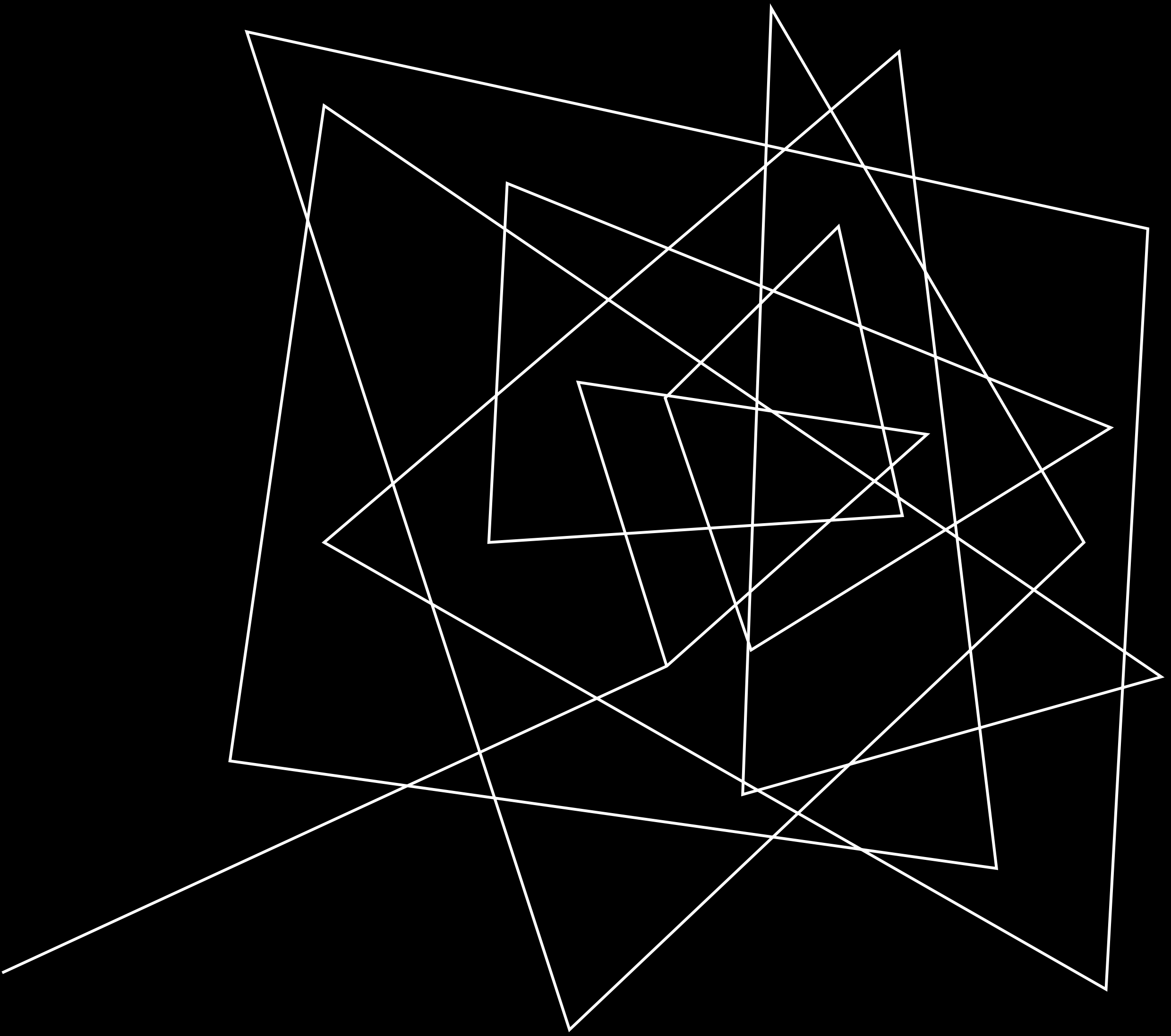
COMP 472 — PROJECT 2

Deus Ex Machina

Sobhan Mehrpour Kevishahi - 40122438

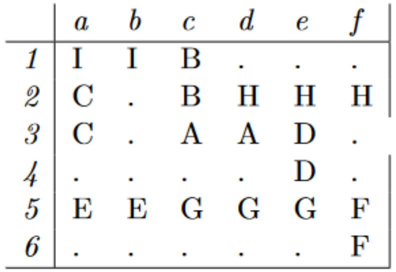


HEURISTICS

HEURISTIC 4

- **Description:** Counts the number of blocks between A and the exit block. Then divides it by 4
- **Pros:**
  - Incentivizes moving to the exit block once there are no cars.
  - Is admissible.
  - Is only 0 at the goal state.
- **Cons:**
  - Does not help remove blocking cars.

