Jiachang (Ernest) Xu

CSCI 360: Introduction to Artificial Intelligence

Project 1: Theoretical Part

25 Sept. 2017

Performing the Experiments

```
Ernest-MacBook:8-Puzzle-Solver xujiachang1024$ make
g++ -std=c++11 -o 8PuzzleSolver main.cpp Puzzle8Solver.cpp
CPUTimer.cpp
Ernest-MacBook:8-Puzzle-Solver xujiachang1024$ ./8PuzzleSolver
807
     w       Cost     Time(ms)   Expansions
  0.00      22.36       667.69     97345.88
  0.25      22.36       313.78     44294.86
  0.50      22.36        95.94     14366.42
  0.75      22.36        27.90      4165.04
  1.00      22.48         8.87      1232.60
  1.50      22.92         4.09       588.20
  2.00      24.20         3.07       467.04
  3.00      27.76         2.79       419.28
  5.00      32.92         1.92       300.12
 10.00      39.28         2.12       311.94
```

Interpreting the Results

1. As the weight *w* increases, the solution cost slightly increases, the running time drastically decreases, and the number of expansions drastically decreases as well.
2. When we choose the weight $w = 0$, it becomes Dijkstra's algorithm, $f(x) = g(x)$. Dijkstra's algorithm is a uniform cost search algorithm. Dijkstra's algorithm guarantees to find an optimal solution, a.k.a. the shortest path. When we choose any weight $w > 0$, it includes a heuristic cost function, $f(x) = g(x) + w * h(x)$, which guarantees completeness, but not an optimal solution.

3. As the weight w increases, I observe a relatively small increase in solution cost. Even when the weight w jumps from 1.00 to 10.00, the solution cost increases no more than 2 times of the optimal solution (w = 0.00). When the weight w = 10, the evaluation of $w * h(x)$ outweighs the evaluation $g(x)$ by 10 times. If we increase the weight w even more, the weighted A* cost function approximate the pure heuristic function: $f(x) = g(x) + w * h(x) \approx w * h(x)$. We learn from the lecture that admissibility of a good pure heuristic search makes the result larger the optimal solution, but smaller than 2 times of the optimal solution.
4. As the weight w increases, I observe a drastic decrease in the number of expansions. Given a heuristic function $h(x)$, A* is optimally effective. It expands the minimal number of nodes needed to find an optimal solution. To guarantee that solution cost c is optimal, all nodes of $f(x) = g(x) + w * h(x) < c$ must be expanded. Otherwise, there might be a path smaller than c. In other words, when the weight w increases, the weighted A* algorithm biases more towards states that are closer to the goal state. Therefore, it first expands the states that are more likely to reach the goal state.
5. As the weight w increases, I observe a drastic decrease in the runtime, because the runtime has a linear relationship with the number of expansions. As Question 4 observes and proves the drastic decrease in the number of expansions, hence the decrease in runtime.
6. The modified heuristic function sometimes overestimates the cost. For example, if we have a puzzle 1023456789, we only need to swap 0 and 1 to reach the goal state (i.e. optimal cost = 1), but the modified heuristic function gives use a Manhattan distance of 2 (larger than optimal cost). Therefore, the modified heuristic function is **NOT admissible**,

which is a disadvantage of the modified heuristic function, because A* algorithm requires an admissible heuristic function to be optimally effective. However, the modified A* algorithm biases more towards the heuristic function, which means it first expands states with higher possibility to reach goal state, and potentially saves time.