**CMPE536 Metaheuristics**

**Fall 2020-2021**

**Assignment I**

**Assist. Prof. Dr. Ahmet Ünveren**

****

**Submitted By**

**Fırat Doğan**

**20500070**

1. **Problem**

Quadratic Assignment Problem is one of the combinatorial optimization problem in the category of optimization research area. A set of n facilities and a set of n locations exist. A distance is defined for each pair of locations and a weight or flow is specified for each pair of facilities (e.g., the amount of supplies transported between the two facilities). In order to minimize the sum of the distances multiplied by the corresponding flows, the problem is to allocate all facilities to various locations.

1. **Dataset**

**Tai12a.dat**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 27 | 85 | 2 | 1 | 15 | 11 | 35 | 11 | 20 | 21 | 61 |
| 27 | 0 | 80 | 58 | 21 | 76 | 72 | 44 | 85 | 94 | 90 | 51 |
| 85 | 80 | 0 | 3 | 48 | 29 | 90 | 66 | 41 | 15 | 83 | 96 |
| 2 | 58 | 3 | 0 | 74 | 45 | 65 | 40 | 54 | 83 | 14 | 71 |
| 1 | 21 | 48 | 74 | 0 | 77 | 36 | 53 | 37 | 26 | 87 | 76 |
| 15 | 76 | 29 | 45 | 77 | 0 | 91 | 13 | 29 | 11 | 77 | 32 |
| 11 | 72 | 90 | 65 | 36 | 91 | 0 | 87 | 67 | 94 | 79 | 2 |
| 35 | 44 | 66 | 40 | 53 | 13 | 87 | 0 | 10 | 99 | 56 | 70 |
| 11 | 85 | 41 | 54 | 37 | 29 | 67 | 10 | 0 | 99 | 60 | 4 |
| 20 | 94 | 15 | 83 | 26 | 11 | 94 | 99 | 99 | 0 | 56 | 2 |
| 21 | 90 | 83 | 14 | 27 | 77 | 79 | 56 | 60 | 56 | 0 | 60 |
| 61 | 51 | 96 | 71 | 76 | 32 | 2 | 70 | 4 | 2 | 60 | 0 |

Table 1.1 Matrix A (d): distance between each location

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 21 | 95 | 82 | 56 | 41 | 6 | 25 | 10 | 4 | 63 | 6 |
| 21 | 0 | 44 | 40 | 75 | 79 | 0 | 89 | 35 | 9 | 1 | 85 |
| 95 | 44 | 0 | 84 | 12 | 0 | 26 | 91 | 11 | 35 | 82 | 26 |
| 82 | 40 | 84 | 0 | 69 | 56 | 86 | 45 | 91 | 59 | 18 | 76 |
| 56 | 75 | 12 | 69 | 0 | 39 | 18 | 57 | 36 | 61 | 36 | 21 |
| 41 | 79 | 0 | 56 | 39 | 0 | 71 | 11 | 29 | 82 | 82 | 6 |
| 6 | 0 | 26 | 86 | 18 | 71 | 0 | 71 | 8 | 77 | 74 | 30 |
| 25 | 89 | 91 | 45 | 57 | 11 | 71 | 0 | 89 | 76 | 76 | 40 |
| 10 | 35 | 11 | 91 | 36 | 29 | 8 | 89 | 0 | 93 | 56 | 1 |
| 4 | 9 | 35 | 59 | 61 | 82 | 77 | 76 | 93 | 0 | 50 | 4 |
| 63 | 1 | 82 | 18 | 36 | 82 | 74 | 76 | 56 | 50 | 0 | 36 |
| 6 | 85 | 26 | 76 | 21 | 6 | 30 | 40 | 1 | 4 | 36 | 0 |

