CSci 384: Artificial Intelligence Derek Trom

Instructor: Dr. M. E. Kim February 13, 2020

**Due: 11:59 PM, February 11*th* (Tue.) 2020**

**Home Assignment 1 (194/200 points.)**

The sliding-title puzzle consists of five black titles, five white titles, and an empty space in the

configuration shown in the Figure.



The puzzle has two legal moves (i.e. actions) with associated costs:

1. A title may move into an adjacent empty location. – This has a cost of 1.
2. A title can hop over one, two or three other tiles into the empty position.

– This has a cost equal to the number of tiles jumped over + 1: i.e. a cost of 2, 3 or 4.

The goal is to have all the white tiles to the right of all the black tiles.

The position of blank is not important.

1. [30/30] **Problem Formulation**

Clearly **formulate** the problem in terms of 6 factors below. – not a verbal description.

* 1. [5/5] States: how do you define and represent a state?

e.g.) (*a, b, c, d, e, f, g, h, i, j, k*)

where *a* is the tile located in the 1st location, b is the tile in the 2nd location, etc.

(tile@loc1, tile@loc2, tile@loc3, tile@loc4, tile@loc5, tile@loc6, tile@loc7, tile@loc8, tile@loc9)

* 1. [5/5] Initial State

(W, W, W, W, E, B, B, B, B)

* 1. [5/5] Goal State

(E, B, B, B, B, W, W, W, W) or

(B, E, B, B, B, W, W, W, W) or

(B, B, E, B, B, W, W, W, W) or

(B, B, B, E, B, W, W, W, W) or

(B, B, B, B, E, W, W, W, W) or

(B, B, B, B, W, E, W, W, W) or

(B, B, B, B, W, W, E, W, W) or

(B, B, B, B, W, W, W, E, W) or

(B, B, B, B, W, W, W, W, E)

* 1. [10/10] The possible Actions: **Formulate** each action with the current state and its successor states.

Blank space can move 1,2,3 to left or right.

