Planning Technique:

The basic concept behind my planning technique is a genetic evolutionary algorithm.

The main problems that I had to tackle and will discuss on this page are:

- How do I encode my data in genomes?
- How do I formulate a fitness function that best represents the given planning problem?
- How do I do the crossover?
- How do I implement an adequate stopping mechanism?

As I implemented the algorithm so that only one patient at a time will be replanned, the population of genomes will consist only of one single patient. For this patient I encoded multiple genomes, in which I set replanning times as the content of the genome. Then I created 10-20 genomes per patient which means 10-20 replanning times.

To come up with the replanning times I used a simple heuristic where I evenly distributed the replanning times over the available time span of possible replanning times during working hours (Mo-Fr 8am-5pm). The possible replanning times are calculated as following:
Start : current_replan_time
End: (First_admission_time + 7 days) = last_possible_replan_time

From this I could calculate all the possible timespans which are working hours, take the overall resulting_time_span, divide it by the number of replanning times/ population size and then deduce the replanning times by adding i-times the time resulting_time_span to the current_replan_time (and of course skip non working hours).

Then I applied the fitness function on each genome and get a resulting score per genome. The population is sorted by score per genome and then I conducted crossover.

The crossover includes some elitism, where I actually keep the better half of the population and do crossover only for the worse half of the population.
For the actual crossover I took 2 of the better genomes and calculated the average replanning time of them and use it as new replanning time.

I am aware that this is not actually how a evolutionary algorithm is optimally implemented and it would probably perform better I had added more patients to a replanning process and form a genome by something like 5 patients replanning times and then have at least 10 patients which I use for replanning and doing some proper crossover. But as I mostly had only 1-2 patients to replan or at most 4, I decided to go for this rather simple approach.

If you print the algorithm iterations in the evolve method of evolution.py, you can see that the algorithm does improve the scores over the iterations. For the very most patients, the improvement stops after at most 10 iterations, so I decided to go for 10 iterations *hard coded*.
I also tried some other stopping mechanism where the algorithm performs evolution cycles until the improvement is not more than 10% for 4 consecutive evolution cycles but this did not perform very well.