**LAB – 7**

**GREY WOLF OPTIMISATION**

**(APPLICATION USED JOB SCHEDULING)**

**CODE :**

import numpy as np

import random

def calculate\_makespan(sequence, processing\_times):

time = 0

for job in sequence:

time += processing\_times[job]

return time

def swap\_positions(sequence, idx1, idx2):

sequence = sequence.copy()

sequence[idx1], sequence[idx2] = sequence[idx2], sequence[idx1]

return sequence

def update\_position(current, alpha, beta, delta, a):

num\_swaps = max(1, int(a \* len(current)))

new\_seq = current.copy()

for \_ in range(num\_swaps):

idx1 = random.randint(0, len(current) - 1)

idx2 = alpha.index(new\_seq[idx1])

new\_seq = swap\_positions(new\_seq, idx1, idx2)

for \_ in range(num\_swaps):

idx1 = random.randint(0, len(current) - 1)

idx2 = beta.index(new\_seq[idx1])

new\_seq = swap\_positions(new\_seq, idx1, idx2)

for \_ in range(num\_swaps):

idx1 = random.randint(0, len(current) - 1)

idx2 = delta.index(new\_seq[idx1])

new\_seq = swap\_positions(new\_seq, idx1, idx2)

return new\_seq

def gwo\_job\_scheduling(processing\_times, population\_size=10, max\_iter=10):

num\_jobs = len(processing\_times)

population = [random.sample(range(num\_jobs), num\_jobs) for \_ in range(population\_size)]

alpha, beta, delta = None, None, None

alpha\_score, beta\_score, delta\_score = float('inf'), float('inf'), float('inf')

for iter in range(max\_iter):

fitness = []

for wolf in population:

makespan = calculate\_makespan(wolf, processing\_times)

fitness.append(makespan)

if makespan < alpha\_score:

alpha, alpha\_score = wolf, makespan

elif makespan < beta\_score:

beta, beta\_score = wolf, makespan

elif makespan < delta\_score:

delta, delta\_score = wolf, makespan

a = 2 - iter \* (2 / max\_iter) # Linearly decrease a from 2 to 0

new\_population = []

for wolf in population:

new\_seq = update\_position(wolf, alpha, beta, delta, a)

new\_population.append(new\_seq)

population = new\_population

print(f"Iteration {iter+1} - Best Makespan: {alpha\_score}")

return alpha, alpha\_score

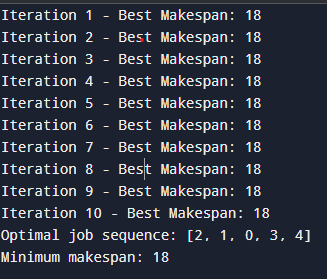
processing\_times = [2, 3, 7, 5, 1] # Processing times of 5 jobs

best\_sequence, best\_makespan = gwo\_job\_scheduling(processing\_times)

print("Optimal job sequence:", best\_sequence)

print("Minimum makespan:", best\_makespan)

**OUTPUT :**

****