**Lince Rumainum**
**DSA 5113**
**Advanced Analytics & Metaheuristics**
**Group 24 - HW 5**

**Problem 1 – Simulated Annealing**

**Part a**
To determine the initial temperature for this problem, the temperature values were increased until objective value is at least over 20,000.

**Part b**
The three different cooling schedules used for this problem are:
Cauchy schedule

$$t_k = \frac{T_0}{1 + k}$$

Boltzman schedule

$$t_k = \frac{T_0}{\log(1 + k)}$$

Alpha constant schedule

$$t_{k+1} = \alpha t_k$$

For number of iterations performed at a given temperature ($M_k$), a constant number, such as, 135 and 120 were used and also two different variations of greater iterations at lower temperatures were used. Both variations increase iterations when temperature drop to between 76%-90% of initial temperature, 51%-75% of initial temperature, 26%-50% of initial temperature, 6%-25% of initial temperature, down till it reaches 5% of initial temperature, which no iteration would be performed.

**Part c**
There are two different stopping criterion used for this problem. First method is when the specified total number of iterations (of the temperatures, *k*) has been reached. Second method is stopping when the cooling temperature reached 15% of the initial temperature.

So, using the maximum weight of **2,500** and **150** number selections, the results are in the table on the next page:

| Algorithm | $t_0$ | Cooling, $t_k$ | $M_k$ | # of temps | Iterations | # Items Selected | Weight | Objective |
|---|---|---|---|---|---|---|---|---|
| Simulated Annealing (k = 100, Mk = 30, 60, 105, 120, 135) | 1000 | $t_0/(1+k)$ | variation-1 | 100 | 2,430 | 23 | 2,484.10 | 20,824.40 |
| Simulated Annealing (k = 100, Mk = 30, 60, 105, 120, 135) | 1000 | $t_0/\log(1+k)$ | variation-1 | 100 | 12,360 | 28 | 2,496.50 | 28,618.00 |
| Simulated Annealing (k = 100, Mk = 30, 60, 105, 120, 135) | 1000 | 0.91tk-1 | variation-1 | 100 | 3,735 | 30 | 2,481.20 | 28,701.20 |
| Simulated Annealing (k = 100, $M_k$ = 50, 100, 200, 400, 800) | 1000 | $t_0/(1+k)$ | variation-2 | 100 | 13,650 | 27 | 2,498.60 | 29,451.10 |
| Simulated Annealing (k = 100, $M_k$ = 50, 100, 200, 400, 800) | 1000 | $t_0/\log(1+k)$ | variation-2 | 100 | 376,550 | 30 | 2,488.40 | 31,962.90 |
| Simulated Annealing (k = 100, $M_k$ = 50, 100, 200, 400, 800) | 1000 | $0.91t_{k-1}$ | variation-2 | 100 | 17,500 | 29 | 2,495.20 | 31,036.30 |
| Simulated Annealing (k = 75) | 800 | $t_0/(1+k)$ | 135 | 75 | 10,125 | 23 | 2,490.90 | 21,378.60 |
| Simulated Annealing (k = 75) | 800 | $t_0/\log(1+k)$ | 135 | 75 | 10,125 | 29 | 2,490.90 | 27,573.20 |
| Simulated Annealing (k = 75) | 800 | $0.91t_{k-1}$ | 135 | 75 | 10,125 | 28 | 2,497.70 | 28,909.30 |
| Simulated Annealing (stop at 15% of $t_0$) | 750 | $t_0/(1+k)$ | 120 | 6 | 720 | 23 | 2,492.70 | 22,344.50 |
| Simulated Annealing (stop at 15% of $t_0$) | 750 | $t_0/\log(1+k)$ | 120 | 785 | 94,200 | 28 | 2,482.80 | 32,353.20 |
| Simulated Annealing (stop at 15% of $t_0$) | 750 | $0.91t_{k-1}$ | 120 | 21 | 2,520 | 26 | 2,496.80 | 28,202.10 |

**Problem 2 – Metaheuristics and Metaphors**

Combination of random restarts, random walk, hill climbing with best improvement, and simulated annealing. The pseudocode:

```
input: starting solution, s_0
input: neighborhood operator, N
input: evaluation function, f
input: initial temperature, t_0
input: the cooling schedule, t_k
input: the number of iterations for each temperature, M_k
input: the number of iterations for random restart, R
input: probability to use random walk-simulated annealing, p
input: probability to use best improvement, 1-p
r ← 0
current ← s_0
while restart counter <= R do
    best_neighbor ← current
    p_current ← random probability between [0,1]
    if p_current < 1-p
        for each s in N(current) do
            if f(s) < f(best_neighbor) then
                best_neighbor ← s
            end if
        end for
    else
        while stopping criterion not met do
            m ← 0
            while m < M_k do
                s – randomly selected solution from N(current)
                if f(s) < f(best_neighbor) then
                    best_neighbor ← s
                    current ← s
                else
                    delta ← f(s) - f(current)
                    epsilon ← a random number, uniformly drawn from [0,1]
                    if epsilon <= e^(-delta/tk) then
                        best_neighbor ← s
                        current ← s
                    end if
                end if
                m ← m +1
            end while
            k ← k+1
        end while
    end if
end while
```

**Extra Credit:**

(i) Lunar Phases Metaheuristics

(ii) Lunar Phases Metaheuristics is the phenomenon where time are important factors. The time determine the initial solution and what feasible solutions are "shown" according to the moon rotation. There are eight lunar phases to consider: New Moon, Waxing Crescent, First Quarter, Waxing Gibbous, Full Moon, Waning Gibbous, Third Quarter, and Waning Crescent. These different phases can either be robust and efficient or useless (end up showing only infeasible solutions). Other than the full moon phase, other phases with non-visible sides of the moon are consider tabu since the "invisible" side of the moon is full of unknown noises or unknown path. On the case of the New Moon phase, it helps show that a new initial solution is needed since that initial solution are most likely to be infeasible solution. Knowing these lunar phases (rotations of the moon) can help with fast forward to the right timing of the optimal solution. Although full moon would be the ideal phase and the one with 100% probability of the optimal solution, by using the other phase, the timing would be much more efficient than evaluating the full moon phase.

**Problem 3 – More Extra-Credit (Tabu Search)**

The tabu criterion is selection in the neighborhood that have less than 23 selections. Those values will be considered tabu-active for a period of time (called tabu tenure). The aspiration criterion to override this tabu search will be allowing those selections as long as they have feasible solutions.

***Code files:***

Group25_HW5_p1.py, Group25_HW5_tabuSearch.py.