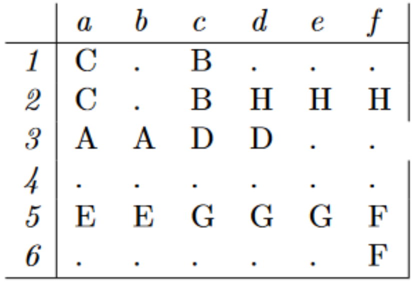
Example 1:



	a	*b*	*c*	*d*	*e*	*f*
1	I	I	B	.	.	.
2	C	.	B	H	H	H
3	C	.	A	A	D	.
4	.	.	.	.	D	.
5	E	E	G	G	G	F
6	.	.	.	.	.	F

h(Example 1)=2/4=0.5

Example 2:

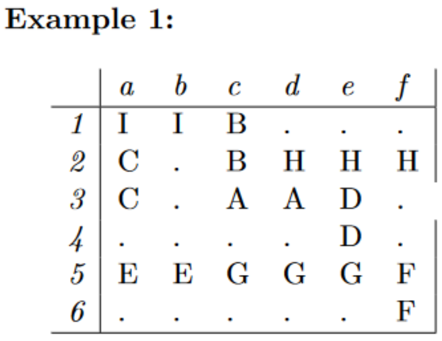


	a	*b*	*c*	*d*	*e*	*f*
1	C	.	B	.	.	.
2	C	.	B	H	H	H
3	A	A	D	D	.	.
4	.	.	.	.	.	.
5	E	E	G	G	G	F
6	.	.	.	.	.	F

h(Example 2)=4/4=1

HEURISTIC 5

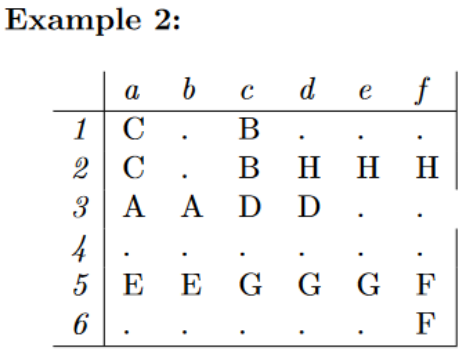
- **Description:** Counts the number of blocking cars and adds it to heuristic 4 (number of positions from A divided by 4).
- **Pros:**
  - Incentivizes moving to the exit block once there are no cars.
  - Incentivizes moving blocked cars out of the way which has more weight than moving to the exit.
  - Is admissible.
  - Is only 0 at the goal state.
- **Cons:**
  - Takes longer to compute.



Example 1:

	a	*b*	*c*	*d*	*e*	*f*
1	I	I	B	.	.	.
2	C	.	B	H	H	H
3	C	.	A	A	D	.
4	.	.	.	.	D	.
5	E	E	G	G	G	F
6	.	.	.	.	.	F

