Alexander Grzesiak

CS 470

Project 2: Scroggle

February 27, 2017

**Solution approach:**

The meat of the program is the scroggle() function which essentially does the same thing for depth first, breadth first, and A\*. The only difference is an if statement which dictates which end to take a path from. There is also another conditional that either just appends a path to the frontier or sends it off to heuristic1() to get the ‘goodness’. Other than that there are some helper functions for printing, finding all valid potential paths from some starting path, and a findBestConstants() function that helped me in my heuristic.

**My heuristic function:**

The inspiration for my heuristic function came in class when we were working through the A\* example using the map of Romania where the ‘goodness’ came from the table of absolute shortest distance from one city to another. Those values were exact distances but that doesn’t work for Scroggle, so I had to use averages instead. While the dictionary is getting loaded, I decided to extract the (1) average number of words with a certain prefix, (2) average number of letters after a given prefix, and (3) the average score of a word starting with a given prefix. At first, I started to make three different heuristic functions, one with each of my pieces of meta-data. But, at some point I realized that if I incorporated all of these into one and gave them constants to weight some more heavily more than others, I could use all three as a heuristic trifecta. Before I brute-forced my way to the constants, my thinking was the more words with a given prefix, the less letters after a given prefix, and the highest score of a word with a given prefix would enable my Scroggle solver to select the very best candidates for expansion. So, this is where my findBestConstants() came in. I iterated through a bunch of randomly generated boards and tried millions of variations of the constants across various expansion limits. This took about 4 hours but at the end I had the optimal average constants for each expansion limit. After finding the constants, there was a weird result. At lower expansion limits (<10), the average number of letters after a prefix and the average score of a word with a prefix played more of a role than number of words with a certain prefix. Then, at expansion limits from about 10-40, the number of words with a certain prefix took precedence over average number of letters after a prefix and average score of a word with a prefix. After about the 60 expansion limit, it seems that the constants don’t play as much of a role as they did at lower limits. I can reason this because as the expansion limit grew, the constants started to converge to the lower limits I set in the findBestConstants() so if the expansion limit was greater than or equal to 60 I just used the constants for the 59th expansion.

**Random 3x3 Stats:**

Board: ['p', 'm', 'i', 'n', 'b', 'o', 'l', 'u', 'r']

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Depth First Search | | | | | Breadth First Search | | | | | A\* | | | | |
| # Expansions | 10 | 50 | 100 | 500 | 1000 | 10 | 50 | 100 | 500 | 1000 | 10 | 50 | 100 | 500 | 1000 |
| # Words | 0 | 0 | 0 | 3 | 5 | 0 | 9 | 13 | 27 | 28 | 2 | 9 | 16 | 28 | 28 |
| Avg. Frontier | 15.5 | 11.9 | 11.5 | 9.89 | 9.6 | 26 | 87.6 | 153 | 502 | 814 | 18.6 | 59 | 104 | 51.9 | 51.8 |
| Max Frontier | 21 | 21 | 21 | 21 | 21 | 43 | 141 | 278 | 896 | 1477 | 25 | 114 | 210 | 292 | 292 |
| Avg. Depth | 5.3 | 6.8 | 7 | 7 | 7 | 1.1 | 1.84 | 2.42 | 3.5 | 4.02 | 3.6 | 3.1 | 3.21 | 3.8 | 3.8 |
| Avg. Branching | 2.1 | 1.14 | 1.1 | 1 | 1 | 4.4 | 4.04 | 3.7 | 2.7 | 2.47 | 2.6 | 3.1 | 3.01 | .97 | .97 |
| Total Word Score | 0 | 0 | 0 | 18 | 26 | 0 | 28 | 54 | 121 | 129 | 8 | 40 | 68 | 129 | 129 |

Board: ['t', 'x', 'j', 'a', 'n', 'e', 's', 'u', 'c']

