n-Queens Heuristic Analysis

Ayushmaan Ganotra

AI 1 Pd 4

TJHSST 2016 Fall

I solved the nQueens problem as a CSP by implementing a DFS algorithm with forward propagation. The variables of the CSP are the n columns of a chessboard, and the values of each variable are the n rows. Initially my code could solve n=20 queens in under 2 minutes. But my best implementation can solve n=150 queens in under 15 seconds.

Code was implemented in Python 3.5 and tests were performed on a custom Desktop with a quad core 3.7GHz i7.

Case 1: DFS

DFS Plain

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| N | Nodes | Goal | Time (seconds) | Node/sec |
| 4 | 5 | 9 | 0.00 | 4988 |
| 5 | 6 | 13 | 0.00 | 1995 |
| 6 | 39 | 45 | 0.00 | 38891 |
| 7 | 10 | 24 | 0.00 | 2492 |
| 8 | 83 | 98 | 0.00 | 41428 |
| 9 | 66 | 87 | 0.00 | 927 |
| 10 | 2106 | 2127 | 0.07 | 87504 |
| 11 | 219 | 252 | 0.02 | 1719 |
| 12 | 2611 | 2650 | 0.13 | 9537 |
| 13 | 10133 | 10184 | 0.27 | 12792 |
| 14 | 33752 | 33805 | 0.79 | 17905 |
| 15 | 76316 | 76380 | 1.89 | 5524 |
| 16 | 369765 | 369838 | 13.81 | 64698 |
| 17 | 197170 | 197251 | 5.72 | 4988 |
| 18 | Too Slow! | 18 | 26.61 | N/A |
| 19 | 343280 | 343374 | 9.79 | N/A |
| 20 | Too Slow! | 7.944847 | 29.59 | N/A |
| 21 | Too Slow! | 7.889834 | 34.49 | N/A |
| 22 | Too Slow! | 8.009695 | 33.97 | N/A |
| 23 | Too Slow! | 8.013674 | 35.45 | N/A |
| 24 | Too Slow! | 9.751035 | 34.10 | N/A |
| 25 | Too Slow! | 7.978654 | 37.60 | N/A |
| 26 | Too Slow! | 7.990577 | 42.29 | N/A |
| 27 | Too Slow! | 8.015988 | 37.47 | N/A |
| 28 | Too Slow! | 8.098947 | 39.52 | N/A |

Case 2: DFS with Random List of Available Values

To create a faster program, I returned a random list of available values in each step of the DFS search.

DFS Plain, Min Choices

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| N | Goals | Nodes | Time (seconds) | Node/sec |
| 4 | 8 | 10 | 0.00 | N/A |
| 5 | 6 | 13 | 0.00 | N/A |
| 6 | 77 | 84 | 0.00 | 41853 |
| 7 | 58 | 69 | 0.00 | 33948 |
| 8 | 31 | 47 | 0.00 | 47014 |
| 9 | 334 | 353 | 0.01 | 58940 |
| 10 | 67 | 96 | 0.00 | 47866 |
| 11 | 237 | 272 | 0.01 | 45187 |
| 12 | 198 | 245 | 0.01 | 48889 |
| 13 | 1054 | 1106 | 0.03 | 36771 |
| 14 | 799 | 859 | 0.02 | 40792 |
| 15 | 2776 | 2839 | 0.09 | 32545 |
| 16 | 4483 | 4562 | 0.13 | 33954 |
| 17 | 1550 | 1639 | 0.05 | 35537 |
| 18 | 4377 | 4481 | 0.13 | 35752 |
| 19 | 1585 | 1696 | 0.05 | 33167 |
| 20 | 17781 | 17911 | 0.63 | 28581 |
| 21 | 6092 | 6236 | 0.22 | 28269 |
| 22 | 45996 | 46148 | 1.50 | 30745 |
| 24 | 33507 | 33698 | 1.16 | 29047 |
| 25 | 42395 | 42589 | 1.56 | 27280 |
| 26 | 30755 | 30983 | 1.14 | 27296 |
| 27 | 53249 | 53488 | 2.00 | 26713 |
| 28 | 193104 | 193352 | 7.45 | 25964 |
| 29 | 6919 | 7192 | 0.31 | 23364 |
| 30 | 4720 | 5019 | 0.22 | 22752 |
| 31 | 31 | 31 | 31 | 22645 |
| 32 | 34986 | 35325 | 1.57 | 22526 |
| 33 | 16177 | 16528 | 0.85 | 19507 |
| 34 | 736363 | 736756 | 39.02 | 18880 |
| 35 | 330081 | 330473 | 15.62 | 21159 |
| 36 | 80611 | 81045 | 4.66 | 17378 |
| 37 | 24773 | 25220 | 1.43 | 17663 |
| 38 | 131655 | 132136 | 7.78 | 16978 |
| 39 | 228589 | 229098 | 14.17 | 16167 |
| 40 | 424126 | 424665 | 24.35 | 17437 |

