Nathan Driscoll z5204935 COMP3411 Assignment 1 - Prolog and Search

Part 2 - Search

Question 1: Search Algorithms for the 15-Puzzle

A)

	Start10	Start12	Start20	Start30	Start40
UCS	2565	Mem	Mem	Mem	Mem
IDS	2407	13812	5297410	Time	Time
A*	33	26	915	Mem	Mem
IDA*	29	21	952	17297	112571

B)

Uniform cost search is not time efficient compared to the other 4 algorithms as it only looking for the least cost path. When all costs are the same, it becomes BFS. Compared to the other 4 searches is the most memory inefficient. It runs out of memory from start12 onwards.

Iterative deepening search is more memory efficient than UCS as it does not run out of memory, however it is not time efficient as it expands too many nodes (compared to A* and IDA*). With start30 and start40 it takes over 5 minutes. This is because it must repeat all the searching it has previously done as it iteratively deepens the search. It has better memory efficiency than UCS as it uses DFS (which uses less memory than BFS (which is the algorithm UCS follows).

A* search is very time efficient; it expands many times less nodes than both UCS and IDS. Though it is not as memory efficient as IDA* as it runs of memory at start30 and start40. This is because it is an informed search (compared to the previous 2 uninformed searches) that uses an admissible heuristic h(n) (in formula f(n) = g(n) + h(n)) which helps it find the shortest path.

Iterative deepening A* search is both time and memory efficient. It expands close to the same number of nodes as A* search, but does not run out of memory at start 30 or start 40. IDA* is clearly the best performing and most efficient search out of the 4. It has the benefits of A* search except iteratively deepens the search – reducing its memory usage.

Question 2: Heuristic Path Search for 15-Puzzle

	Start50		Start60		Start64	
IDA*	50	14642512	60	321252368	64	1209086782
1.2	52	191438	62	230861	66	431033
1.4	66	116174	82	3673	94	188917
1.6	100	34647	148	55626	162	235852
Greedy	164	5447	166	1617	184	2174

b)

Line 43 from idastar.pl is:

F1 is G1 + H1,

In heuristic.pl, it is changed to:

F1 is (2 - 1.6) * G1 + 1.6 * H1,

(where 1.6 is w. This number is changed to the required values of w (1.2, 1.4, 1.6))

This reflects the formula in question 5 of the week 3 tutorial:

$f(n) = (2 - w) \cdot g(n) + w \cdot h(n)$ where $0 \leq w \leq 2$

d)

As W increases, the number of nodes (N) expanded decreases which increases speed, but the length of the path (G), increases, meaning we have a longer path and a less quality solution. The closer to 0 that w is, the shorter the path is, but the more expanded nodes. This means the closer to 0 w is, the higher quality of the algorithm in terms of finding the shortest path, but the longer it will take.

Also, as w decreases, the algorithm comes closer in performance to IDA* and as it increases towards 2, it comes closer to the performance of greedy search.