Akira Wang

OVERVIEW OF RESULTS (Tested via a UNIX Server)

Non-Optimised IDA*

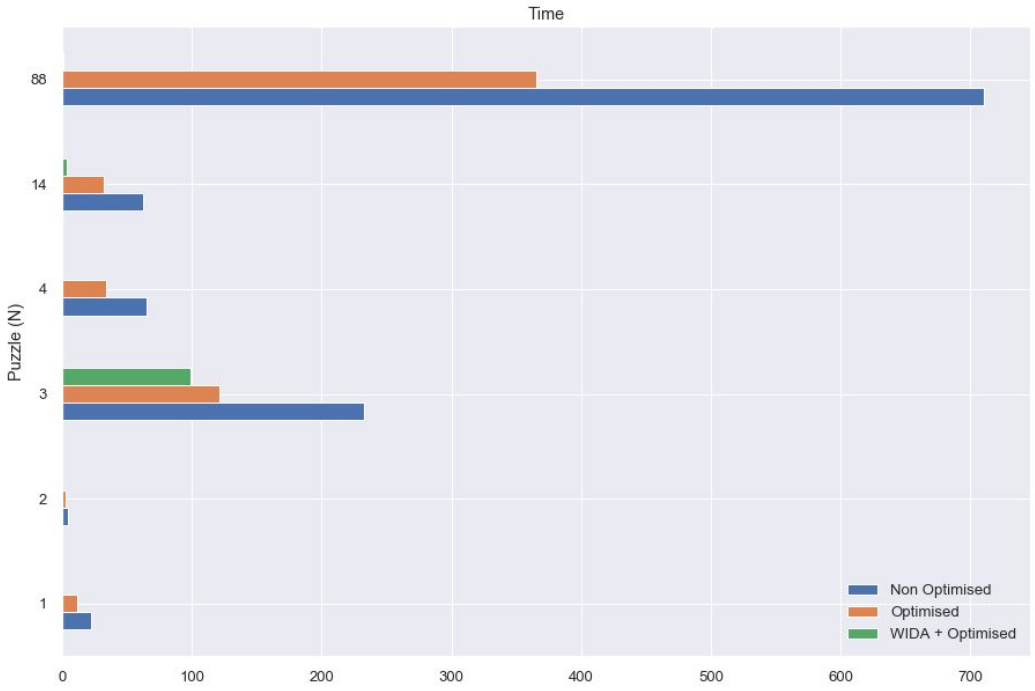
Puzzle (N)	Solution	Generated	Expanded	Time	Expanded/Second
1	57	243248017	120524730	22.416592	5376586
2	55	48083305	23368126	4.454322	5246169
3	59	2543473814	1284136412	233.055573	5510001
4	60	703975730	346669506	65.093102	5325749
14	59	675410413	335302025	62.179546	5392481
88	65	7664208980	3736150178	710.522034	5258317.5