* 1. [5/5] A step cost and a path cost

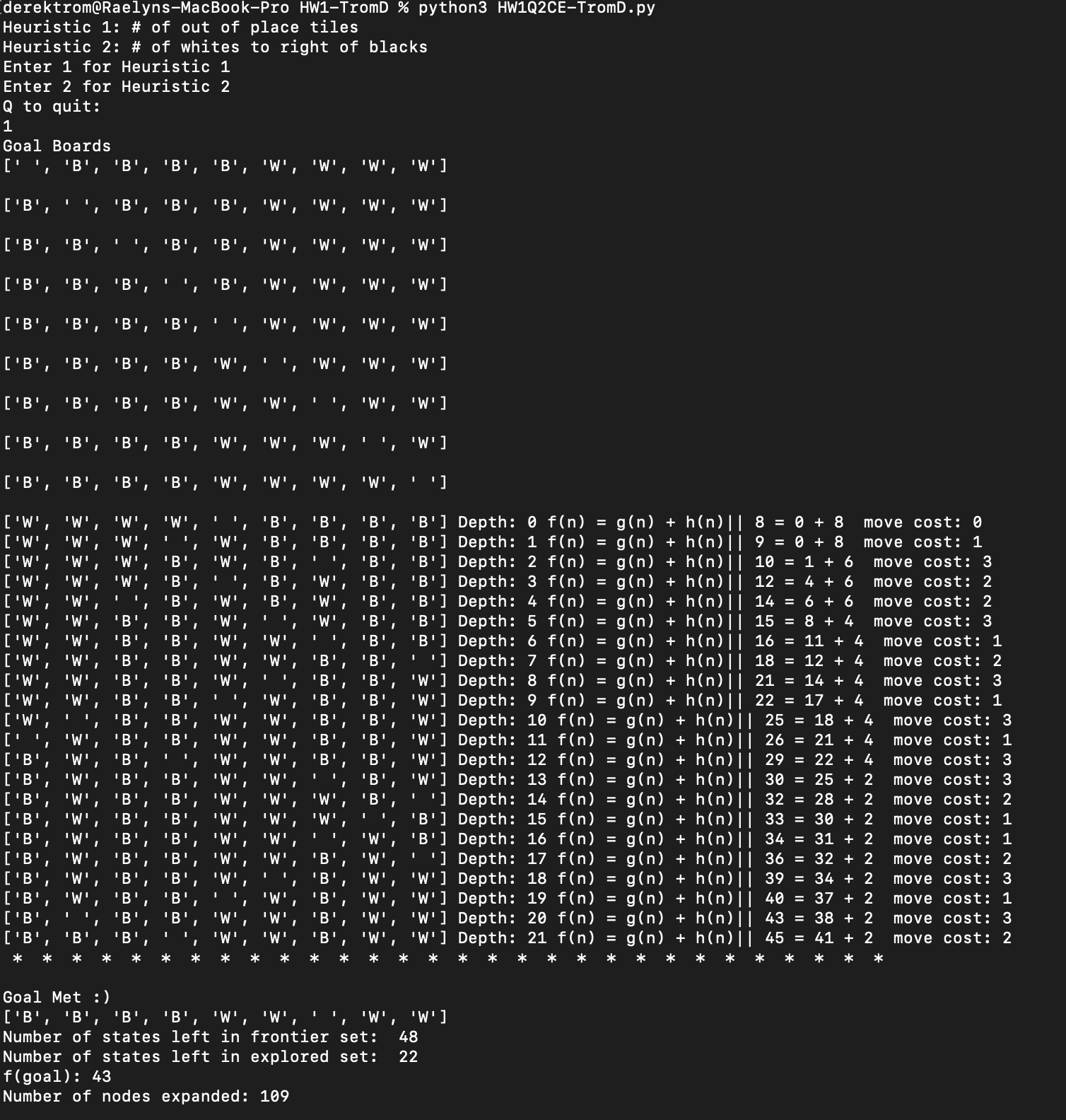
1 space move = 1 cost

2,3,4 move = 1 + tiles jumped over

1. [154/170] **Solve** the problem to find an optimal solution using an **A\* algorithm** with **GRAPH- SEARCH**.
   1. [10/10] Define your admissible and consistent heuristic function, *h1*.

H1 = # of tiles not in correct spot

* 1. [*Optional, 0/*10] Prove that your heuristic in (a) is both admissible and consistent.
  2. [80/90] Implement the problem to solve it. Use any programming language of your preference.



* 1. [24/ 25] Your output has to print the following result:

(d1) [10/10] the ***optimal solution***? Represent your optimal solution as a sequence of states.

See above picture HW1Q2C-TromD.png

(d2) [4/ 5] the ***optimal cost*** of solution (i.e. *f* (*Goal*))?

See above picture HW1Q2C-TromD.png

Optimal cost of f(goal) = 37.

(d3) [10/10] the number of the states in the ***explored*** set and the number of the states remaining in the ***frontier*** list when a goal is reached, respectively.

See above picture HW1Q2C-TromD.png

* 1. [30/35] Solve the same problem with the 2nd heuristic function *h2* by A\* with GRAPH- SEARCH (and print the following result).

(e1) [8/10] Define your 2nd heuristic function, *h2*, which is admissible and consistent.

# of whites to left of rightmost black

h2 = the sum of the distances of the tiles from their goal positions.

(e2) [9/10] What is your ***optimal solution***? Give it as a sequence of states.