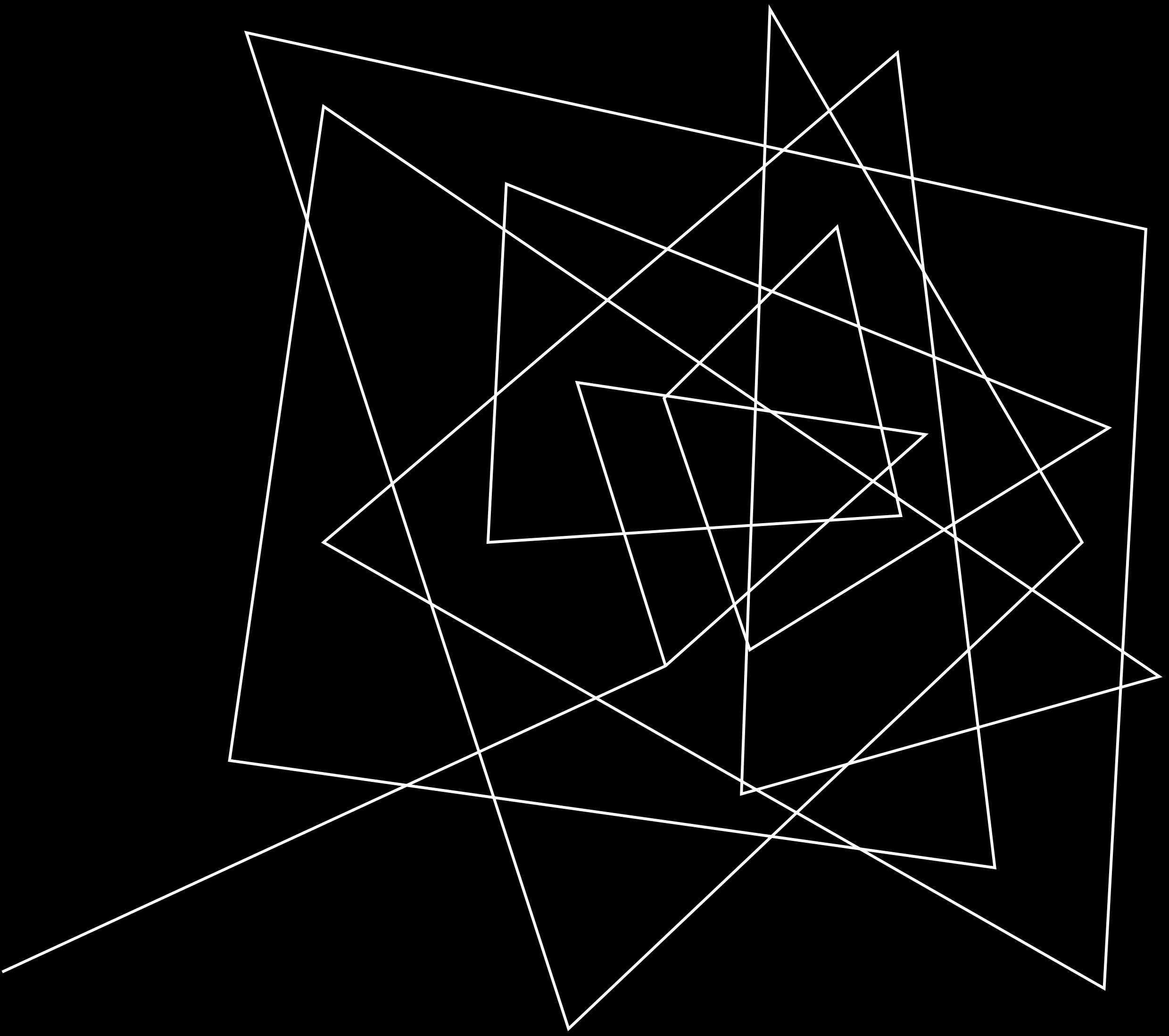
h(Example 1)=1 + 2/4=1.5



Example 2:

	a	*b*	*c*	*d*	*e*	*f*
1	C	.	B	.	.	.
2	C	.	B	H	H	H
3	A	A	D	D	.	.
4	.	.	.	.	.	.
5	E	E	G	G	G	F
6	.	.	.	.	.	F