Case 3: DFS with Minimum Choices and Random Choices

In addition to the random list of choices, I choose the best variable in each step of the DFS search. The best variable is the one that has the minimum number of choices available. If there was a tie, the first column which has the minimum number of choices available would be selected.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| N | Goals | Nodes | Time | Node/sec |
| 4 | 5 | 9 | 0.0005 | 17984 |
| 5 | 9 | 16 | 0 | N/A |
| 6 | 22 | 29 | 0.0005 | 57948 |
| 7 | 13 | 25 | 0 | N/A |
| 8 | 48 | 64 | 0.001502 | 42622 |
| 9 | 32 | 55 | 0.001001 | 54964 |
| 10 | 242 | 263 | 0.006506 | 40422 |
| 11 | 100 | 129 | 0.003503 | 36822 |
| 12 | 253 | 293 | 0.01051 | 27877 |
| 13 | 281 | 324 | 0.015015 | 21578 |
| 14 | 210 | 264 | 0.013011 | 20289 |
| 15 | 134 | 194 | 0.01051 | 18458 |
| 16 | 59 | 136 | 0.007007 | 19410 |
| 17 | 773 | 858 | 0.035533 | 24146 |
| 18 | 29 | 126 | 0.004504 | 27972 |
| 19 | 116 | 230 | 0.013012 | 17676 |
| 20 | 50 | 173 | 0.009509 | 18193 |
| 21 | 3750 | 3877 | 0.238723 | 16240 |
| 22 | 762 | 911 | 0.050047 | 18202 |
| 23 | 11514 | 11674 | 0.511005 | 22845 |
| 24 | 412 | 591 | 0.024534 | 24088 |
| 25 | 282 | 476 | 0.022021 | 21615 |
| 26 | 2365 | 2566 | 0.111104 | 23095 |
| 27 | 22570 | 22799 | 1.126553 | 20237 |
| 28 | 9061 | 9301 | 0.434907 | 21386 |
| 29 | 641 | 900 | 0.044541 | 20206 |
| 30 | 4654 | 4936 | 0.241745 | 20418 |
| 31 | 5814 | 6117 | 0.315835 | 19367 |
| 32 | 1590 | 1910 | 0.101094 | 18893 |
| 33 | 2706 | 3047 | 0.16115 | 18907 |
| 34 | 415 | 792 | 0.050047 | 15825 |
| 35 | 8454 | 8833 | 0.475496 | 18576 |
| 36 | 36631 | 37041 | 2.068653 | 17905 |
| 37 | 9208 | 9640 | 0.557522 | 17290 |
| 38 | 38468 | 38933 | 2.362504 | 16479 |
| 39 | 223 | 722 | 0.058555 | 12330 |
| 40 | 1711 | 2228 | 0.149141 | 14938 |
| 41 | 48 | 603 | 0.057052 | 10569 |
| 42 | 15568 | 16133 | 1.044184 | 15450 |
| 43 | 10466 | 11071 | 0.735305 | 15056 |
| 44 | 1484 | 2124 | 0.183674 | 11563 |
| 46 | 13404 | 14088 | 1.072512 | 13135 |
| 47 | 2621 | 3349 | 0.263749 | 12697 |
| 48 | 6833 | 7573 | 0.540072 | 14022 |
| 49 | 9413 | 10186 | 0.784796 | 12979 |
| 50 | 15748 | 16553 | 1.294235 | 12789 |
| 51 | 3329 | 4204 | 0.400124 | 10506 |
| 52 | 66933 | 67821 | 5.189854 | 13067 |
| 53 | 19012 | 19947 | 1.533499 | 13007 |
| 54 | 101 | 1064 | 0.137631 | 7730 |
| 56 | 11256 | 12279 | 1.061584 | 11566 |
| 57 | 633890 | 634959 | 49.00294 | 12957 |
| 58 | 33676 | 34792 | 2.914638 | 11936 |
| 59 | 35316 | 36482 | 3.017383 | 12090 |
| 60 | 160446 | 161617 | 13.60855 | 11876 |
| 61 | 2493 | 3722 | 0.424463 | 8768 |
| 62 | 536 | 1852 | 0.273256 | 6777 |
| 63 | 64166 | 65460 | 5.860193 | 11170 |
| 64 | 174 | 1554 | 0.260881 | 5956 |
| 67 | 37678 | 39156 | 3.808197 | 10282 |
| 69 | 115284 | 116868 | 11.15523 | 10476 |
| 70 | 475 | 2129 | 0.493461 | 4314 |
| 73 | 38210 | 39985 | 4.27847 | 9345 |
| 76 | 3702 | 5608 | 0.904073 | 6203 |
| 77 | 4012 | 6004 | 0.988016 | 6076 |
| 79 | 681 | 2772 | 0.745848 | 3716 |
| 82 | 16341 | 18629 | 2.752132 | 6768 |
| 85 | 180 | 2577 | 0.841322 | 3063 |
| 86 | 567660 | 570167 | 67.77452 | 8412 |
| 87 | 217991 | 220507 | 25.50257 | 8646 |
| 89 | 218715 | 221392 | 27.01042 | 8196 |
| 90 | 254439 | 257174 | 31.60701 | 8136 |
| 93 | 21661 | 24566 | 4.055701 | 6057 |
| 94 | 51286 | 54250 | 7.861824 | 6900 |
| 95 | 461763 | 464842 | 60.81931 | 7642 |
| 96 | 299 | 3397 | 1.458767 | 2328 |
| 97 | 191 | 3342 | 1.670422 | 2000 |
| 98 | 36820 | 40061 | 6.815042 | 5878 |

