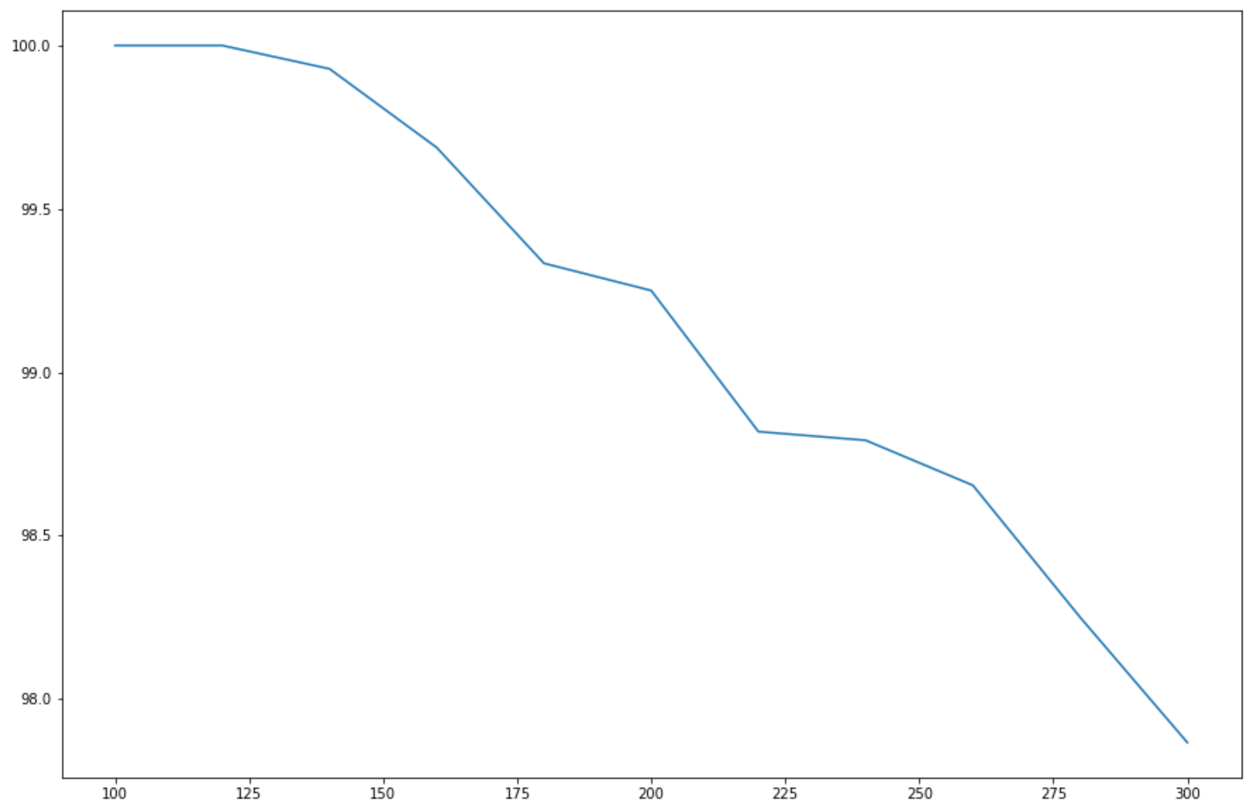
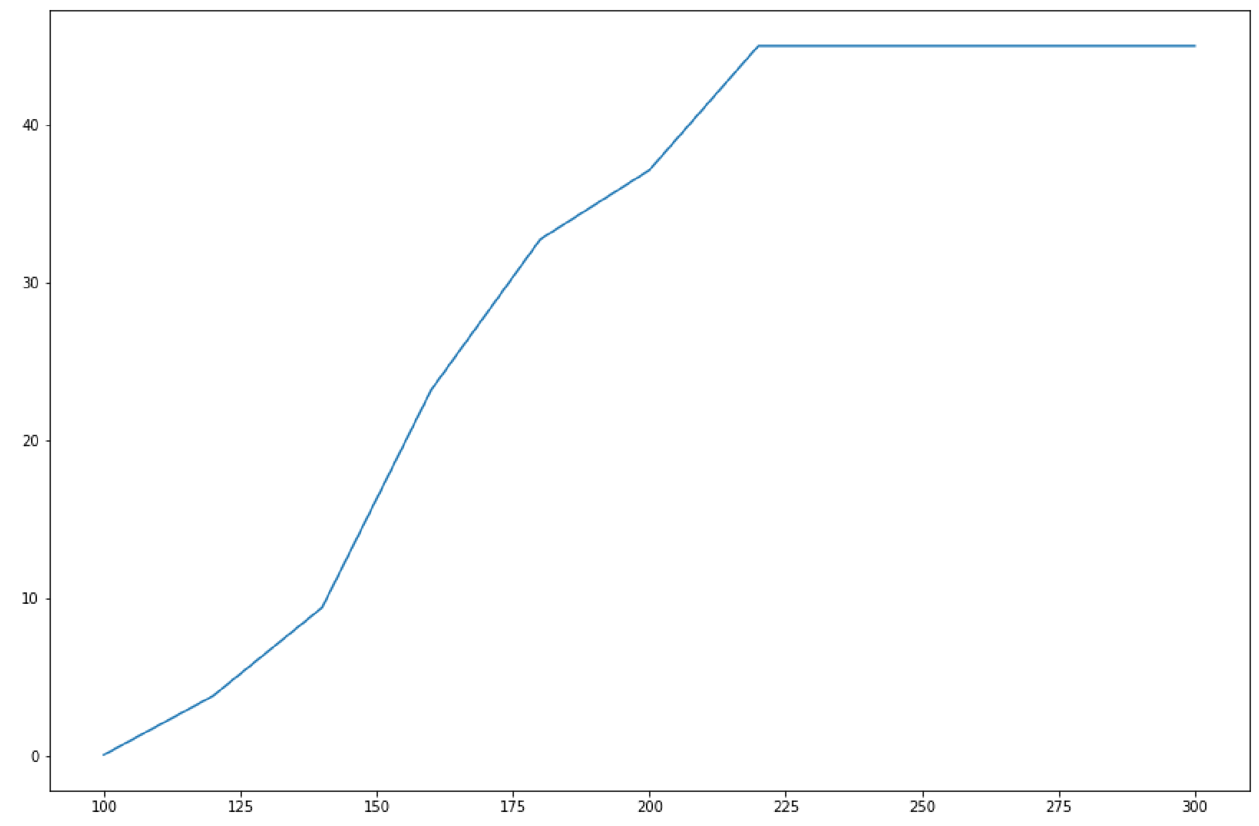
Average fitness function value of the best model found by the improved algorithm for different values of m.



Average running time of the improved GA algorithm for different values of m



3> I started off with using roulette-wheel selection for choosing the parents. I used a single point crossover. I also tried two points crossover but that did not change the efficiency by much. For mutation, I flipped multiple states randomly. For this, I used a mutation rate of 0.015. I started off with using a mutation rate of 1/max_fitness to utilise the explore first exploit next idea but it didn't seem to help my code. Then I just checked for small values of mutation rate and ended up at 0.015(1.5%). I also tried going from left to right and flipping states which improved the fitness function, which improved the efficiency but slowed down the code. After this, I used elitism to ensure that I used the best states of the previous generation as well. Culling didn't help the speed of the code as I had to generate more than 20 for every iteration.

4> If not implemented properly, the GA might fail to converge to the best possible solution. It might cause a premature convergence which means that it can get stuck at a local optimal solution instead of a global optimal solution .

5> The best possible fitness decreases with increase in the number of clauses. This is because as the number of clauses increase, the chances of conflicting clauses increase as well i.e. satisfying one clause would directly imply another clause can not be satisfied