h(Example 2)=1 + 4/4=2

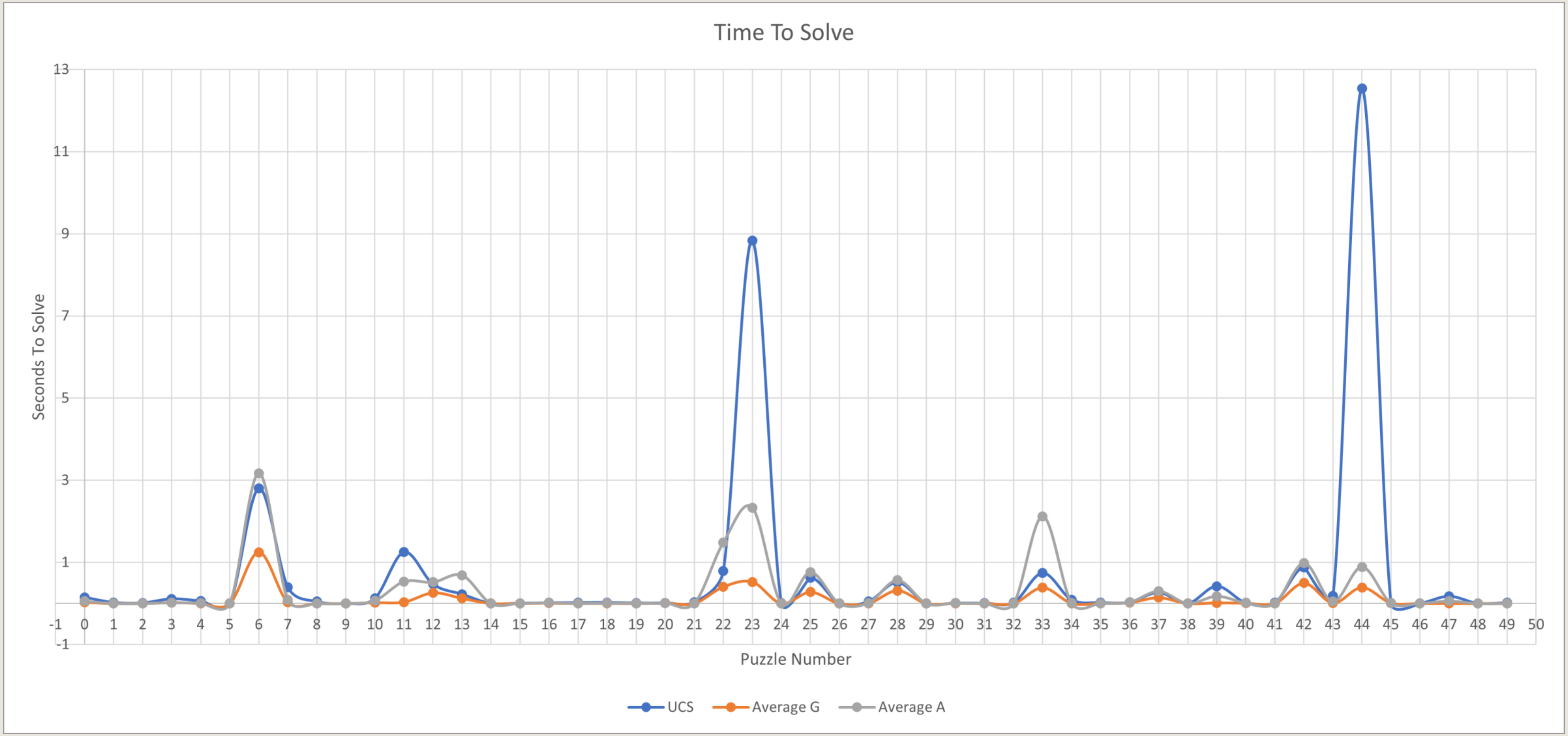


ANALYSIS

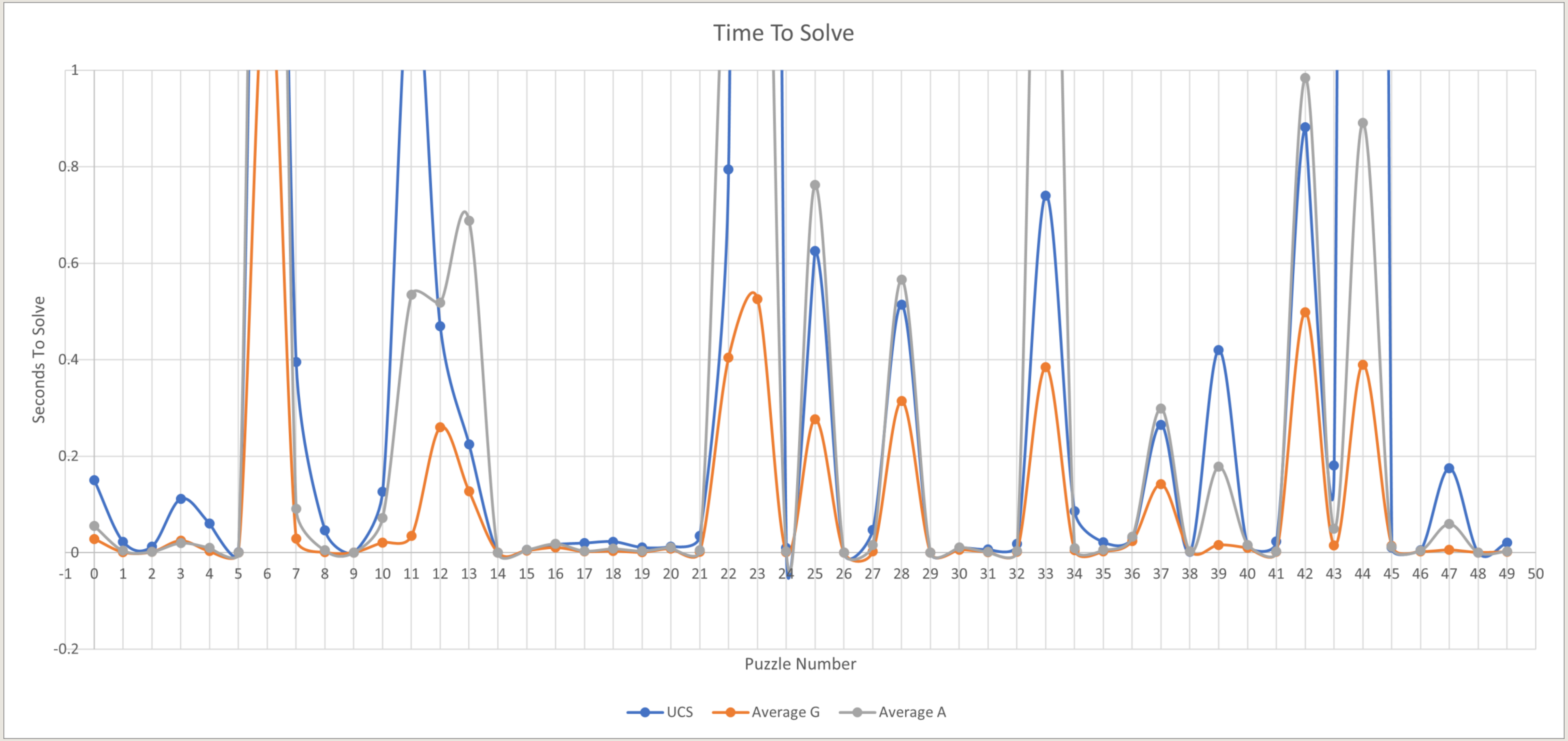
SOLUTION TIME ANALYSIS

- **Uniform Cost Search:** Tends to be one of **the most time-consuming** algorithms because it methodically searches for the solution. **It often has large spikes**.
- **Algorithm A/A*:** Also tends to be very time-consuming, because the admissible heuristics try to find the optimal solution.
- **Greedy Best First Search:** Tends to be **the least time-consuming** algorithm, because it doesn’t care about the optimal solution.

