IHTC2024 - Operation STOR-i Time(tabling)

Team participants

All participants of "Operation STOR-i Time(tabling)" are PhD students at the STOR-i Centre for Doctoral Training based at Lancaster University, UK.

- Matthew Davison (contact: m.davison2@lancaster.ac.uk)
- Adam Page
- Ben Lowery
- Graham Burgess
- Rebecca Hamm

Method

Overview

Our fully open source approach starts by attempting to generate a feasible initial solution using a greedy heuristic. With the remaining clock time, we run an iterative hyper-heuristic procedure consisting of a move selection and move acceptance step. The move portion is parallelised, allowing for up to 4 different move evaluations for each iteration step. The elements of the algorithm are described below.

Initial Feasible Solution

The initial feasible solution generator applies the following steps to try and find a feasible solution:

1. Create an ordered list of mandatory patients
2. Set up a blank solution
3. Take the list of patients, and try and admit them in sequence
4. If any mandatory patients remain unassigned, move these to the front of list and return to step 2. Otherwise, continue
5. Assign all non-mandatory an admission day of "None"
6. Calculate workload for each room
7. Iterate over days, shifts, nurses and rooms. Assign nurses to rooms, shifts and days if they still have a non-negative amount of remaining workload
8. Return the resulting solution

At the end of Step 3, the algorithm will check if the feasible solution generator has been running for more than 55 seconds. If it has, then it will return the current solution it has found, which may not be feasible. Likewise, the resulting solution at Step 7 may not be feasible. The algorithm checks the solution and will only continue to the next step (the iterative hyper-heuristic improvement) if the solution is feasible.

Iterative improvement hyper-heuristic

Move Selection

In each parallel evaluation, a single low-level heuristic is applied to the solution. A simple random approach is used to select the low-level heuristic from the following list:

Move 1 - Insert non-mandatory patient

Move 2 - Insert non-mandatory patient with least workload

Move 3 - Insert a non-mandatory patient to available surgeon/theater/room

Move 4 - Remove a single non-mandatory patient

Move 5 - Remove any patient

Move 6 - Remove then insert a non-mandatory patient sequentially

Move 7 - Remove then insert any patient sequentially

Move 8 - Change a patient room

Move 9 - Change patient's admission day

Move 10 - Change a patient's theatre

Move 11 - Change a patient's room, admission, and theatre sequentially

Move 12 - Change a patient's room and admission sequentially

Move 13 - Change a patient's admission and theatre sequentially

Move 14 - Add a nurse to a room

Move 15 - Remove a nurse from a room

Move 16 - Remove a nurse from a room but add them to another

Move Acceptance

Once the chain of moves and corresponding costs are recorded. We accept each parallel evaluation according to a simulated annealing based acceptance procedure. If the number of accepted moves is less than 4, then we populate remaining spots in the solution pool with copies of the current best solution (which could be from the previous pool if no improving solutions were found).

Solution scores

To control the stochasticity of our method as much as possible the algorithm accepts a seed as input. To benefit the competition organisers, we have ran the algorithm 20 times for each instance to obtain a mean score and a standard deviation of these scores. Our best solution scores and the instances provided in the .zip file are our official solution entries.