Offline 02: Hill Climbing and Simulated Annealing

Solve the 8-puzzle problem using **steepest ascent hill climbing algorithm** and **simulated annealing algorithm**.

- The successor of each state is found by moving the blank space to *Left*, *Right*, *Up* or *Down*.
- Implement the **#misplace-tiles** and **manhattan-distance** as the heuristic cost h(n) of a state n.
- Terminate the algorithm when the goal state is reached or after 300 iterations.
- ~~The memory requirement of your implementation must be O(1).~~
- The schedule function of simulated annealing must be such that as t increases, T decreases and eventually becomes 0.
- A template code is available here
- Run hill climbing for sample inputs using #misplace-tiles and manhattan-distance and **log the results in this report.** Similarly, run hill climbing for sample input using #misplace-tiles and manhattan-distance and **log the results in the same report.**

0	1	2
3	4	5
6	7	8

Goal state (Assume 0 means blank)

Sample Input:

Sample Input	Sample output
3 1 2 6 4 5 0 7 8	Neighbor [[3, 1, 2], [0, 4, 5], [6, 7, 8]] h=2 Neighbor [[0, 1, 2], [3, 4, 5], [6, 7, 8]] h=0 Neighbor [[1, 0, 2], [3, 4, 5], [6, 7, 8]] h=2 solution [[0, 1, 2], [3, 4, 5], [6, 7, 8]] h=0 [Note that you need to follow the sample output format but the actual values may be different depending on the heuristic and the algorithm]
3 1 2 6 4 0 7 8 5	

Instructions:

- Read the questions very carefully and answer all parts of the question.
- Your output should match the sample output format. Your code will be tested on other inputs not given in the sample input.
- You will get -100% for adopting any unfair means.
- **Your marks will fully depend on your viva and understanding.**
    - **Total 20 marks**
        - Heuristics = 2+4 marks
        - Hill climbing = 6 marks (implementation) + 2 marks (report)
        - Simulated annealing = 4 marks (implementation) + 2 marks (report)
- **Submit the .ipynb file**

Pseudocodes

Steepest ascent hill climbing

function HILL-CLIMBING($problem$) **returns** a state that is a local maximum

$current \leftarrow$ MAKE-NODE($problem$.INITIAL-STATE)
loop do
  $neighbor \leftarrow$ a highest-valued successor of $current$
  if neighbor.VALUE $\leq$ current.VALUE **then return** $current$.STATE
  $current \leftarrow neighbor$

- Memory requirement O(1)
- If you use heuristic cost instead of heuristic value, then you should pick the lowest-cost successor as neighbor and stop when neighbor cost is higher than current cost.

Simulated annealing

function SIMULATED-ANNEALING($problem, schedule$) **returns** a solution state
  inputs: $problem$, a problem
      $schedule$, a mapping from time to "temperature"

