Optimizing Surgical Scheduling in Healthcare Services: Models and Algorithms for Minimizing Makespan in No-Wait Scenarios

This thesis job studies algorithms aimed at optimizing two problems encountered in the organization of operating rooms in healthcare facilities. In particular the problems regard the scheduling of patients in a surgery environment in which only two rooms are available for the set of procedures to execute. Each patient must go through a sequence of tasks to perform the surgery: Anestesia, Surgery and Awakening. Each task is bound to be performed in one of the two rooms or in any of the two, with the constraint that subsequent tasks of the same patient must have no time in between. The objective of the algorithms is to reduce the makespan of the schedule, so the time between the beginning of the first operation of the schedule and the ending of the last operations. The two problems are similar and one the generalization of the other: in the first one the ordering of patients is pre-determined and only the placement (with respect to the available rooms) must be decided, the problem has a complexity of 2^n with n being the number of patients. In the second problem the ordering of patients must also be decided, thus leading to an event space of possible combinations of dimension $2^n * n!$. Such a great amount of combinations leads to the inconvenience of using linear programming to solve the problem, due to the computation time becoming too big. This thesis first explores various models to solve the problems through the usage of a commercial solver, and then explores some ad hoc algorithms aimed at solving the problem as efficiently as possible with a very low computation time (and no need for a very expensive license for the solver). The results obtained will show an optimal algorithm found for the first problem, and an algorithm for the second problem that doesn't always find an optimal solution but always a very close one compared with the solver's solution but with fractions of the time needed for computation.

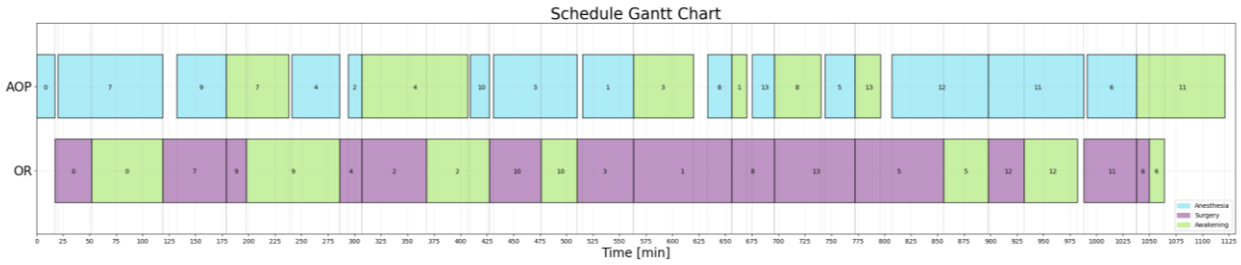

Figure 1 - Example of optimized schedule