogether requires 1 hour.
- The integrality gap is very small (3%).
- **Increasing the number of employees does not lead to longer running times, but the integrality gap reduces**.
- **The method breaks down if there are too few employees**.

Extension: pairs of employees

1. Employees working in fixed couples.
   - Easy to model: selecting a roster corresponds to allocating two people.
   - May complicate small instances of the problem (or bigger ones if there are many of these combinations).
2. Employees who should never work together.
   - Easy to model: construct **combined rosters** for these two people.
   - No problems, once the combined rosters have been generated.

A few of these pairs of employees are easy to handle.

Extension: working in teams

1. Working in fixed teams
   - Easy to model: one roster for all.
   - Takes away a lot of freedom: causes problems for medium sized instances.
2. Each person in the team must meet each other person at least $\cdots$ times per month.
   - Very challenging to model.
   - Work-around: construct solution, check the constraint, and adjust it locally.
3. Supervisor must meet each employee at least $\cdots$ times per month.
   - Supervisor's schedule is given: allow only use four-week schedules per employee that satisfy this constraint.
   - Supervisor's schedule is **not** given: challenging, like above.

Extension: qualified people required

- Only a subset of the employees has a special qualification.
- At least $\cdots$ people with this qualification are required per shift.
    - Easy to model: add a constraint for each shift.
    - Hard to solve: the model gets 1000+ additional constraints (per qualification).
    - Work-around: ignore constraint and adjust solution locally. Smart preprocessing will help.

Questions?