Since a random choice and random column was selected, there isn’t a very strong correlation. The major bottle neck in this is the many nodes that are made that are not needed.

Case 4: DFS with Minimum Choices and Random Choices (Best Case)

In addition to the random list of choices, I choose the best variable in each step of the DFS search. The best variable is the one that has the minimum number of choices available. If there was a tie, the first column which has the minimum number of choices available would be selected. I also attempted to do least constraining values, but it made my code slower so I discarded it.

DFS Plain, Min Choices, Random Choices

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| N | Goals | Nodes | Time | Node/sec |
| 4 | 9 | 10 | 0.00 | N\A |
| 5 | 6 | 12 | 0.00 | N\A |
| 6 | 13 | 19 | 0.00 | N\A |
| 7 | 8 | 18 | 0.00 | 36002.61 |
| 8 | 16 | 29 | 0.00 | 57948.94 |
| 9 | 23 | 39 | 0.00 | 38937.84 |
| 10 | 28 | 48 | 0.00 | 31850.43 |
| 11 | 15 | 43 | 0.00 | 43209.17 |
| 12 | 18 | 49 | 0.00 | 32617.19 |
| 13 | 32 | 70 | 0.00 | 27983.35 |
| 14 | 48 | 94 | 0.00 | 23465.34 |
| 15 | 25 | 78 | 0.00 | 22273.67 |
| 16 | 17 | 81 | 0.00 | 23078.5 |
| 17 | 38 | 107 | 0.00 | 23758.1 |
| 18 | 38 | 118 | 0.01 | 16829.7 |
| 19 | 22 | 108 | 0.00 | 30827.88 |
| 20 | 21 | 122 | 0.00 | 30471.33 |
| 21 | 90 | 201 | 0.01 | 26778.96 |
| 22 | 25 | 146 | 0.01 | 13173.46 |
| 23 | 144 | 277 | 0.02 | 16278.63 |
| 24 | 1119 | 1262 | 0.05 | 27114.36 |
| 25 | 51 | 211 | 0.01 | 23423.18 |
| 26 | 69 | 236 | 0.01 | 24817.74 |
| 27 | 316 | 504 | 0.02 | 23977.51 |
| 28 | 88 | 286 | 0.01 | 20412.32 |
| 29 | 50 | 257 | 0.01 | 21394.41 |
| 30 | 31 | 265 | 0.02 | 17645.79 |
| 31 | 221 | 474 | 0.03 | 14572.84 |
| 32 | 190 | 447 | 0.02 | 18608.97 |
| 33 | 50 | 326 | 0.03 | 10339.94 |
| 34 | 78 | 382 | 0.03 | 13909.89 |
| 35 | 201 | 521 | 0.03 | 16790.75 |
| 36 | 97 | 428 | 0.03 | 14744.17 |
| 37 | 728 | 1083 | 0.08 | 12804.74 |
| 38 | 127 | 501 | 0.04 | 13171.86 |
| 39 | 91 | 489 | 0.04 | 12526.73 |
| 40 | 56 | 475 | 0.05 | 10429.82 |
| 41 | 48 | 503 | 0.04 | 11686.66 |
| 42 | 54 | 521 | 0.07 | 7711.56 |
| 43 | 71 | 570 | 0.06 | 10353.7 |
| 44 | 297 | 808 | 0.08 | 10024.34 |
