CS303 Project3: Solving Influence Maximization Problem Using the IMM Algorithm

Shijie Chen
Department of Computer Science and Engineering
Southern University of Science and Technology
Shenzhen, Guangdong, China
Email: 11612028@mail.sustc.edu.cn

***Abstract*—**

I. Preliminaries

II. Methodology

A. Notation

The notations used in this report is shown in the table below.

TABLE I
REPRESENTATION

Name	Variable
All tasks	tasks
Tasks remain	undone

B. Genetic Algorithm

Algorithm 1 Genetic Algorithm Framework

```
population ← initPopulation()
size ← len(population)
while end condition not met do
    offSpring ← genOffspring(population)
    population ← population + offSpring
    population.sort(key = cost)
    population ← population[0 : size]
end while
    return population[0]
```

1) Framework:

2) Initial Population:

- Ulusoy split Ulusoy Split works by converting the tasks to be done to a directed acyclic graph (DAG). Then obtain the optimal solution by searching for a shortest path in the DAG. Details of the algorithm can be found elsewhere. XXXXX

Algorithm 2 Ulusoy Split

```
function ULUSOYSPLIT(tasks, depot, shortestPath, load)
    DAG, incoming, outgoing ← toDAG(tasks)
    for node ∈ DAG do
        minCost ← inf
        bestEdge ← minCost(incoming[node])
        node.bestPath ← bestEdge.bestPath.append(node)
        node.cost ← bestEdge.cost
        for edge ∈ outgoing[node] do
            edge.cost ← edge.cost + node.cost
            edge.bestPath ← node.bestPath
        end for
    end for
    x ← 1
     return node[−1].cost, node[−1].bestPath
end function
function TODAG(tasks, depot, shortestPath, load)
    construct the DAG
     return DAG
end function
```

- Generalized Path Scanning

3) Genetic Operators:

III. Validation

IV. Discussion

V. Conclusion

Acknowledgment

The authors would like to thank the TAs for their hint in the Lab and maintaining a online runtime platform.