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Depth First Search | | | | | Breadth First Search | | | | | A\* | | | | |
| # Expansions | 10 | 50 | 100 | 500 | 1000 | 10 | 50 | 100 | 500 | 1000 | 10 | 50 | 100 | 500 | 1000 |
| # Words | 0 | 0 | 0 | 0 | 0 | 0 | 4 | 6 | 8 | 8 | 1 | 7 | 8 | 8 | 8 |
| Avg. Frontier | 15.5 | 11.9 | 11.5 | 9.89 | 9.6 | 26 | 87.6 | 153.5 | 502 | 814 | 23.7 | 71.2 | 49.6 | 22.6 | 22.6 |
| Max Frontier | 21 | 21 | 21 | 21 | 21 | 43 | 141 | 278 | 896 | 1477 | 36 | 143 | 161 | 161 | 161 |
| Avg. Depth | 5.3 | 6.8 | 7 | 7 | 7 | 1.1 | 1.84 | 2.42 | 3.5 | 4.02 | 2.1 | 2.4 | 2.95 | 3.1 | 3.1 |
| Avg. Branching | 2.1 | 1.14 | 1.1 | 1 | 1 | 4.4 | 4.04 | 3.7 | 2.7 | 2.47 | 3.7 | 3.7 | 2.1 | .95 | .95 |
| Total Word Score | 0 | 0 | 0 | 0 | 0 | 0 | 18 | 32 | 50 | 50 | 8 | 45 | 50 | 50 | 50 |

Board: ['i', 'm', 'l', 'i', 'w', 'c', 'f', 'f', 'q']

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Depth First Search | | | | | Breadth First Search | | | | | A\* | | | | |
| # Expansions | 10 | 50 | 100 | 500 | 1000 | 10 | 50 | 100 | 500 | 1000 | 10 | 50 | 100 | 500 | 1000 |
| # Words | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 2 | 5 | 5 | 0 | 5 | 5 | 5 | 5 |
| Avg. Frontier | 15.5 | 11.9 | 11.5 | 9.89 | 9.6 | 26 | 87.6 | 153.5 | 502 | 814 | 25.4 | 60.7 | 30.7 | 18.4 | 18.4 |
| Max Frontier | 21 | 21 | 21 | 21 | 21 | 43 | 141 | 278 | 896 | 1477 | 36 | 123 | 123 | 123 | 123 |
| Avg. Depth | 5.3 | 6.8 | 7 | 7 | 7 | 1.1 | 1.84 | 2.42 | 3.5 | 4.02 | 2.1 | 2.42 | 2.87 | 2.9 | 2.9 |
| Avg. Branching | 2.1 | 1.14 | 1.1 | 1 | 1 | 4.4 | 4.04 | 3.7 | 2.7 | 2.47 | 3.7 | 3.16 | 1.58 | .94 | .94 |
| Total Word Score | 0 |  | 0 | 0 | 0 | 0 | 18 | 18 | 70 | 70 | 0 | 70 | 70 | 70 | 70 |

**Random 4x4 stats:**

Board: ['k', 'o', 'f', 't', 'p', 'n', 'p', 'o', 'j', 'y', 'e', 'w', 'm', 'i', 'v', 's']

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Depth First Search | | | | | Breadth First Search | | | | | A\* | | | | |
| # Expansions | 10 | 50 | 100 | 500 | 1000 | 10 | 50 | 100 | 500 | 1000 | 10 | 50 | 100 | 500 | 1000 |
| # Words | 1 | 1 | 1 | 1 | 1 | 0 | 7 | 16 | 51 | 65 | 2 | 19 | 36 | 94 | 94 |
| Avg. Frontier | 28.4 | 26.4 | 25.5 | 23.7 | 23.5 | 35.6 | 117 | 214 | 901 | 1691 | 31.7 | 104.5 | 184 | 299 | 149 |
| Max Frontier | 42 | 44 | 44 | 44 | 44 | 61 | 220 | 408 | 1732 | 3269 | 49 | 191 | 349 | 993 | 993 |
| Avg. Depth | 5.5 | 11.5 | 12.8 | 13.6 | 13.6 | 1 | 1.68 | 1.84 | 2.77 | 3.38 | 3 | 3 | 3.14 | 3.71 | 4 |
| Avg. Branching | 3.6 | 1.48 | 1.24 | 1.04 | 1.02 | 5.5 | 5.08 | 4.92 | 4.43 | 4.3 | 4.3 | 4.5 | 4.33 | 2.54 | 1.27 |
| Total Word Score | 7 | 7 | 7 | 7 | 7 | 0 | 25 | 71 | 340 | 449 | 16 | 189 | 310 | 775 | 775 |