Table 1.2 ⦁ Matrix B (f) : flow between each facility

**Tai17a.dat**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 20 | 86 | 4 | 77 | 15 | 89 | 48 | 14 | 89 | 44 | 59 | 22 | 57 | 63 | 6 | 0 |
| 20 | 0 | 62 | 41 | 62 | 46 | 25 | 75 | 76 | 40 | 66 | 58 | 30 | 68 | 78 | 91 | 13 |
| 86 | 62 | 0 | 59 | 49 | 85 | 84 | 8 | 38 | 41 | 56 | 39 | 53 | 77 | 50 | 30 | 58 |
| 4 | 41 | 59 | 0 | 55 | 19 | 85 | 52 | 34 | 53 | 40 | 69 | 12 | 85 | 72 | 7 | 49 |
| 77 | 62 | 49 | 55 | 0 | 46 | 87 | 58 | 17 | 68 | 27 | 21 | 6 | 67 | 26 | 82 | 44 |
| 15 | 46 | 85 | 19 | 46 | 0 | 35 | 3 | 62 | 8 | 51 | 1 | 91 | 39 | 87 | 72 | 45 |
| 89 | 25 | 84 | 85 | 87 | 35 | 0 | 96 | 7 | 87 | 68 | 33 | 3 | 21 | 90 | 45 | 47 |
| 48 | 75 | 8 | 52 | 58 | 3 | 96 | 0 | 25 | 30 | 43 | 97 | 33 | 35 | 61 | 42 | 36 |
| 14 | 76 | 38 | 34 | 17 | 62 | 7 | 25 | 0 | 43 | 7 | 84 | 6 | 0 | 0 | 48 | 62 |
| 89 | 40 | 41 | 53 | 68 | 8 | 87 | 30 | 43 | 0 | 59 | 29 | 94 | 82 | 29 | 3 | 3 |
| 44 | 66 | 56 | 40 | 27 | 51 | 68 | 43 | 7 | 59 | 0 | 51 | 67 | 39 | 15 | 66 | 42 |
| 59 | 58 | 39 | 69 | 21 | 1 | 33 | 97 | 84 | 29 | 51 | 0 | 23 | 62 | 62 | 28 | 76 |
| 22 | 30 | 53 | 12 | 6 | 91 | 3 | 33 | 6 | 94 | 67 | 23 | 0 | 66 | 82 | 98 | 35 |
| 57 | 68 | 77 | 85 | 67 | 39 | 21 | 35 | 0 | 82 | 39 | 62 | 66 | 0 | 15 | 17 | 77 |
| 63 | 78 | 50 | 72 | 26 | 87 | 90 | 61 | 0 | 29 | 15 | 62 | 82 | 15 | 0 | 44 | 26 |
| 6 | 91 | 30 | 7 | 82 | 72 | 45 | 42 | 48 | 3 | 66 | 28 | 98 | 17 | 44 | 0 | 76 |
| 0 | 13 | 58 | 49 | 44 | 45 | 47 | 36 | 62 | 3 | 42 | 76 | 35 | 77 | 26 | 76 | 0 |

Table 2.1 ⦁ Matrix A (d): distance between each location

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 21 | 95 | 82 | 56 | 41 | 6 | 25 | 10 | 4 | 63 | 6 | 44 | 40 | 75 | 79 | 0 |
| 21 | 0 | 89 | 35 | 9 | 1 | 85 | 84 | 12 | 0 | 26 | 91 | 11 | 35 | 82 | 26 | 69 |
| 95 | 89 | 0 | 56 | 86 | 45 | 91 | 59 | 18 | 76 | 39 | 18 | 57 | 36 | 61 | 36 | 21 |
| 82 | 35 | 56 | 0 | 71 | 11 | 29 | 82 | 82 | 6 | 71 | 8 | 77 | 84 | 30 | 89 | 76 |
| 56 | 9 | 86 | 71 | 0 | 76 | 40 | 93 | 56 | 1 | 50 | 4 | 36 | 27 | 85 | 2 | 1 |
| 41 | 1 | 45 | 11 | 76 | 0 | 15 | 11 | 35 | 11 | 20 | 21 | 61 | 80 | 58 | 21 | 76 |
| 6 | 85 | 91 | 29 | 40 | 15 | 0 | 72 | 44 | 85 | 94 | 90 | 51 | 3 | 48 | 29 | 90 |
| 25 | 84 | 59 | 82 | 93 | 11 | 72 | 0 | 66 | 41 | 15 | 83 | 96 | 74 | 45 | 65 | 40 |
| 10 | 12 | 18 | 82 | 56 | 35 | 44 | 66 | 0 | 54 | 83 | 14 | 71 | 77 | 36 | 53 | 37 |
| 4 | 0 | 76 | 6 | 1 | 11 | 85 | 41 | 54 | 0 | 26 | 84 | 76 | 91 | 13 | 29 | 11 |
| 63 | 26 | 39 | 71 | 50 | 20 | 94 | 15 | 83 | 26 | 0 | 77 | 32 | 87 | 67 | 94 | 79 |
| 9 | \*1 | 18 | 8 | 4 | 21 | 90 | 83 | 14 | 87 | 77 | 0 | 2 | 10 | 99 | 56 | 70 |
| 44 | 11 | 57 | 77 | 36 | 61 | 51 | 96 | 71 | 76 | 32 | 2 | 0 | 99 | 60 | 4 | 56 |
| 40 | 35 | 36 | 74 | 27 | 80 | 3 | 74 | 77 | 91 | 87 | 10 | 99 | 0 | 2 | 60 | 72 |
| 75 | 82 | 61 | 30 | 85 | 58 | 48 | 45 | 36 | 13 | 67 | 99 | 60 | 2 | 0 | 74 | 46 |
| 79 | 26 | 36 | 89 | 2 | 21 | 29 | 65 | 53 | 59 | 94 | 56 | 4 | 60 | 74 | 0 | 13 |
| 0 | 69 | 21 | 76 | 1 | 76 | 90 | 40 | 37 | 11 | 79 | 70 | 56 | 72 | 46 | 13 | 0 |

Table 2.2 ⦁ Matrix B (f) : flow between each facility

1. **Algorithm : Iterated Local Search**

Iterated local search algorithm is step forward of local search algorithm.Local Search Algorithm can stuck in a local minimum, where no improving neighbors are available. In order to avoid this problem we can use iterated local search algorithm.The local search methodologies are applied to iterated local search algorithm (best search, first search, random search) and we build the algorithm in a loop to make it iterative. Until the criterion occur, the algorithm iterate. While every iteration, the solution is perturbed (kicked out from the local) and the algorithm checks the local solution to the current one. If the local solution is better(less) than the current solution, we assign the local solution to the current solution.

