84

UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN
FACULTAD DE INGENIERÍA MECÁNICA Y ELÉCTRICA
TIPO DE EXAMEN Y/O EVALUACIÓN:
FINAL ORDINARIO (*Final Exam*)

MATERIA/UNIDAD DE APRENDIZAJE: Temas Selectos de Optimización

LEARNING UNIT: Selected Topics on Optimization (in English)
SEMESTER: January – June 2024
ACADEMY: Statistics and Operations Research (*Estadística e Investigación de Operaciones*).
INSTRUCTOR: Dr. Roger Z. Ríos Mercado

STUDENT NAME: **SAUL EMMANUEL CARREON ALANIS 1901555 ITS**

ANSWER SHEET

Section 1

1. Heuristics are appropriate for solving combinatorial optimization problems when: The problem size is large, solutions are needed quickly, near optimal solutions are acceptable, the problem is NP-hard

2. A combinatorial optimization problem is considered "hard" to solve when it belongs to a class of problems known as NP-hard and the solution space grows exponentially with input size.

3. A problem is considered "easy" if it can be solved in polynomial time. the time it takes to solve the problem grows with the size of the input.

4. No, constructive heuristics do not guarantee feasibility
the feasibility depends on the rules used and constraints may prevent finding a feasible solution.

5. A construction heuristic builds a solution from scratch by adding elements iteratively based on rules of heuristic.

6. A local search heuristic starts with an initial solution and iteratively explores neighboring solutions to find improvements.

7. The 2-OPT heuristic for the Traveling Salesman Problem (TSP) is a local search algorithm that iteratively improves a given tour by reversing the order of segments of the tour to reduce the total travel distance.

1. Initial Tour: Start with an initial tour.
2. Select Two Edges: At each iteration, select two edges in the tour that do not share a vertex.
3. Reverse Segment: Remove these two edges and reconnect the tour by reversing the segment between the selected edges, creating a new tour.
4. Evaluate: Calculate the total distance of the new tour.
5. Accept or Reject: If the new tour is shorter, accept it as the current tour. Otherwise, revert to the previous tour.
6. Repeat: Continue this process until no further improvements can be made by any 2-opt move.

8. To compare two different heuristics for a given combinatorial optimization problem, we can use the following:
compare the solutions produced by each heuristic, on base of the objective function value, Record the time each heuristic takes to produce a solution, how well the heuristics handle increasing problem sizes, For heuristics that do not guarantee feasible solutions, compare the rate at which they find feasible solutions.

Section 2

9a. $x^{(1)} = (\{3,7,8\}, \{1,4,9\}, \{2,5,10\})$ is a feasible solution,
because the solution must divide the original set V into exactly p subsets. In this case, p is equal to 3.
Each subset must contain different elements.
All elements of V must be in at least one subset.

9b. $x^{(2)} = (\{2,4,6,,8,10\}, \{1,5,9\}, \{3,7\})$ is a feasible solution,
because the solution must divide the original set V into exactly p subsets. In this case, p is equal to 3.
Each subset must contain different elements.
All elements of V must be in at least one subset.

9c. The objective is to minimize so the best result is $x^{(4)} = 169$ the next one is $x^{(5)} = 343$ and last $x^{(3)}$ = 355

Solution . $x^{(3)} = (\{1,5,9\}, \{2,4,6,8,10\}, \{3,7\})$
$\{1,5,9\}$: $d_{15} + d_{19} + d_{59} = 31 + 20 + 11 = 62$
$\{2,4,6,8,10\}$:$d_{24} + d_{26} + d_{28} + d_{210} + d_{46} + d_{48} + d_{410} + d_{68} + d_{610} + d_{810} = 9 + 33 + 39 + 30 + 12 + 8 + 18 + 29 + 32 + 46 = 256$
$\{3,7\}$:$d_{37} = 37$
Total = 355
Solution $x^{(4)} = (\{2,5,9\}, \{1,3,4,8\}, \{6,10\})$
$\{2,5,9\}$: $d_{25} + d_{29} + d_{59} = 19 + 14 + 11 = 44$
$\{1,3,4,8\}$:$d_{13} + d_{14} + d_{18} + d_{34} + d_{38} + d_{48} = 26 + 14 + 35 + 25 + 20 + 8 = 93$
$\{6,10\}$:$d_{610} = 32$
Total=169