Board: ['s', 'h', 'p', 'x', 't', 'k', 'n', 'e', 'l', 'a', 'y', 'f', 'z', 't', 'o', 'h']

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Depth First Search | | | | | Breadth First Search | | | | | A\* | | | | |
| # Expansions | 10 | 50 | 100 | 500 | 1000 | 10 | 50 | 100 | 500 | 1000 | 10 | 50 | 100 | 500 | 1000 |
| # Words | 0 | 0 | 0 | 0 | 0 | 0 | 5 | 24 | 53 | 60 | 4 | 18 | 29 | 104 | 104 |
| Avg. Frontier | 28.4 | 26.4 | 25.5 | 23.7 | 23.5 | 35.6 | 117 | 214 | 901 | 1691 | 33.1 | 105 | 190 | 350 | 175 |
| Max Frontier | 42 | 44 | 44 | 44 | 44 | 61 | 220 | 408 | 1732 | 3269 | 52 | 188 | 356 | 1077 | 1077 |
| Avg. Depth | 5.5 | 11.5 | 12.8 | 13.6 | 13.6 | 1 | 1.68 | 1.84 | 2.77 | 3.38 | 2 | 2.94 | 3.03 | 3.9 | 4.07 |
| Avg. Branching | 3.6 | 1.48 | 1.24 | 1.04 | 1.02 | 5.5 | 5.08 | 4.92 | 4.43 | 4.3 | 4.6 | 4.4 | 4.4 | 2.76 | 1.3 |
| Total Word Score | 0 | 0 | 0 | 0 | 0 | 0 | 19 | 107 | 219 | 375 | 24 | 174 | 254 | 837 | 837 |

Board: ['j', 'e', 'l', 'c', 'd', 't', 'c', 'd', 'i', 'r', 'p', 'i', 'n', 'p', 'f', 'k']

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Depth First Search | | | | | Breadth First Search | | | | | A\* | | | | |
| # Expansions | 10 | 50 | 100 | 500 | 1000 | 10 | 50 | 100 | 500 | 1000 | 10 | 50 | 100 | 500 | 1000 |
| # Words | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 10 | 29 | 29 | 0 | 12 | 31 | 42 | 42 |
| Avg. Frontier | 28.4 | 26.4 | 25.5 | 23.7 | 23.5 | 35.6 | 117 | 214 | 901 | 1691 | 35.6 | 109 | 195.5 | 97.7 | 66.6 |
| Max Frontier | 42 | 44 | 44 | 44 | 44 | 61 | 220 | 408 | 1732 | 3269 | 57 | 200 | 367 | 570 | 570 |
| Avg. Depth | 5.5 | 11.5 | 12.8 | 13.6 | 13.6 | 1 | 1.68 | 1.84 | 2.77 | 3.38 | 1.9 | 2.42 | 2.8 | 3.9 | 3.75 |
| Avg. Branching | 3.6 | 1.48 | 1.24 | 1.04 | 1.02 | 5.5 | 5.08 | 4.92 | 4.43 | 4.3 | 5.1 | 4.7 | 4.5 | 1.4 | .97 |
| Total Word Score | 0 | 0 | 0 | 0 | 0 | 0 | 11 | 43 | 181 | 181 | 0 | 100 | 236 | 312 | 312 |

**Analysis:**

My tables don’t show this, but when running the three search strategies, I did run into cases where depth first search would beat A\*. It happened once with a board that only had one vowel so my heuristic didn’t prove as strong in deciding which prefix would go next since there were no situations where it would go past 2 letters. Also it didn’t help that the ‘e’ was one of the first letters expanded in BFS so it got lucky. There were about 5 possible words on that board and the BFS hit all of them within 50 expansions whereas it took A\* within 100 expansions. Other than abnormal cases like this, A\* beat BFS and DFS consistently. Even when I turned dumbness off for DFS and BFS (pruning turned on), A\* still scored higher in fewer turns. Without pruning, DFS is awful. Since it explores many hopeless paths, even 1000 expansions were often not enough to find a single word. Since most of the found words are shorter, BFS gave A\* a run for its money since it would explore shortest paths first. This, however, is also taken into account in my heuristic in the form of average number of letters after the prefix with some associated constant based on the expansion limit so the edge is given to both.

