For the baseline, we tried to generate a uniform sample from the entire state space.

There are multiple variables.

1. Varying how courses are divided into different lessons.
2. Varying the distribution of the lessons over the “roomslots”.
3. Varying the distribution of students over the lessons.

If all the lectures, tutorials and labs are scheduled once, there are 72 lessons. However, each tutorial and lab has a maximum number of students per lesson. To be able to fit students in the maximum number of students per lesson, tutorials, and labs must be split into multiple lessons. Ultimately, there is a minimum of 129 lessons if we state that all students should be scheduled in the required lessons for their courses, considering the maximum of students per lesson.
The maximum number of lessons is equal to the maximum number of “roomslots”, which is 145.

For the baseline, we have done the following:

- we have used the minimum number of lessons (129) as the way to divide the courses. This simplifies the solution. Later, we can optimize the distribution of courses over lessons in iterative steps or generate random solutions including 129-145 lessons.
- we have randomized the placement of the lessons over the schedule.
- we have used an even distribution of the number of students over the lessons. This simplifies the solution. Later on, we can still move students between different classes in iterative steps or generate random solutions with varying number of student numbers per lesson.
- we have randomized which students are associated with the lessons.

We ran the random algorithm 50.000 times and computed the malus points of the valid schedules (Figure 1).

The minimum number of malus points was 578, and the average was 848. Because 50.000 randomly generated schedules only reflect a very small portion of the total number of possible solutions, this is only an indication of what can be reached before we optimize the schedule. There is not enough time to go over the entire state space by doing random sampling, so the chance that we will find a “good” solution using random schedules is slim. Using some simple rules, we have already found solutions with less than 200 malus points. This does, however, limit the flexibility by which we can apply our algorithms. Therefore, we will still use the random results as a basis for our iterative algorithms.

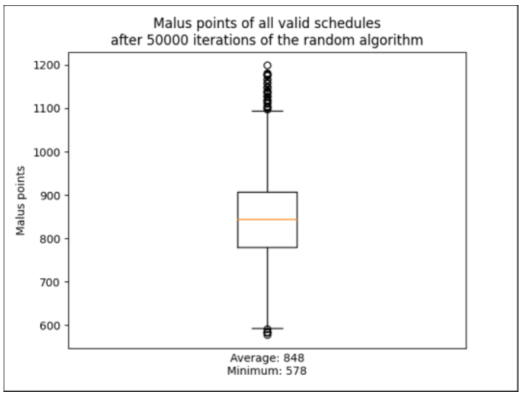

Figure 1. Boxplot showing the distribution of malus points associated with the randomly generated schedules.