Max Gasser
mrgasser@ucsc.edu
1684340

Report

For my implementation of Hill climb I started with creating a solution using backtracking and a hamiltonian sequence. Essentially the algorithm keeps trying nodes until a solution is found and the end is reached by visiting each node once. The hill climbing algorithm itself is more simple and works by attempting a pairwise swap. If the swap is valid and produces a total cost that's smaller than the prior path, the swap is made. This continues until each node has had a chance to swap and we have reached a local maximum.

Random Restart works in a very similar way. An initial random path is found, Hill climbing is then run on siad path, then after the algo restarts with another random path saving the path with the lowest cost as the min.

Stochastic hill climbing works very similarly to regular hill climbing. Each time a swap is made, it's either randomly chosen to be done or thrown out based on some probability that's created based on the difference between the costs of each path.

Unfortunately I wasn't able to make it to Part 2

For part 3 I used a backtracking search algorithm to find the correct order of color on the graph. The algorithm works very similarly to how I found the initial path in my part one. My stochastic hill climb seems to break on vertex 6 and 11 for some reason, it begins infinite looping while finding a random path. Before that though I was able to calculate the runtime of each algorithm. Stochastic and regular hill climbing run in about the same amount of time roughly a few milliseconds. Random hill climbing runs in about 5 seconds and sometimes even longer.