When comparing the memory usage of the search methods, whether or not ‘dumbness’ is turned on for BFS is crucial. If pruning is disabled, like it was in the tests above, the memory usage for A\* is significantly reduced from that of BFS. Since BFS, by definition, has to search those nodes closest to the root first, its frontier can get massive. This differs from A\* because unlike BFS, A\* has the option to explore a path straight down the tree as long as the heuristic determines it is the best path to take. Because of this freedom, the lower bound of the memory usage for A\* is the upper bound for DFS and the upper bound for A\* is the upper bound of BFS (Ω(bm), O(bd+1) where b=branching factor, m = max tree depth, and d = goal depth). In a test case with a 4x4 board with unlimited expansions, DFS, BFS, and A\* has a max frontier size of 44, 2645352, and 504 respectively. This is completely dependent on the board but the couple fully exhaustive tests I ran had the memory usage much closer to DFS than BFS. Going back to the dumbness, if it is turned off meaning pruning is allowed, A\* had the higher max/average frontier size. Since it is allowed to jump around the board, it might not have the possibility to prune a particular branch as quickly as DFS does.

Comparing the runtime of the search algorithms, with dumbness turned ON and expansions limited to less than 10000, BFS used about twice the amount of time as DFS with A\* following far behind at roughly 15 times slower than BFS (on a 4x4 board). In my test case with an unlimited amount of expansions on a 4x4 board, the time taken by DFS and BFS was 90 and 110 seconds respectively with A\* coming in at 174 seconds. When it comes to A\*, a major part of its time is taken sorting the frontier after every round of expansions. Pythons sorted() function is O(nlog(n)). Using the logic from before in the memory usage, the lower bound for A\* is the upper bound for DFS and the upper bound for A\* is the upper bound for BFS with the sorted function being factored into both (Ω(bmnlog(n)), O(bd+1nlog(n)) where b=branching factor, m=max tree depth, d=goal depth, and n=number of nodes in the frontier). In our Scroggle case, the max tree depth is the goal depth since we have to search every path to ensure there isn’t a goal down each path so the worst case scenario is only b times better than the best case. Because of this, we can just neglect Ω and say the runtime is O(nd+1nlog(n)).

The results of the search functions are clear. A\* produces a higher score almost all the time while improving the memory usage over BFS but sacrificing the time taken to find a higher score. An idea to cut down on sorting operations would be to only sort every other expansion causing the two best paths to be expanded before sorting. This might decrease the score but could help mitigate the time taken by A\*. I should also mention, my heuristic attempts to produce the highest score it can, not necessarily the most words.

**Output: Original 3x3**

Board:   
['y', 'q', 'i', 't', 'b', 'g', 'e', 'r', 'o']  
Dumbness turned ON

Search Type: Depth First Search

Number of expansions: 100  
0 Words found: []  
Average frontier size: 11.465346534653465  
Max frontier size: 21  
Average depth: 7.01  
Average branching factor: 1.09  
Total score: 0  
Runtime: 0.0007113235433493514

Search Type: Breadth First Search

Number of expansions: 100  
10 Words found: ['be', 'bi', 'bo', 'by', 'er', 'et', 'go', 'or', 'qi', 're']  
Average frontier size: 153.47524752475246  
Max frontier size: 278  
Average depth: 2.42  
Average branching factor: 3.69  
Total score: 41  
Runtime: 0.0009877074480204584

Search Type: A\* using h1()

Number of expansions: 100  
35 Words found: ['be', 'berg', 'bet', 'bi', 'big', 'bo', 'bog', 'bore', 'bort', 'borty', 'bro', 'by', 'byte', 'er', 'erg', 'ergo', 'et', 'gib', 'gibe', 'giber', 'go', 'gob', 'goby', 'gor', 'gore', 'obe', 'obi', 'ogre', 'or', 'orb', 'orby', 'ore', 'ort', 'qi', 'trog']  
Average frontier size: 110.73267326732673  
Max frontier size: 196  
Average depth: 3.07  
Average branching factor: 2.87  
Total score: 194  
Runtime: 0.0018847384342679518

**Output: Original 4x4**

Board:   
['v', 'b', 'r', 'p', 'w', 'w', 'l', 'j', 'g', 'f', 'r', 'i', 'z', 'i', 'd', 'u']  
Dumbness turned ON

