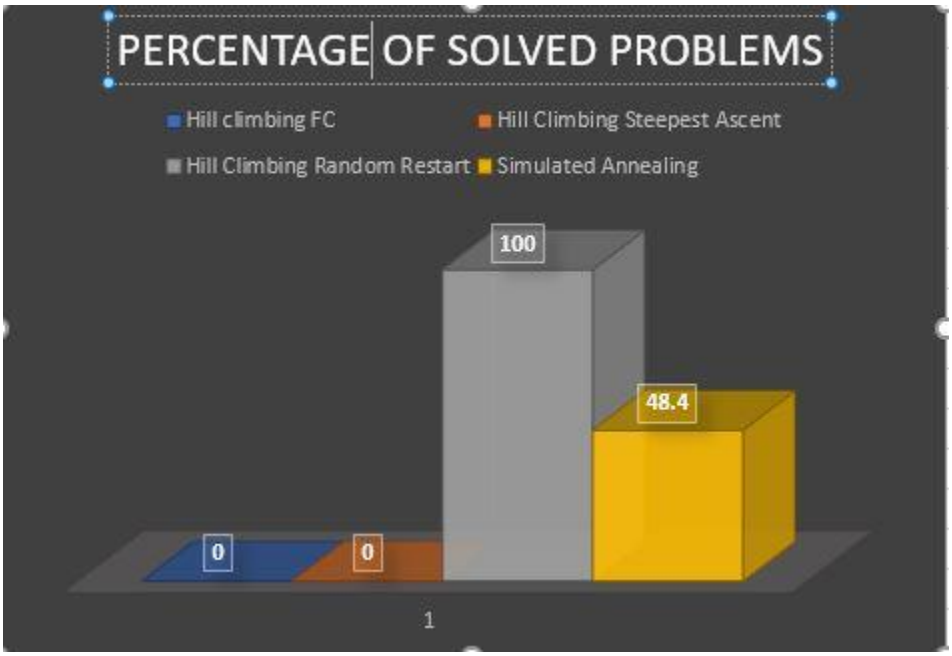
Report on using different local search algorithms for solving 8 puzzle problem

1000 different random initial states were generated and the following local search algorithms were run to solve the problem each time:

1. Hill Climbing First Choice
2. Hill Climbing Steepest Ascent
3. Hill Climbing Random Restart
4. Simulated Annealing

The percentage of problems (out of 1000) solved by each algorithm has been depicted in the graph below:



It can be noticed that the hill climbing first choice and hill climbing random restart did not solve any problem while hill climbing with random restart was able to solve all the problems.

Simulated annealing was able to solve 48.4% of the problems. I believe that this number could be increased by picking up a better initial temperature and cooling rate for the algorithm. For this experiment, initial temperature was chosen as the value given by “Integer.MAX_VALUE” in java and cooling rate was chosen to be 0.00001.

Further, search cost was also noticed for these experiments and the average search costs have been plotted in the graph below.

It can be noticed that simulated annealing has the greatest search cost but again it depends on the initial temperature and cooling rate to some extent. It can be seen even when optimized solution cost is very small, simulated annealing and hill climbing random restart take a large number of steps to reach goal state.

*Hill climbing first choice and hill climbing steepest ascent actually have infinite search cost but just for visualization simplicity(somewhat wrong), they are considered as 0.