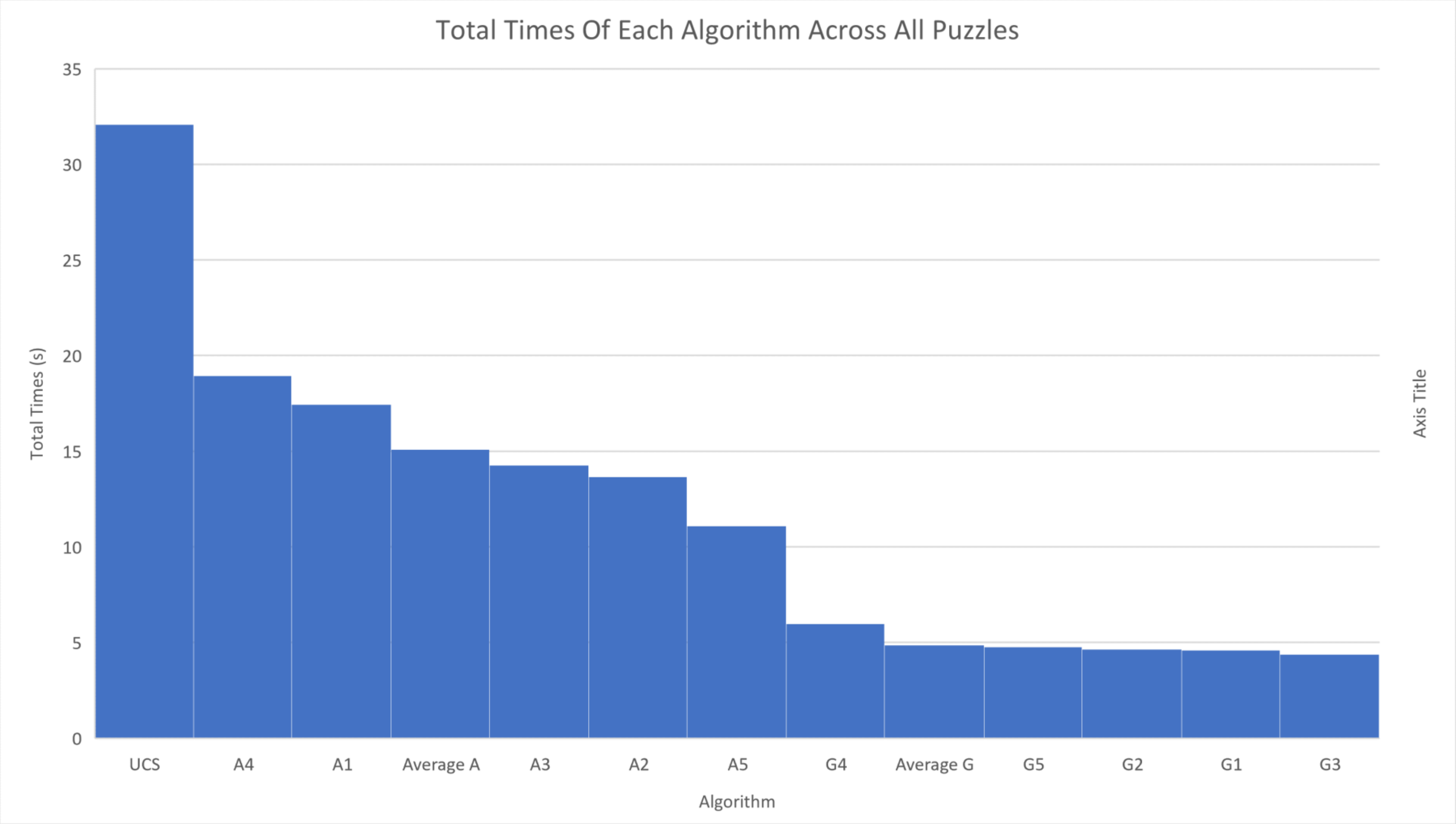
AVERAGE SOLUTION TIMES - 1



AVERAGE SOLUTION TIMES - 2



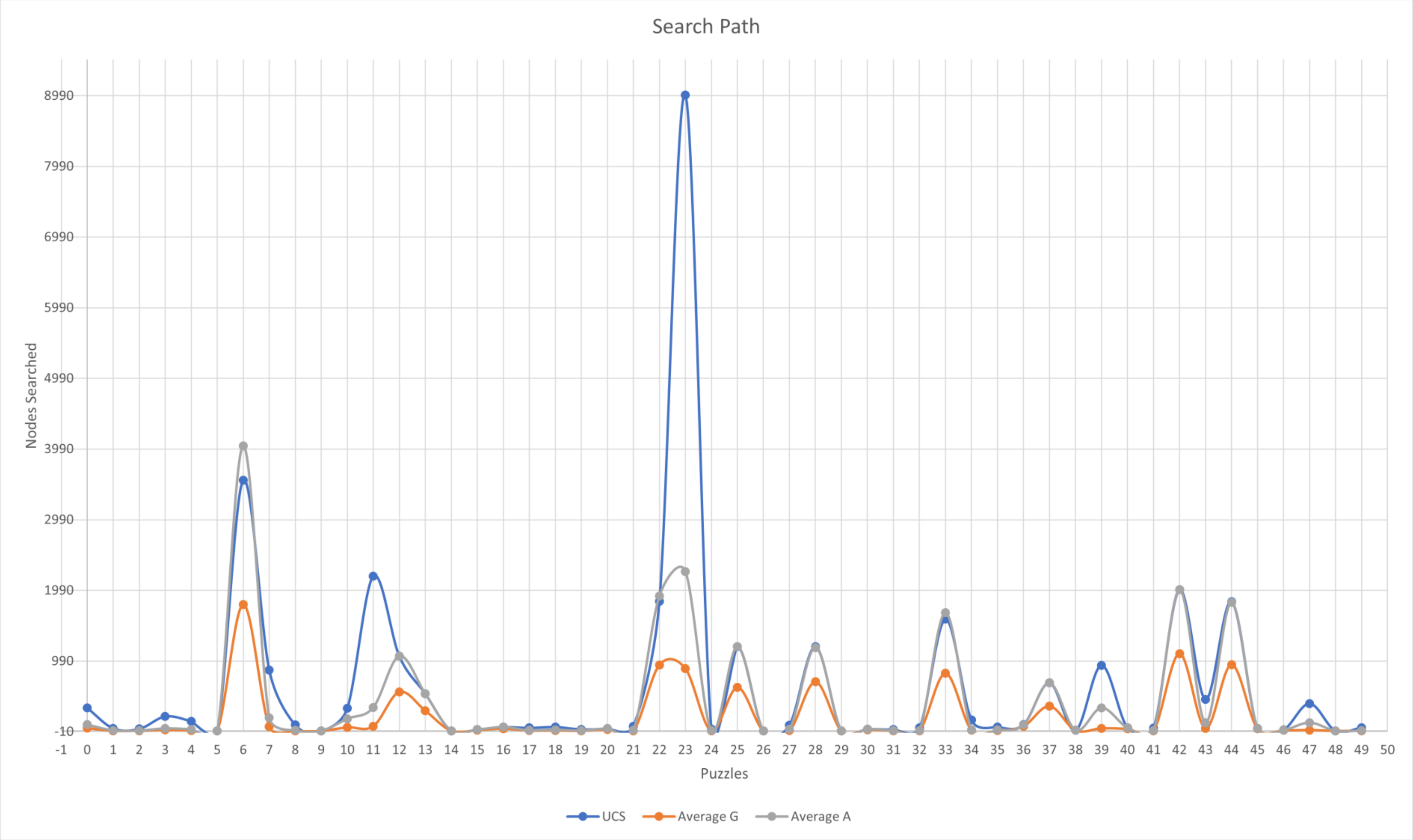
TOTAL SOLUTION TIMES



SEARCH PATH ANALYSIS

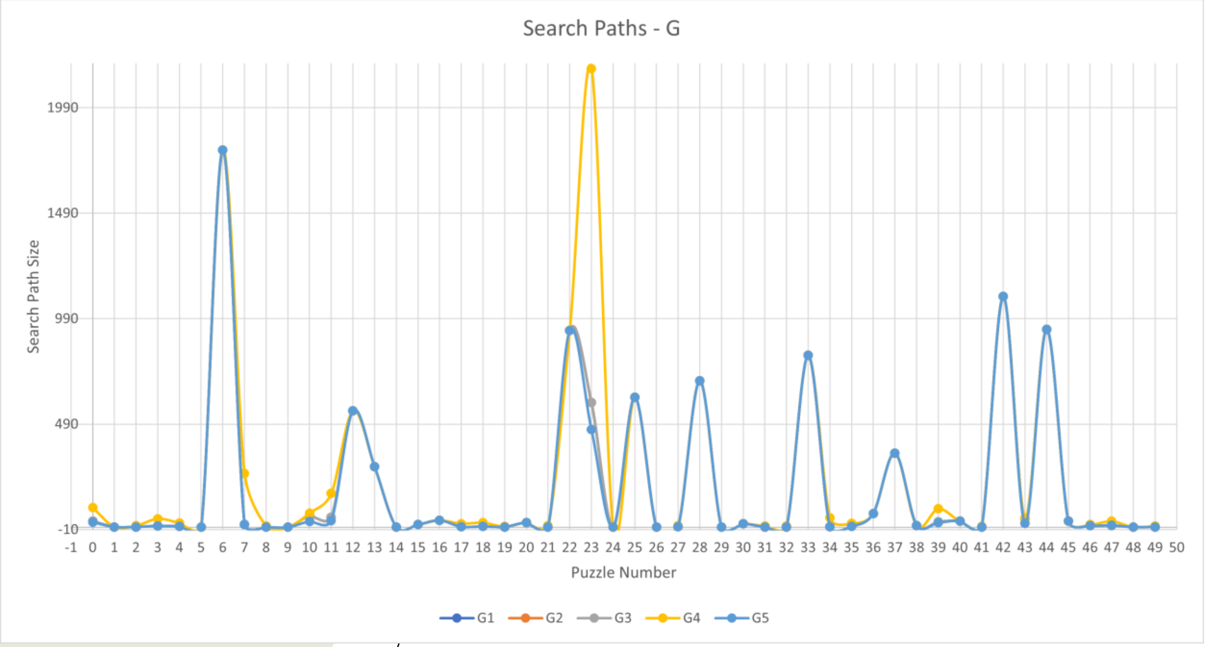
- **Uniform Cost Search:** Tends to **search the most