Search Type: Depth First Search

Number of expansions: 50  
0 Words found: []  
Average frontier size: 11.862745098039216  
Max frontier size: 21  
Average depth: 6.78  
Average branching factor: 1.14  
Total score: 0  
Runtime: 0.00036424273627755355

Search Type: Breadth First Search

Number of expansions: 50  
10 Words found: ['be', 'bi', 'bo', 'by', 'er', 'et', 'go', 'or', 'qi', 're']  
Average frontier size: 87.56862745098039  
Max frontier size: 161  
Average depth: 1.84  
Average branching factor: 4.04  
Total score: 41  
Runtime: 0.0006598377557320845

Search Type: A\* using h1()

Number of expansions: 50  
17 Words found: ['gib', 'gibe', 'giber', 'go', 'gob', 'goby', 'gor', 'gore', 'obe', 'obi', 'ogre', 'or', 'orb', 'orby', 'ore', 'ort', 'qi']  
Average frontier size: 60.21568627450981  
Max frontier size: 111  
Average depth: 2.88  
Average branching factor: 3.04  
Total score: 97  
Runtime: 0.0008376046045199814

**Output: Dynamic 2x2**

Board: ['h', 'm', 'n', 't']  
Dumbness turned ON

Search Type: Depth First Search

Number of expansions: 30  
1 Words found: ['nth']  
Average frontier size: 3.3870967741935485  
Max frontier size: 7  
Average depth: 3.033333333333333  
Average branching factor: 0.9666666666666667  
Total score: 6  
Runtime: 0.0001698262546776519

Search Type: Breadth First Search

Number of expansions: 30  
1 Words found: ['hm']  
Average frontier size: 19.29032258064516  
Max frontier size: 24  
Average depth: 2.3333333333333335  
Average branching factor: 1.6666666666666667  
Total score: 7  
Runtime: 0.0001972341117673082

Search Type: A\* using h1()

Number of expansions: 30  
2 Words found: ['hm', 'nth']  
Average frontier size: 9.483870967741936  
Max frontier size: 18  
Average depth: 2.7333333333333334  
Average branching factor: 1.2666666666666666  
Total score: 13  
Runtime: 0.00024667071380779504

**Output: Dynamic 3x3**

['y', 't', 'd', 'h', 'i', 'g', 'm', 'x', 'o']

Search Type: Depth First Search  
Dumbness turned ON

Number of expansions: 50  
0 Words found: []  
Average frontier size: 11.862745098039216  
Max frontier size: 21  
Average depth: 6.78  
Average branching factor: 1.14  
Total score: 0  
Runtime: 0.0003944682235355046

Search Type: Breadth First Search

Number of expansions: 50  
10 Words found: ['go', 'hi', 'hm', 'id', 'it', 'mi', 'oi', 'ox', 'ti', 'xi']  
Average frontier size: 87.56862745098039  
Max frontier size: 161  
Average depth: 1.84  
Average branching factor: 4.04  
Total score: 46  
Runtime: 0.0004889869082660425

Search Type: A\* using h1()

Number of expansions: 50  
17 Words found: ['gid', 'git', 'go', 'gox', 'hi', 'hid', 'him', 'hit', 'hm', 'id', 'it', 'oi', 'ox', 'oxid', 'oxim', 'xi', 'yid']  
Average frontier size: 77.07843137254902  
Max frontier size: 140  
Average depth: 2.42  
Average branching factor: 3.62  
Total score: 113  
Runtime: 0.0009510782558352915

**Output: Dynamic 4x4**

Board: ['n', 'p', 't', 'v', 'p', 'l', 'f', 'h', 'g', 'a', 't', 'i', 'l', 'l', 'a', 'r']  
Dumbness turned ON

Search Type: Depth First Search

Number of expansions: 120  
0 Words found: []  
Average frontier size: 25.165289256198346  
Max frontier size: 44  
Average depth: 13.15  
Average branching factor: 1.1666666666666667  
Total score: 0  
Runtime: 0.001118599176740176

Search Type: Breadth First Search

Number of expansions: 120  
14 Words found: ['aa', 'ag', 'ai', 'al', 'ar', 'at', 'fa', 'hi', 'if', 'it', 'la', 'pa', 'ta', 'ti']  
Average frontier size: 253.11570247933884  
Max frontier size: 483  
Average depth: 2.033333333333333  
Average branching factor: 4.891666666666667  
Total score: 58  
Runtime: 0.0016739284680546973