Solution $x^{(5)} = (\{1,3,7,8\}, \{2,5,10\}, \{4,6,9\})$
$\{1,3,7,8\}$:$d_{13} + d_{17} + d_{18} + d_{37} + d_{38} + d_{78} = 26 + 46 + 35 + 37 + 20 + 36 = 200$
$\{2,5,10\}$: $d_{25} + d_{210} + d_{510} = 19 + 30 + 27 = 76$
$\{4,6,9\}$: $d_{46} + d_{49} + d_{69} = 12 + 15 + 40 = 67$
Total=343

9d. Pseudocode for Constructive Heuristic:
Input: n objects, p clusters, distance matrix D
Output: Feasible solution X

1. Initialize p empty clusters: X = {X1, X2, ..., Xp}
2. Sort all pairs (i, j) in ascending order of distance dij
3. For each pair (i, j) in sorted order:

   a. If i and j are not yet in the same cluster and can be added to any cluster without violating feasibility:
      i. Add i and j to the same cluster with the least total distance increase
4. Assign any remaining objects to the cluster with the least total distance increase
5. Return X

9e.
Create p=3 empty clusters:
X={{},{},{}}

Sort all pairs (i,j) in ascending order of dij
(1,10),(2,4),(3,5),(4,8),(7,9),(1,2),(3,6),(4,9),(6,7),(8,9)

Iterate over the sorted pairs and assign objects to clusters.

First Pair: (1, 10)

Distance: 10

Assign to $X1$ X={{1,10},{},{}}X={{1,10},{},{}}

Second Pair: (2, 4)

Distance: 9

Assign to $X2$ X={{1,10},{2,4},{}}

Third Pair: (3, 5)

Distance: 9

Assign to $X3$ X={{1,10},{2,4},{3,5}}**Fourth Pair: (4, 8)**

Distance: 8

4 is already in $X2$, assign 8 to $X2$ X={{1,10},{2,4,8},{3,5}}

Fifth Pair: (7, 9)

Distance: 9

Assign to $X1$ X={{1,10,7,9},{2,4,8},{3,5}}

Remaining object: {6}

Assign 6 to the cluster with the least total distance increase.

6 can be assigned to $X1$, $X2$, or $X3$.

Check distance increases.

Assign 6 to $X3$ (assumption based on checking distances): X={{1,10,7,9},{2,4,8},{3,5,6}}

Solution

X={{1,10,7,9},{2,4,8},{3,5,6}}

{1,10,7,9}$d_{110} + d_{17} + d_{19} + d_{107} + d_{109} + d_{79} = 10 + 18 + 20 + 21 + 9 + 9 = 87$

{2,4,8}: $d_{24} + d_{28} + d_{48} = 9 + 35 + 8 = 52$

{3,5,6}: $d_{35} + d_{37} + d_{13} = 59$

Total =198

Conclusion

The new solution has a total of 198, which is significantly better than the total of for solution $x^{(3)}$

9f. pseudocode for Local Search Heuristic

Input: Initial feasible solution X, Distance matrix D
Output: Improved feasible solution X'

```
Initialize: X' = X

repeat
   best_move = None
   best_dissimilarity_change = 0

   for each cluster Xa in X:
       for each object i in Xa:
           for each cluster Xb in X where Xb ≠ Xa:
               // Single Transfer Move
               dissimilarity_change = compute_dissimilarity_change_single_transfer(i, Xa, Xb, D)
               if dissimilarity_change < best_dissimilarity_change:
                   best_move = (i, Xa, Xb)
                   best_dissimilarity_change = dissimilarity_change

               // Swap Move
               for each object j in Xb:
                   dissimilarity_change = compute_dissimilarity_change_swap(i, j, Xa, Xb, D)
                   if dissimilarity_change < best_dissimilarity_change:
                       best_move = (i, j, Xa, Xb)
                       best_dissimilarity_change = dissimilarity_change

   if best_move is None:
       break // No improvement found

   apply_best_move(best_move, X')

until no improvement found or max_iterations reached

return X'
```

9g.