| 45 | 46 | 586 | 0.06 | 10271.05 |
| 46 | 543 | 1105 | 0.11 | 9990.614 |
| 47 | 66 | 658 | 0.08 | 8427.878 |
| 48 | 461 | 1077 | 0.10 | 10653.43 |
| 49 | 90 | 736 | 0.08 | 9191.353 |
| 50 | 69 | 738 | 0.17 | 4337.142 |
| 51 | 169 | 867 | 0.12 | 7467.159 |
| 52 | 351 | 1085 | 0.12 | 9344.753 |
| 53 | 406 | 1157 | 0.13 | 8823.82 |
| 54 | 84 | 872 | 0.12 | 7082.86 |
| 55 | 71 | 885 | 0.16 | 5704.361 |
| 56 | 177 | 1024 | 0.13 | 7721.151 |
| 57 | 64 | 936 | 0.15 | 6234.179 |
| 58 | 87 | 996 | 0.20 | 5088.953 |
| 59 | 408 | 1354 | 0.18 | 7412.278 |
| 60 | 202 | 1163 | 0.22 | 5404.264 |
| 61 | 727 | 1745 | 0.35 | 5024.067 |
| 62 | 83 | 1116 | 0.22 | 5068.014 |
| 63 | 90 | 1173 | 0.22 | 5265.577 |
| 64 | 331 | 1454 | 0.28 | 5270.693 |
| 65 | 76 | 1223 | 0.25 | 4977.018 |
| 66 | 525 | 1714 | 0.32 | 5326.199 |
| 67 | 68 | 1299 | 0.32 | 4106.936 |
| 68 | 209 | 1483 | 0.42 | 3527.65 |
| 69 | 8361 | 9681 | 1.13 | 8604.217 |
| 70 | 91 | 1429 | 0.34 | 4230.119 |
| 71 | 151 | 1539 | 0.39 | 3978.157 |
| 72 | 76 | 1485 | 0.39 | 3802.672 |
| 73 | 331 | 1787 | 0.46 | 3868.252 |
| 74 | 90 | 1585 | 0.42 | 3751.36 |
| 75 | 76 | 1622 | 0.53 | 3037.364 |
| 76 | 327 | 1924 | 0.50 | 3866.614 |
| 77 | 879 | 2500 | 0.65 | 3839.25 |
| 78 | 79 | 1731 | 0.48 | 3616.514 |
| 79 | 84 | 1796 | 0.68 | 2660.076 |
| 80 | 81 | 1831 | 0.57 | 3211.36 |
| 81 | 308 | 2095 | 0.73 | 2888.728 |
| 82 | 119 | 1943 | 0.59 | 3301.17 |
| 83 | 99 | 1997 | 0.61 | 3257.354 |
| 84 | 3563 | 5509 | 1.22 | 4507.13 |
| 85 | 213 | 2183 | 0.73 | 2985.077 |
| 86 | 802 | 2830 | 0.97 | 2908.34 |
| 87 | 88 | 2165 | 0.86 | 2526.466 |
| 88 | 218 | 2319 | 0.87 | 2679.485 |
| 89 | 205 | 2375 | 0.86 | 2753.101 |
| 90 | 91 | 2290 | 0.93 | 2472.866 |
| 91 | 92 | 2380 | 1.05 | 2266.554 |
| 92 | 107 | 2450 | 0.98 | 2491.282 |
| 93 | 94 | 2491 | 1.39 | 1791.973 |
| 94 | 108 | 2565 | 1.15 | 2230.287 |
| 95 | 1009 | 3536 | 1.54 | 2293.987 |
| 96 | 97 | 2621 | 1.41 | 1855.73 |
| 97 | 513 | 3135 | 1.55 | 2029.101 |
| 98 | 133 | 2781 | 1.46 | 1906.268 |
| 99 | 224 | 2934 | 1.44 | 2030.798 |

In the chart below, you can see that the major bottleneck is the fact that many nodes are being made which are not needed.