Simple Annealing

import math

import random

def simulated\_annealing(start\_state, goal\_state, get\_neighbors, get\_cost, initial\_temp, cooling\_rate):

current = start\_state

current\_cost = get\_cost(current)

temp = initial\_temp

while temp > 0.1:

if current == goal\_state:

print("Goal reached!")

return current

neighbors = get\_neighbors(current)

if not neighbors:

print("No neighbors to move to.")

return current

next\_state = random.choice(neighbors)

next\_cost = get\_cost(next\_state)

delta\_e = current\_cost - next\_cost

probability = math.exp(delta\_e / temp) if delta\_e < 0 else 1

if random.random() < probability:

print(f"Moving from {current} to {next\_state} (Temp: {temp:.2f})")

current = next\_state

current\_cost = next\_cost

temp \*= cooling\_rate

print("Finished cooling. Best state found:", current)

return current

# ---------- USER INPUT SECTION ----------

def get\_neighbors\_func():

neighbors = {}

n = int(input("Enter number of states: "))

for \_ in range(n):

state = input("Enter state name: ")

adj = input(f"Enter neighbors of {state} (space-separated): ").split()

neighbors[state] = adj

return neighbors

def get\_cost\_func(states):

cost = {}

for state in states:

val = int(input(f"Enter cost for {state}: "))

cost[state] = val

return cost

def main():

start = input("Enter start state: ")

goal = input("Enter goal state: ")

neighbors\_map = get\_neighbors\_func()

cost\_map = get\_cost\_func(neighbors\_map.keys())

initial\_temp = float(input("Enter initial temperature (e.g., 100): "))

cooling\_rate = float(input("Enter cooling rate (e.g., 0.95): "))

def get\_neighbors(state):

return neighbors\_map.get(state, [])

def get\_cost(state):

return cost\_map[state]

result = simulated\_annealing(start, goal, get\_neighbors, get\_cost, initial\_temp, cooling\_rate)

print("Final result:", result)

if \_\_name\_\_ == "\_\_main\_\_":

main()

OUTPUT:

Enter start state: A

Enter goal state: D

Enter number of states: 4

Enter state name: A

Enter neighbors of A (space-separated): B C

Enter state name: B

Enter neighbors of B (space-separated): A D

Enter state name: C

Enter neighbors of C (space-separated): A D

Enter state name: D

Enter neighbors of D (space-separated): B C

Enter cost for A: 10

Enter cost for B: 8

Enter cost for C: 5

Enter cost for D: 1

Enter initial temperature (e.g., 100): 100

Enter cooling rate (e.g., 0.95): 0.9

Moving from A to B (Temp: 100.00)

Moving from B to D (Temp: 90.00)

Goal reached!

Final result: D