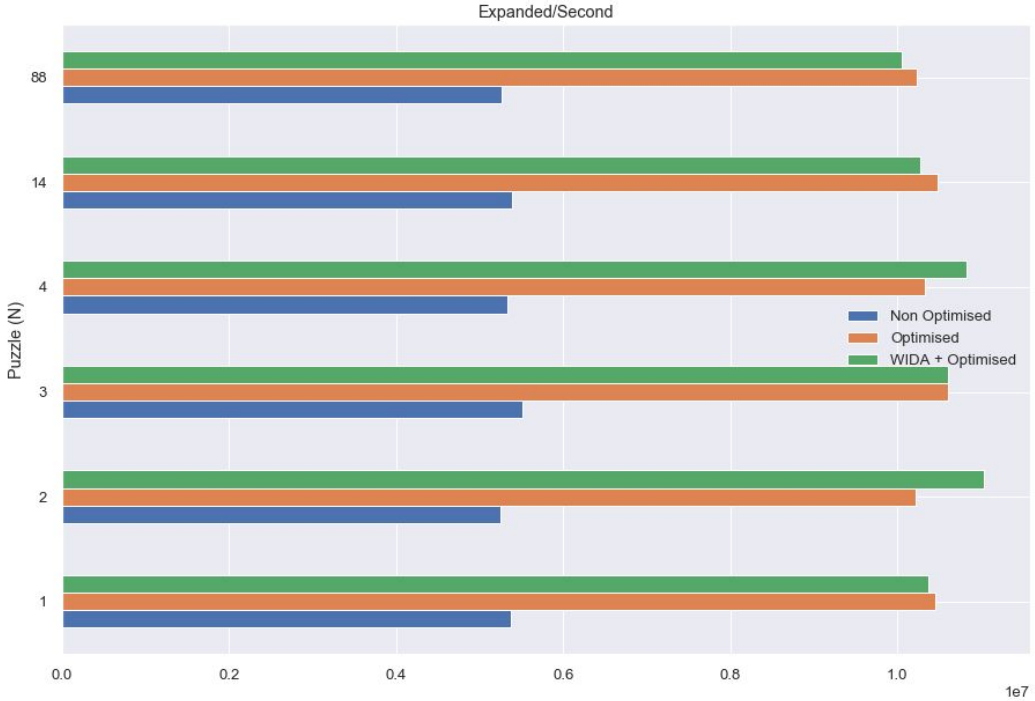
Optimised IDA* using Last Move Heuristics

Puzzle (N)	Solution	Generated	Expanded	Time	Expanded/Second
1	57	243248017	120524730	11.530247	10452918
2	55	48083305	23368126	2.284651	10228313
3	59	2543473814	1284136412	120.95961	10616242
4	60	703975730	346669506	33.533901	10337882
14	59	675410413	335302025	31.995134	10479781
88	65	7664208980	3736150179	365.035492	10235033

Optimised WIDA with Last Move Heuristics

Puzzle (N)	Solution	Generated	Expanded	Time	Expanded/Second
1	61	3113345	1576318	0.151976	10372151
2	63	194625	99344	0.008998	11040676
3	59	2016035885	1046650594	98.632011	10611673
4	64	1097303	562936	0.051992	10827358
14	63	73757587	37406665	3.641447	10272472
88	69	20352678	10209310	1.014845	10059970


Time
88
14
4
3
2
1
Puzzle (N)
0
100
200
300
400
500
600
700
Non Optimised
Optimised
WIDA + Optimised


Expanded/Second
88
14
4
3
2
1
Puzzle (N)
0.0
0.2
0.4
0.6
0.8
1.0
1e7
Non Optimised
Optimised
WIDA + Optimised

CONCLUSIONS (Based on Results)

Generally, we can conclude that IDA* without any optimisations is the slowest at expanding nodes per second whilst IDA* with the Last Move Heuristics will cut the time required to run by half, with almost double the number of expanded nodes per second.

The Last Move Heuristics was implemented by adding a previous move value in the node. Therefore, even if the move was applicable, if it was done previously then the IDA* would skip it and continue to the next possible move - eliminating nearly half the time and doubling the number of expanded nodes.

I had also implemented the Corner Tiles Heuristics but found that increased the time quite a bit and according to the paper - only 22% of cases were valid for the Corner Tiles Heuristics anyway. This was implemented by checking if the top-right, bottom-left, bottom-right were 3, 12 and 15 respectively and adding 2 moves for any incorrectly placed tiles adjacent to it. However, due to the time drawbacks I decided to not implement it into the final optimisation.

Also, implementing the WIDA algorithm in place will yield a significantly faster result but will over-estimate the solution and increase the cost of running. The graphs above demonstrate the time differences between WIDA and IDA*.

Finally, in the actual implementation of the IDA*, some coding changes included:

- An effort to avoid mallocing variables and rather use local variables
- Breaking for loops when values are found rather than finishing it
- Eliminating if statements wherever possible