Search Type: A\* using h1()

Number of expansions: 120  
29 Words found: ['gal', 'gala', 'gall', 'gap', 'gat', 'glair', 'la', 'laari', 'lag', 'lair', 'laith', 'lap', 'lar', 'lari', 'lat', 'lath', 'lathi', 'lati', 'latria', 'rai', 'raita', 'rat', 'ratal', 'rath', 'ria', 'rial', 'riata', 'rif', 'rift']  
Average frontier size: 208.2479338842975  
Max frontier size: 384  
Average depth: 3.691666666666667  
Average branching factor: 4.066666666666666  
Total score: 211  
Runtime: 0.004767174096245164

**Output: Dynamic 2x2 with frontier**

Board: ['h', 'm', 'n', 't']  
Dumbness turned ON

Node Format: [word, [path], g(n), h(n)]  
h(n) is optional is this case. Single letter words have it by default.

Search Type: Depth First Search

Frontier(1):

['h', [[0, 0]], 4, -6801.448370239285], ['m', [[1, 0]], 3, -10962.93046359313]  
['n', [[0, 1]], 1, -4789.033130727763], ['th', [[1, 1], [0, 0]], 5]  
['tn', [[1, 1], [0, 1]], 2], ['tm', [[1, 1], [1, 0]], 4]

Frontier(2):

['h', [[0, 0]], 4, -6801.448370239285], ['m', [[1, 0]], 3, -10962.93046359313]  
['n', [[0, 1]], 1, -4789.033130727763], ['th', [[1, 1], [0, 0]], 5]  
['tn', [[1, 1], [0, 1]], 2], ['tmh', [[1, 1], [1, 0], [0, 0]], 8]  
['tmn', [[1, 1], [1, 0], [0, 1]], 5]

Frontier(3):

['h', [[0, 0]], 4, -6801.448370239285], ['m', [[1, 0]], 3, -10962.93046359313]  
['n', [[0, 1]], 1, -4789.033130727763], ['th', [[1, 1], [0, 0]], 5]  
['tn', [[1, 1], [0, 1]], 2], ['tmh', [[1, 1], [1, 0], [0, 0]], 8]  
['tmnh', [[1, 1], [1, 0], [0, 1], [0, 0]], 9]

Frontier(4):

['h', [[0, 0]], 4, -6801.448370239285]  
['m', [[1, 0]], 3, -10962.93046359313]  
['n', [[0, 1]], 1, -4789.033130727763]  
['th', [[1, 1], [0, 0]], 5]  
['tn', [[1, 1], [0, 1]], 2]  
['tmhn', [[1, 1], [1, 0], [0, 0], [0, 1]], 9]

Number of expansions: 30  
1 Words found: ['nth']  
Average frontier size: 3.3870967741935485  
Max frontier size: 7  
Average depth: 3.033333333333333  
Average branching factor: 0.9666666666666667  
Total score: 6  
Runtime: 0.00031378154144867665

Search Type: Breadth First Search

Node Format: [word, [path], g(n), h(n)]  
h(n) is optional is this case. Single letter words have it by default.

Node Format: [word, [path], g(n)]

Frontier:  
['m', [[1, 0]], 3, -10962.93046359313], ['n', [[0, 1]], 1, -4789.033130727763]  
['t', [[1, 1]], 1, -9597.31268937656], ['hn', [[0, 0], [0, 1]], 5]  
['hm', [[0, 0], [1, 0]], 7], ['ht', [[0, 0], [1, 1]], 5]

Frontier:

['n', [[0, 1]], 1, -4789.033130727763], ['t', [[1, 1]], 1, -9597.31268937656]  
['hn', [[0, 0], [0, 1]], 5], ['hm', [[0, 0], [1, 0]], 7]  
['ht', [[0, 0], [1, 1]], 5], ['mh', [[1, 0], [0, 0]], 7]  
['mn', [[1, 0], [0, 1]], 4], ['mt', [[1, 0], [1, 1]], 4]

Frontier:

