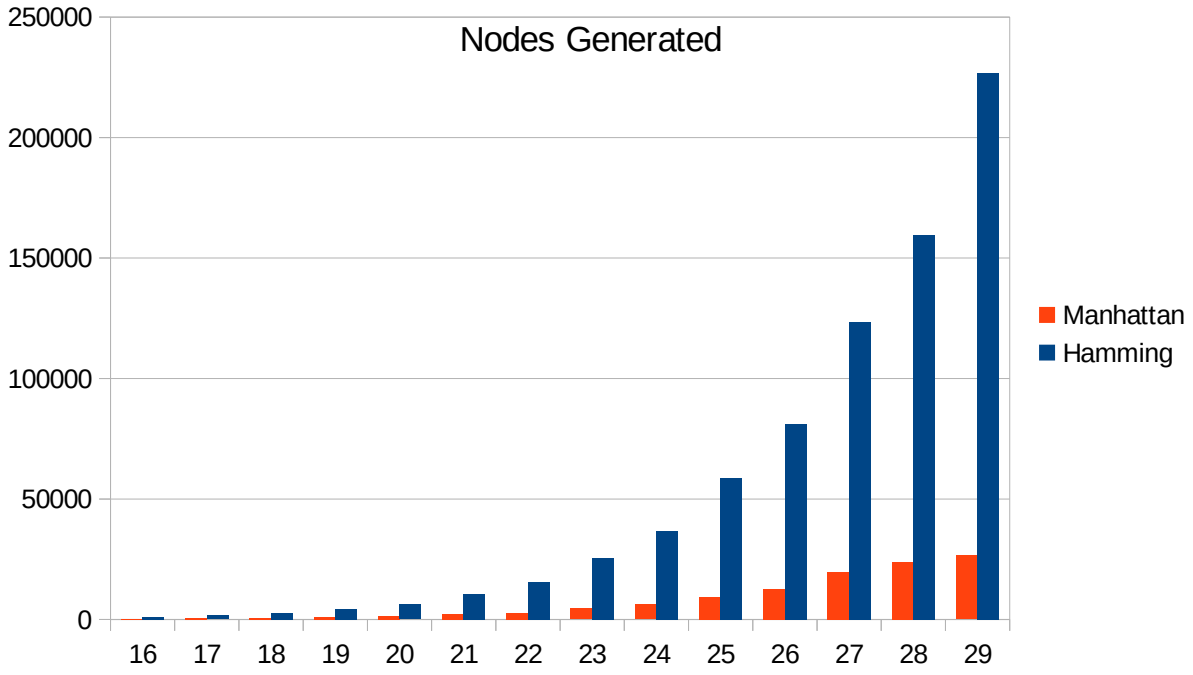
CS4200

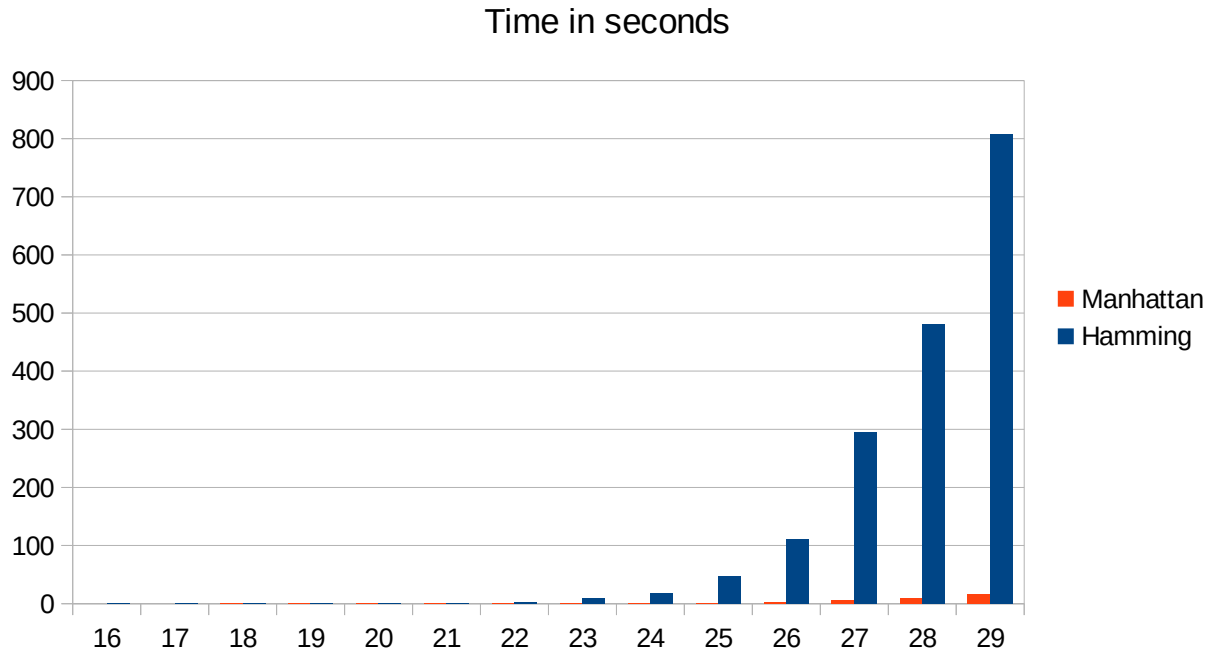
Project 1: 8-Puzzle

Ryan Skinner

For this project I implemented an A* algorithm to solve the 8-puzzle. Two different heuristics were used: Hamming, which is the number of misplaced tiles; Manhattan, the distance each number is from its goal state. The Manhattan heuristic is way more efficient because it rewards moves in the right direction. For example if you need to move a number two positions, Manhattan will recognize the first move in the right direction and be able to make the second move sooner, while Hamming will try every possible two move combination until it finds the right one.

Depth	Avg. Nodes		Avg. Time(milliseconds)		Count
	Hamming	Manhattan	Hamming	Manhattan	
7	28.00	37.00	0.01	0.01	1
8	28.00	28.00	0.01	0.01	1
10	66.33	44.67	0.04	0.02	3
11	104.50	72.50	0.07	0.04	2
12	158.50	68.50	0.13	0.04	2
13	329.00	204.00	0.53	0.22	2
14	374.29	139.86	0.73	0.13	7
15	644.80	199.60	2.18	0.26	10
16	1008.83	322.69	5.77	0.64	29
17	1691.47	528.31	17.47	1.78	32
18	2548.92	721.37	46.52	3.46	52
19	4276.06	1103.90	162.48	8.12	50
20	6229.27	1407.30	415.75	14.95	104
21	10544.86	2246.99	1360.45	42.75	100
22	15481.85	2839.68	3244.23	79.48	134
23	25581.94	4527.68	9507.32	232.12	108
24	36721.97	6215.87	17207.96	477.76	133
25	58676.04	9172.91	46417.76	1136.15	89
26	81094.45	12821.20	111473.74	2446.45	76
27	123295.88	19551.69	295786.79	5928.52	42
28	159673.59	23652.70	481543.47	9175.97	22
29	226849.00	26538.00	807486.79	15597.55	1





Putting the data into charts really shows off how much more efficient Manhattan is compared to Hamming. It almost looks linear in comparison. The data for depths 7-15 have been removed because their bars are not visible at this range.