ORTEC's contribution to the Second International Nurse Rostering Competition

Hujin Jin · Gerhard Post · Egbert van der Veen

1 Introduction

ORTEC is a company based in the Netherlands that provides optimisation solutions to their clients. One of these solutions is ORTEC Workforce Scheduling. This product enables the planners of the clients to generate and maintain schedules. The generation of schedules can be controlled by a very extensive list of hard and soft constraints, and for this reason the engine is constructed as a "black box" engine: only the total cost steers the algorithm, not the nature of the violations.

ORTEC's aim for participating in the Second International Nurse Rostering Competition was to assess the current algorithms. This assessment focuses on the two main aspects present in the competition. The first aspect is the generation of a week's schedule. This is the classical shift assignment problem, that our solver should be able to handle. The second problem is robustness to uncertainties in the upcoming week. The randomness in work demands and employees requests in practice exists, but not by the amount present in the competition, and usually more ahead in time. Scheduling the Sunday with little knowledge on the Monday might not be very realistic, but scheduling a month with uncertainties for the upcoming month surely is. Hence the second problem is relevant in practice for connecting month schedules.

2 Connecting weeks

Before deciding on how to tackle the problem of the connecting weeks, we analyzed the instances and ran our algorithm without adjustments. From this we saw that infeasibilities are absent for instances with 30 employees or more. Another observation was, that though the variation in demand per shift type and skill is huge, the variation in the total number of shifts per week is almost

ORTEC, Houtsingel 5, 2719 EA Zoetermeer

absent. Moreover the data is such that reaching the minimum staffing demands is not really an issue, but reaching the optimal staffing demands always violates the requested maximum number of assignments.

The high variation in demand per shift type and skill will probably also induce unavoidable costs, as it will be impossible to satisfy the constraints on consecutive assignments per shift type. However these costs are more difficult to determine. in practical situations such high fluctuations of shift demands are not likely.

These considerations led us to the conclusion that during the generation of a week schedule, we will steer on a good connection to the next week by introducing some artificial soft constraints (see the next section). In this way we can use our solver without big changes.

3 Generating a schedule for a week

Thanks to the choice for connecting weeks, we are on solid ground now, as we could apply our solver as is. To improve the results, for the competition we investigated (and also used) some ejection chain methods. It turned out to be difficult to assess the results, mainly because of the high randomness in the selected weeks.

To improve the connection to the next week, we added two soft constraints.

- We make sure that on the last day of the week (Sunday) we could schedule the next Monday, assuming that the shift demands for this Monday were the same as the current Monday. That is, we put a quite high penalty if this would not be the case, and a lower penalty if the situation would be tight. We noticed as side-effect that the cost decreased; the increased flexibility could sometimes be used to improve the next week's schedule.
- At the end of the week, different shift length constraints could have different opinions on how to continue next week. Suppose for example that we place a stint of three night shifts on Friday, Saturday, and Sunday, but that we wish to have stints of five shifts long, but at the same time at most three night shifts in a row. Then a penalty for this stint is unavoidable.

For the global counters for the maximum number of shifts and the number of worked weekends, we adjusted the parameters in a proportional way. For example, if the schedule is for four weeks, and the maximum number of shifts for an employee is 16, then in the first week we put the maximum to 4, in the second week the maximum of that week to 8 (including the count for the first week), etcetera.

4 Random seeds optimization

The selection of the finalists of the competition was based on the best results on a set of given instances, i.e. we still should choose the random seeds of our algorithm to tune to the randomness of the instances. Principally this way of selecting the finalists is rather strange. The competition is about creating robust scheduling methods for connecting the different weeks, but the adjudication is based on generating outliers, lucky shots, that create the best overall result. This tuning annihilates the essence of the competition, i.e. to deal with uncertainties in the upcoming weeks. Fortunately in practice the effects seemed to be mild. The effect can seen the best in average rank scores for number one of the competition (1.21 with random seeds optimization, 1.76 without).

To have reproducable results turned out to be difficult, as a small change in one of the weeks, can have a tremendous effect on the complete schedule.

We aimed at obtaining for each instance a solution of two standard deviations below the average cost in the runs. Assuming a normal distribution for the costs of the generated schedules, we can expect that we need around 40 runs to obtain such result. In the tight time frame this was all we could achieve.

5 Conclusion

With the methods described above, we were selected as finalist and ranked fifth in the final ranking. This shows that we are able to compete with the best. On the other hand, we know that there are issues we can still improve on. This competition has given a new impulse in improving our algorithms.