['t', [[1, 1]], 1, -9597.31268937656], ['hn', [[0, 0], [0, 1]], 5]  
['hm', [[0, 0], [1, 0]], 7], ['ht', [[0, 0], [1, 1]], 5]  
['mh', [[1, 0], [0, 0]], 7], ['mn', [[1, 0], [0, 1]], 4]  
['mt', [[1, 0], [1, 1]], 4], ['nh', [[0, 1], [0, 0]], 5]  
['nm', [[0, 1], [1, 0]], 4], ['nt', [[0, 1], [1, 1]], 2]

Frontier:

['hn', [[0, 0], [0, 1]], 5], ['hm', [[0, 0], [1, 0]], 7]  
['ht', [[0, 0], [1, 1]], 5], ['mh', [[1, 0], [0, 0]], 7]  
['mn', [[1, 0], [0, 1]], 4], ['mt', [[1, 0], [1, 1]], 4]  
['nh', [[0, 1], [0, 0]], 5], ['nm', [[0, 1], [1, 0]], 4]  
['nt', [[0, 1], [1, 1]], 2], ['th', [[1, 1], [0, 0]], 5]  
['tn', [[1, 1], [0, 1]], 2], ['tm', [[1, 1], [1, 0]], 4]

Number of expansions: 30  
1 Words found: ['hm']  
Average frontier size: 19.29032258064516  
Max frontier size: 24  
Average depth: 2.3333333333333335  
Average branching factor: 1.6666666666666667  
Total score: 7  
Runtime: 0.0003575828831534622

Search Type: A\* using h1()  
Dumbness turned ON

Node Format: [word, [path], g(n), h(n)]  
In this case, the frontier is sorted based on h(n)

Frontier(1):

['tn', [[1, 1], [0, 1]], 2, -99999999999], ['m', [[1, 0]], 3, -38936.33350997487]  
['h', [[0, 0]], 4, -28770.41085171584], ['n', [[0, 1]], 1, -23532.05443786732]  
['th', [[1, 1], [0, 0]], 5, -12895.192755], ['tm', [[1, 1], [1, 0]], 4, -3549.18]

Frontier(2):

['tn', [[1, 1], [0, 1]], 2, -99999999999],   
['tmh', [[1, 1], [1, 0], [0, 0]], 8, -99999999999]  
['tmn', [[1, 1], [1, 0], [0, 1]], 5, -99999999999]  
['m', [[1, 0]], 3, -38936.33350997487]  
['h', [[0, 0]], 4, -28770.41085171584]  
['n', [[0, 1]], 1, -23532.054437867326]  
['th', [[1, 1], [0, 0]], 5, -12895.192755632572]

Frontier(3):

['tn', [[1, 1], [0, 1]], 2, -99999999999]  
['tmh', [[1, 1], [1, 0], [0, 0]], 8, -99999999999]  
['tmn', [[1, 1], [1, 0], [0, 1]], 5, -99999999999]  
['thn', [[1, 1], [0, 0], [0, 1]], 6, -99999999999]  
['thm', [[1, 1], [0, 0], [1, 0]], 8, -99999999999]  
['m', [[1, 0]], 3, -38936.33350997487]  
['h', [[0, 0]], 4, -28770.41085171584]  
['n', [[0, 1]], 1, -23532.054437867326]

Frontier(4):

['tn', [[1, 1], [0, 1]], 2, -99999999999]  
['tmh', [[1, 1], [1, 0], [0, 0]], 8, -99999999999]  
['tmn', [[1, 1], [1, 0], [0, 1]], 5, -99999999999]  
['thn', [[1, 1], [0, 0], [0, 1]], 6, -99999999999]  
['thm', [[1, 1], [0, 0], [1, 0]], 8, -99999999999]  
['nh', [[0, 1], [0, 0]], 5, -99999999999]  
['nm', [[0, 1], [1, 0]], 4, -99999999999]  
['m', [[1, 0]], 3, -38936.33350997487]  
['h', [[0, 0]], 4, -28770.41085171584]  
['nt', [[0, 1], [1, 1]], 2, -3543.59]

Number of expansions: 30  
2 Words found: ['hm', 'nth']  
Average frontier size: 9.483870967741936  
Max frontier size: 18  
Average depth: 2.7333333333333334  
Average branching factor: 1.2666666666666666  
Total score: 13  
Runtime: 0.0021034889945923396