**Name: Ruixue Zhang Student ID: 20619404**

**Informed Search**

In all problems, we assume that the cost for any movement is positive and that the branch factor is finite.

**Problem 1**

1. Manhattan distance heuristic (h2) is expected to perform better than Misplaced tile heuristic (h1). Since for every node n, h2 is closer to actual cost to a nearest goal h\*, i,e, h\*(n) ≥h2(n) ≥h1(n), h2 provides more information to reach the goal. Therefore, it needs fewer nodes to expand when using Manhattan distance heuristic.
2. Both the two heuristics are consistent.

Misplaced tile heuristic: for each movement, the cost(n, n1) will be 1 and the heuristic estimate h will remain same or minus 1. Therefore, we have h(n) = h0, h(n1) = h0 or h(n1) = h0 – 1. Thus, h(n) ≤ cost(n, n1) + h(n1). Also, for one movement (from n1 to n2), we have h(n1) ≤ cost(n1, n2) + h(n2). Therefore, adding two inequalities, we get h(n) ≤ cost(n, n2) + h(n2). By induction, we can prove the consistency of misplaced tile heuristic.

Manhattan distance heuristic: for each movement, the cost(n, n1) will be 1 and the heuristic estimate h will add 1 or minus 1. Therefore, we have h(n) = h0, h(n1) = h0 – 1 or h(n1) = h0 +1. Thus, h(n) ≤ cost(n, n1) + h(n1). For other nodes, the proof is same. Also, for one movement (from n1 to n2), we have h(n1) ≤ cost(n1, n2) + h(n2). Therefore, adding two inequalities, we get h(n) ≤ cost(n, n2) + h(n2). By induction, we can prove the consistency of manhattan distance heuristic.

**Problem 2**

1. Exponential worst case time O(bm) where m is the maximum depth of the tree. Space complexity is O(bm). If heuristic function h is admissible, change m in time complexity and space complexity with d (the depth of optimal path to the goal).
2. IDA\* is complete. IDA\* is based on IDS algorithm with f cost used as threshold instead of depth. For the worst condition, we need to transverse all the nodes but in this case, we still can find a solution.
3. IDA\* is optimal when heuristic function h is admissible.

A instance fail to find the optimal solution when heuristic is inadmissible.

As the graph shown below (inadmissible), the initial threshold is 2 according to IDA\*. Next step is to increase the threshold as 3 (the minimum f cost of children of S). Thus, we choose G and return it.

h = 2

3

S

G

h = 0

S

1

1

A

G

f = 3

f = 6

h = 5

A

**Constraint Satisfaction**

Variables: each cell on the grid

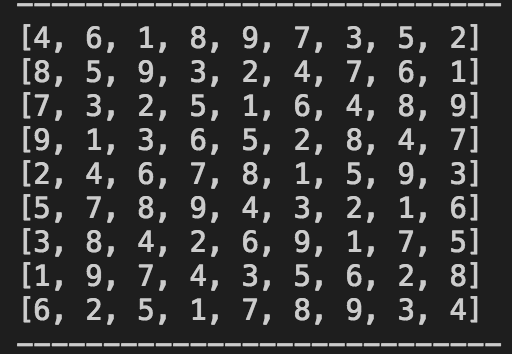
Domains: all the digits from 1 to 9

Constraints:

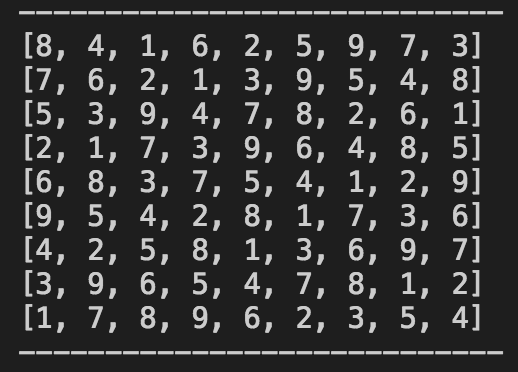
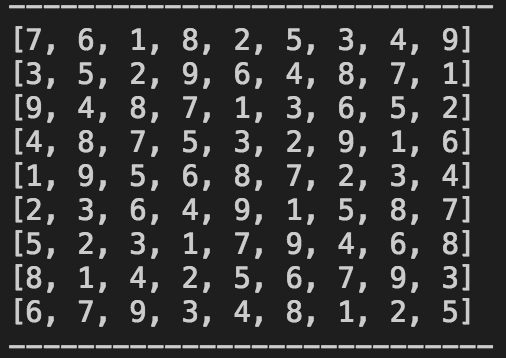
1. The variables filled with initial values equal to the given values.
2. The same single integer may not appear twice in the same row.
3. The same single integer may not appear twice in the same column.
4. The same single integer may not appear twice any of the nine 3×3 sub-regions of the 9x9 playing board.
5. Every row contains all the digits from 1 to 9.
6. Every column contains all the digits from 1 to 9.
7. Every 3×3 sub-region contains all the digits from 1 to 9.