See below picture HW1Q2E-TromD.png

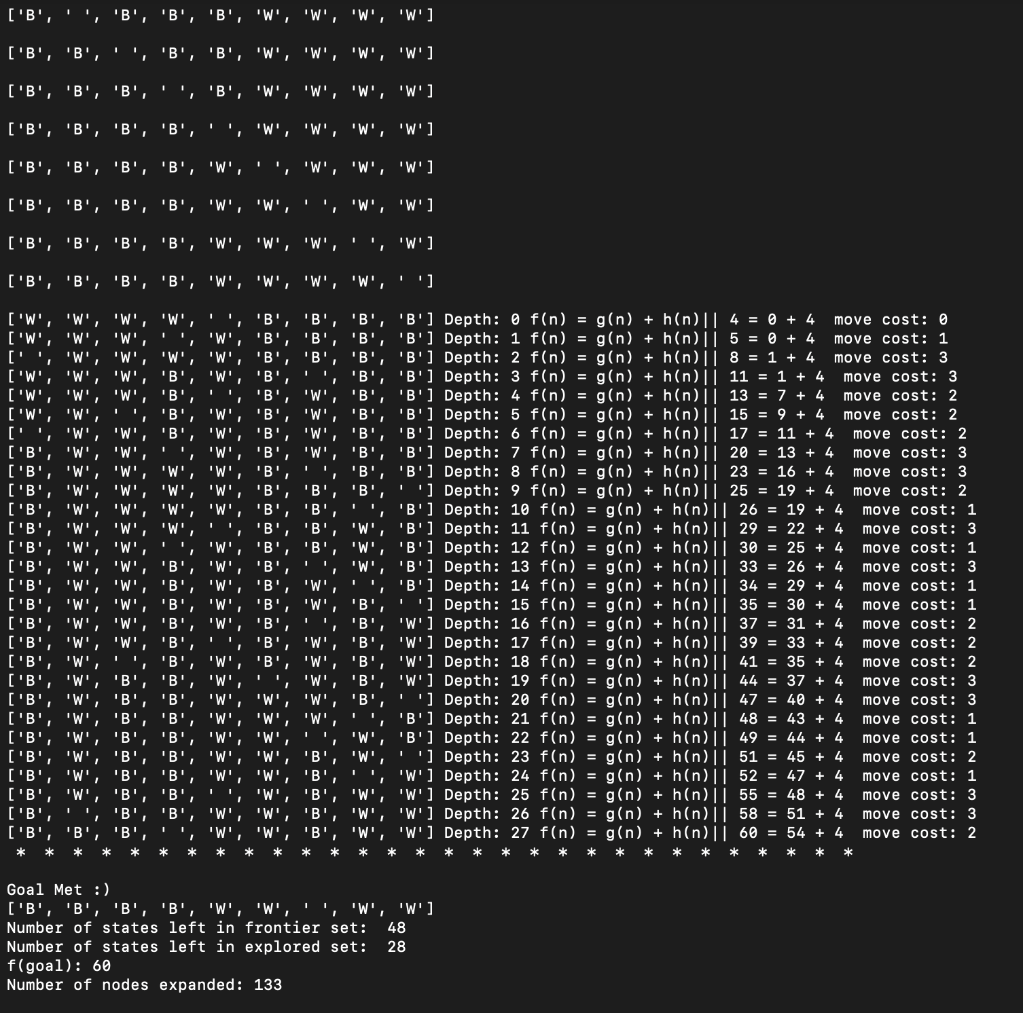
(e3) [3/5] What is the ***optimal cost***of the solution?

* Optimal cost is 37, which is must be equal for both h1 and h2.

HW1Q2E-TromD.png

(e4) [10/10] Give the total number of expanded nodes, i.e. those in the explored list, and those in the frontier list, respectively.

HW1Q2E-TromD.png



* 1. [10/10] Compare the performance of both heuristics *h1* and *h2,* in terms of the total number of expanded nodes in the explored set.
     + - Both were very close in the nodes expanded area, h1 at 133 and h2 at 109 but h1 solved it quicker and took up less memory space.

1. [Optional, 10/30] Solve the same problem by Iterative Deepening Search (IDS). Assume that the maximum depth, *m*=50. Implement IDS to solve it.
   1. [0/5] What is your ***optimal solution?*** Give it as a sequence of states?
   2. [0/5] What is the ***optimal cost***of the solution?
   3. [0/5] What is the ***depth of optimal goal*** in the tree?
   4. [0/5] Give the total number of expanded nodes in the ***explored*** list.
   5. [10/10] Its Implementation= see below copied from terminal

I started to solve this but kept getting stuck in a loop of the same moves

See HW1Q3-TromD.py