nodes** when finding a solution, as this is a very meticulous algorithm.
- **Greedy Best First Search:** Tends to **search the least nodes**, as this doesn’t strive to find the optimal solution.
- **Algorithm A/A*:** Tends to be somewhere in the middle.

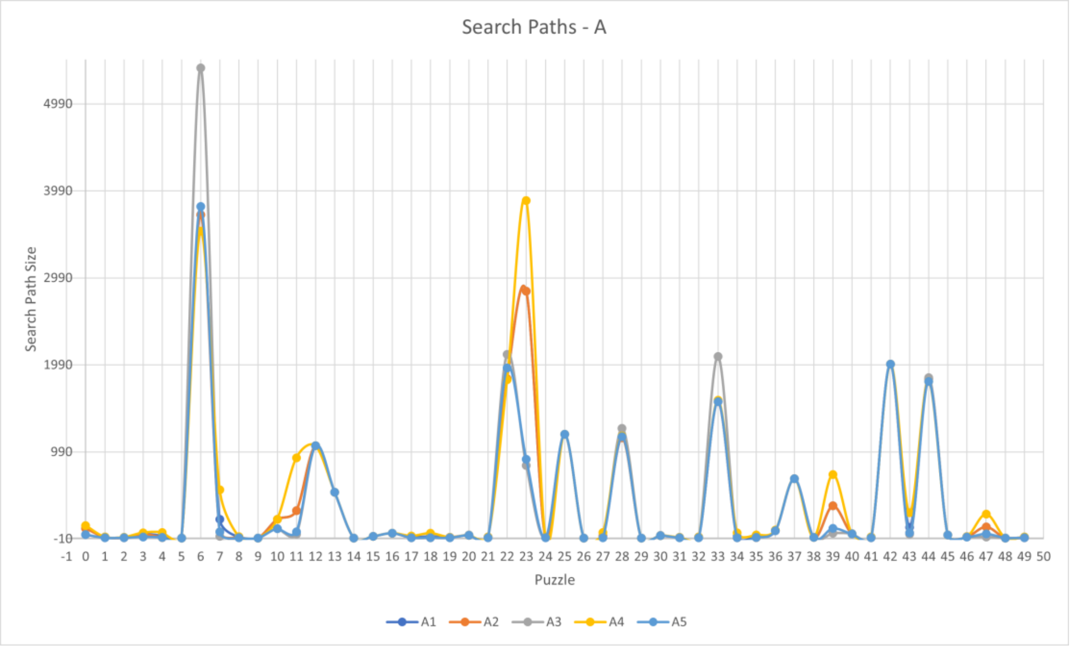
SEARCH PATHS



HEURISTIC ANALYSIS

- **Heuristic 4:** Seems to search the most nodes. This lines up with the results seen with solution times as well.


Search Paths - G
Search Path Size
Puzzle Number
G1
G2
G3
G4
G5


Search Paths - A
Search Path Size
Puzzle
A1
A2
A3
A4
A5