Compsci 367 A2 Report

# Table of Results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Method | # of queens | Probability of Solving (%) | Average time to solve  (mean ± sd) | Average # of nodes expanded  (mean ± sd) |
| hill climbing | 10 | 16.4 | 0.008 ± 0.003 | 4 ± 0.9 |
| hill climbing with sideways moves | 10 | 43.2 | 0.191 ± 0.164 | 86.8 ± 74.2 |
| random restart hill climbing | 10 | 99.6 | 0.057 ± 0.052 | 26.6 ± 24.6 |
| hill climbing | 11 | 14.8 | 0.015 ± 0.004 | 4.5 ± 1 |
| hill climbing with sideways moves | 11 | 44.8 | 0.291 ± 0.257 | 84.5 ± 74.3 |
| random restart hill climbing | 11 | 99.0 | 0.102 ± 0.091 | 30.3 ± 27.3 |
| hill climbing | 12 | 8.6 | 0.025 ± 0.006 | 5 ± 1 |
| hill climbing with sideways moves | 12 | 27.2 | 0.585 ± 0.348 | 113.1 ± 67.1 |
| random restart hill climbing | 12 | 87.4 | 0.333 ± 0.246 | * 1. ± 49.1 |

Parameters for algorithms:

Random Restarts = 30  
Max number of sideways moves = 150  
Number of iterations run per problem = 500

# Discussion of Results

**How you think the number of expanded nodes, the amount of time, and the probability will change as the number of queens increased?**

Overall, as the number of queens increases, the amount of time to solve increases, the probability of solving decreases and the number of nodes expanded increases.

As the number of queens increases, the number of possible board states increases. Therefore, as there are more board states to search through and each algorithm on average expands more nodes. Furthermore, a consequence of having searched through more nodes, the average solve time of the algorithms increases. Finally, as the number of board states increases, there is a greater chance of getting stuck in a local optimum and hence the probability of success decreases.

When comparing the 3 algorithms using the results table above, Random Restart Hill Climbing vastly outperforms both Hill Climbing with sideways moves and normal Hill Climbing.

**How you determined the number of queens to run?**

I initially tried a few different numbers of queens to get a feel for how quickly the algorithms ran and what the results looked like. In the end I decided using 10 to 12 queens. This is because the statistics provided show an apparent pattern and that didn’t take a very long time to run.

**How you determined the number of problems you need to run for each of them?**

I selected 500 runs for each combination of method and number of queens. This number was decided as it provided consistent results and was the largest number of iterations, I could run that didn’t take too long with my computing power.