a) Let a state be represented by the set V',such that V'contains the vertices we have added so far to the partial solution, and the cost represented by the cardinality of V'. The neighbor function N(x), where x is the current state, will produce the neighbors of x by adding one valid vertex to x OR by removing one vertex from x. The neighbors will be evaluated incrementally by adding 1 to their total cost if a vertex is added and subtracting 1 if a vertex is removed. The cost will also subtract the total number of edges that are covered since this is a desirable quality. The independent set problem can be solved by taking the complement of the optimal solution to the vertex cover problem V/V' [1]. Therefore the ISP can also be solved using this method [2].

b) According to [1] and [3] we will start with an initial state, which in our case will be 0 vertices in V'. Choose an appropriate temperature function and set temp to be the max temp initially. We will use the neighbor function defined above to randomly choose a neighbor of our current state. We will input this new state along with the current state and the temperature to the probability function and decide whether or not to move to the new state or stay where we are. The temperature will determine how willing we are to go to a less optimal state.

c) Representation 1: Will simply contain the customer ID as well as the route they are coming from and going to
Representation 2: This, slightly more memory intensive representation will contain the customer, to and from routes as well as cost of the move and total cost at the previous step
Representation 3: this will be the most memory intensive representation containing everything from 1 and 2 as well as all possible routes the customer could have taken instead of the one they did as well as the associated costs.

d)

Algorithm Tabu MIS

```
    **Initialize** $S_0$ to be an empty set
    Let the **tabu list** be the set of previous solutions (sped up by hash table [1])
    Let the **intensification** list be the vertices that appear most in the best solutions
    Let the **diversification** be a restart strategy if a previous local minimum is picked
    Let the **aspiration** criteria be if a move is better solution than best found
    Let the **stopping condition** be a check of the complement of the current solution
    to see if it covers all edges
    **Loop**
        Find best admissible neighbor(s) $s'$
        s = $s'$
        Update tabu list, aspiration conditions, medium and long term memories
        **If** intensification criterion holds **Then** intensification
        **If** diversification criterion holds **Then** diversification
    **Until** stopping condition
    **Output** best solution
```

This algorithm is based closely on [1] and [4] with the addition of the explained memory, aspiration, and stopping conditions. According to [1] the fitness landscape is defined as the tuple (G,f) where g is the graph of the search space and f is the objective function. In the example described above the fitness landscape will be similar to that of the vertex cover problem since the search space and objective function are closely related. However, this would be less similar to the TSP described in [4] which verifies the “no free lunch” theorem stating that not all metaheuristic approaches will work in all scenarios.

References

[1]El-Ghazali Talbi. Metaheuristics From Design to Implementation. Wiley and Sons, 2009
[2]https://en.wikipedia.org/wiki/Vertex_cover
[3]https://en.wikipedia.org/wiki/Simulated_annealing
[4]https://en.wikipedia.org/wiki/Tabu_search