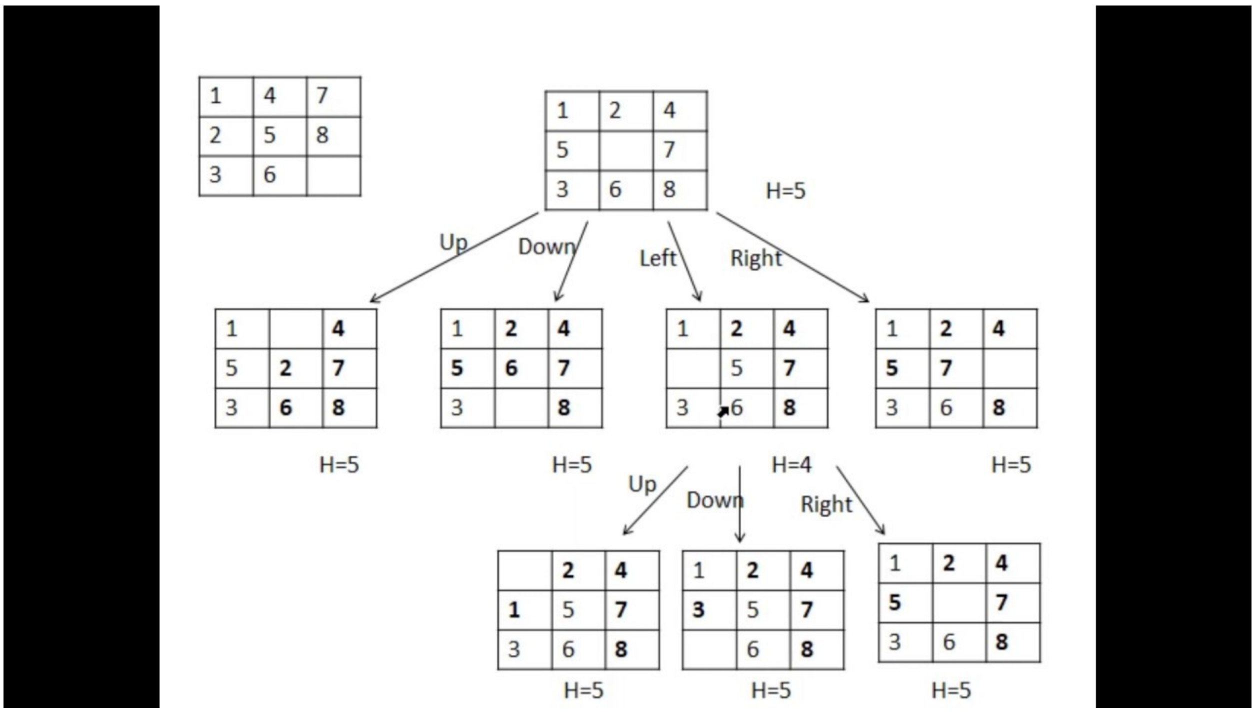
$current \leftarrow$ MAKE-NODE($problem$.INITIAL-STATE)
for $t = 1$ **to** ∞ **do**
  $T \leftarrow schedule(t)$
  if $T = 0$ **then return** $current$
  $next \leftarrow$ a randomly selected successor of $current$
  $\Delta E \leftarrow next$.VALUE – $current$.VALUE
  if $\Delta E > 0$ **then** $current \leftarrow next$
  else $current \leftarrow next$ only with probability $e^{\Delta E/T}$

- Memory requirement O(1)
- If you use heuristic cost instead of heuristic value, then ΔE should be current.cost - next.cost.

Class Lecture:

Heuristic cost and value
Good move and bad move

8-puzzle



Sorting an array

Sort a given array in descending order using **steepest ascent hill climbing** and using **simulated annealing**.

- The value of a state = $\sum_{i} number\ of\ elements\ smaller\ than\ A[i]\ in\ index\ j > i$
  e.g. value of the state [2 5 -1 4]: 1+2+0+0=3

- The successor of a state is found by swapping a pair of numbers for all possible pairs
  - Successor of state `[2 5 -1 4]`
    - `[5 2 -1 4] value: 3+1+0+0=4`
    - `[-1 5 2 4]`
    - `[4 5 -1 2]`
    - `[2 -1 5 4]`

- [2 4 -1 5]
- [2 5 4 -1]

Hill climbing

- No bad moves allowed
- No side away moves allowed

Simulated Annealing

- Sometimes allows bad moves
- Sometimes allows side away moves
- Simulated annealing maintains a temperature
  - Initially the temperature is high
  - Slowly the temperature decreases
  - When the temperature is high
    - The probability of bad moves is high
  - When the temperature is low
    - The probability of bad move decreases
  - e^(-1/500) = 0.998
  - e^(-1/250) = 0.996
  - So, with the decrease of temperature, $e^{\Delta E/T}$ is decreasing

Scheduling function

- A Comparison of Cooling Schedules for Simulated Annealing (Artificial Intelligence)
- Investigation of a Simulated Annealing Cooling Schedule used to Optimize the Estimation of the Fiber Diameter Distribution in a

1. **Exponential Decay** $T_k = \alpha T_{k-1} \quad 0 < \alpha < 1$

   Suppose **T_0 = 100** and $\alpha = 0.9$

   Then T_1 = 0.9 * 100 = 90

   T_2 = 0.9 * 90 = 81

   .....
2. **Logarithmic Decay** $T(t) = \frac{c}{log\,(t+d)}$
3. **Linear Decay** $T_k = T_{k-1} - linear\ factor$

Table 11 Summary of the Best Performing Schedules in Each Category

	Exponential	Logarithmic	Linear	LinEx	Adaptive
Starting Temp	10	0.001	0.1	10	10
Number of Transitions per Temperature Step	500	500	500	500	500
Number of Temperature Steps	133	150	200	56	80
Cooling Ratio	0.9	N/A	N/A	Variable	Variable
Decrement Factor	N/A	N/A	0.0005	Variable	N/A
C Value (if applicable)	N/A	0.001	N/A	N/A	N/A
Stopping Temperature	1.00E-05	1.66E-04	0	1.00E-05	1.00E-05
Final Error	0.2104	0.2216	0.2236	0.22	0.218