Solution for each test puzzle:

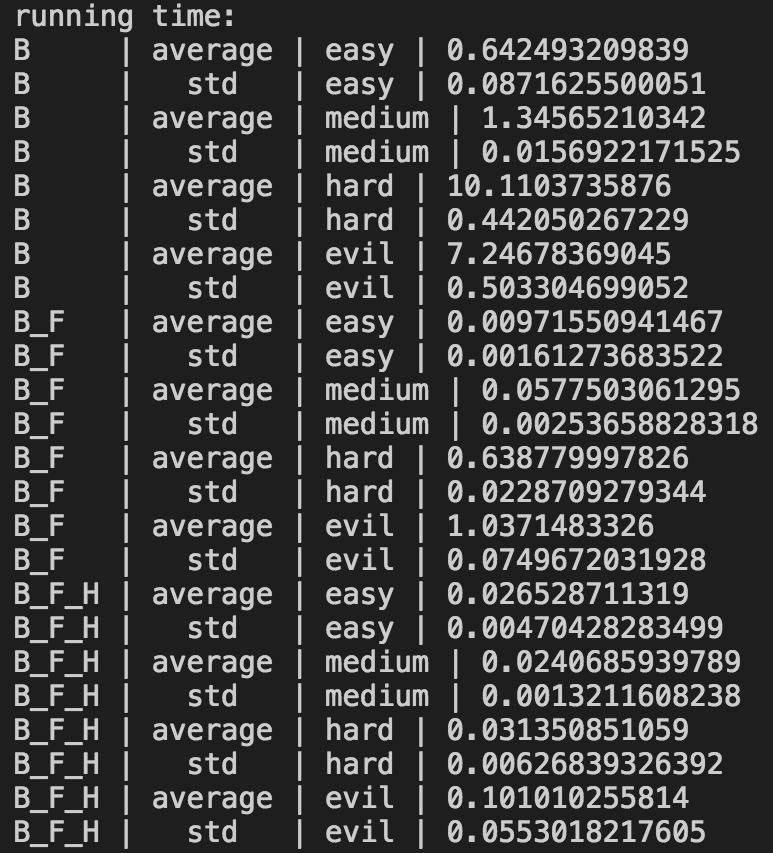
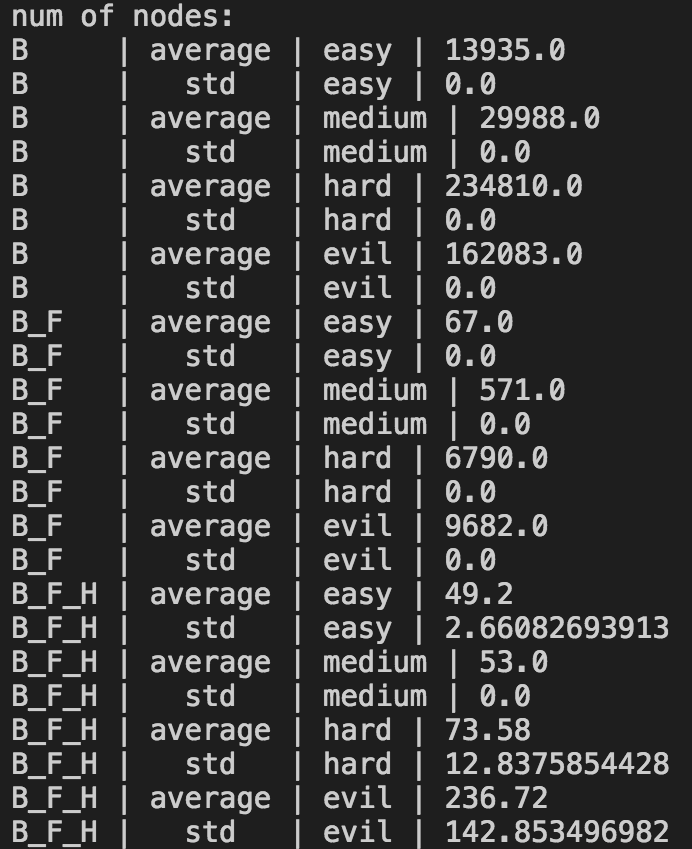
Easy Medium

Hard Evil

Result:

**Time**

|  |  |  |  |
| --- | --- | --- | --- |
|  | B | B+FC | B+FC+H |
| Easy | 0.643 ± 0.087 | 0.010 ± 0.002 | 0.027 ± 0.005 |
| medium | 1.346 ± 0.016 | 0.058 ± 0.003 | 0.024 ± 0.001 |
| hard | 10.110 ± 0.442 | 0.639 ± 0.023 | 0.031 ± 0.006 |
| evil | 7.247 ± 0.503 | 1.037 ± 0.075 | 0.101 ± 0.055 |

**# of Nodes**

|  |  |  |  |
| --- | --- | --- | --- |
|  | B | B+FC | B+FC+H |
| Easy | 13935.0 ± 0.0 | 67.0 ± 0.0 | 49.2 ± 2.7 |
| medium | 29988.0 ± 0.0 | 571.0 ± 0.0 | 53.0 ± 0.0 |
| hard | 234810.0 ± 0.0 | 6790.0 ± 0.0 | 73.6± 12.8 |
| evil | 162083 ± 0.0 | 9682.0 ± 0.0 | 236.7 ± 142.9 |

Brief discussion for the performance:

In two tables, B, B+FC and B+FC+H represent basic backtracking algorithm, backtracking with forward checking and backtracking with forward checking and three heuristics respectively. Seen from two tables above, from B to B+FC, then to B+FC+H, the running time for all puzzle tests is reducing and the number of nodes expanded in the whole procedure also goes down.

Forward checking improves backtracking performance with deleting the potential error before next assignment. Fewer nodes are expected to expand and running time consequently will be shorter. Also, three heuristics speed up the algorithm since they provide more information for more efficient forward checking. This help to avoid expand some bad nodes.

In second table, the standard deviation is 0 in all conditions except the last column. This is because I set a specific order when algorithm choosing next node to expand when three heuristics shut down. For each run, same procedure will be exactly repeated. Whereas, I set random choice for B+FC+H if there is a tie after first two heuristics applied, most constrained variables and most constraining variables. Choosing different variables results in different condition.