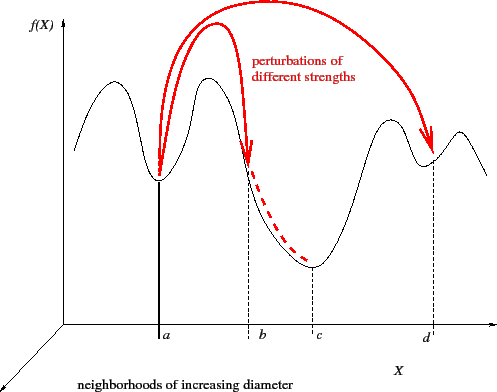


Image 3.1 Iterated local search kicks a solution out from a local optimum

1. **Representation**

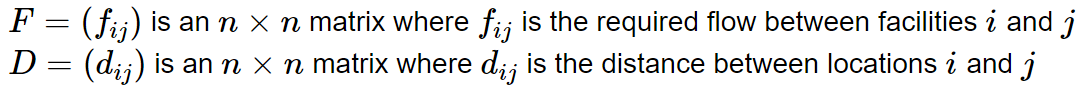
Given a set of facilities and locations along with the flows between facilities and the distances between locations, the objective of the Quadratic Assignment Problem is to assign each facility to a location in such a way as to minimize the total cost.

**Sets**

N={1,2,⋯,n}

Sn =ϕ:N→N is the set of all permutations

**Parameters**



**Object Function**

****

1. **Applying Algorithm**

As a local search method, I have used **Select Best Improvement** to find solutions’ the best neighbor. Thus, it selects the best among them.

As a perturbation technique, I have used the **Swap** technique. In the swap, I swapped 3 of the elements with each other. In my point of view, 3 is the good number in order to not get stuck in a local and not become a randomized solution.

About the iteration, I decided to make it 500 at first, but when I run the algorithm in matlab 10 times, it doesn’t find the optimum solution at 3 out of 10 runs. Thus, I decided to make it 1000 iterations to find the optimum value every time.

1. **Results**

In the Iterated Local Search algorithm, It became successful and found the optimal solution when the data size is 12.

**Tai12a.dat Dataset (n=12)**

**newSol = 8 5 11 11 1 4 12 10 9 3 6 2 (Iteration 1)**

**newSol = 8 5 7 11 1 4 8 10 9 3 6 2 (Iteration 2)**

**.**

**.**

**.**

**.**

**.**

**newSol = 8 11 6 2 11 10 3 5 9 7 12 4 (Iteration 997)**

**newSol = 8 11 6 2 1 10 3 5 11 7 12 4 (Iteration 998)**

**newSol = 8 9 6 2 1 10 3 5 11 7 10 4 (Iteration 999)**

**sprime = 8 1 6 2 11 10 3 5 9 7 12 4 (Iteration 1000)**

**Best Solution cost**

**224416 (optimal)**

**Iteration Number**

**1000**

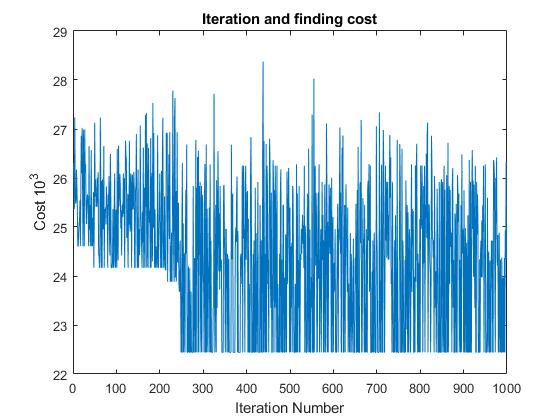
****

Image 6.1 Tai12a.dat Dataset

**Tai17a.dat Dataset (n=17)**

When the Tai17a.dat dataset is used, For 1000 iterations the algorithm couldn’t found the optimal solution but it found near optimal solution ;

**Attempt 1**

**sprime = 13 2 6 14 16 8 12 5 3 9 1 17 15 7 11 10 4**

**Best Solution cost**

**506088**

**Iteration Number**

**1000**

When the iteration size increase to 2000

**Attempt 2**

**sprime = 5 3 17 6 16 8 13 2 4 12 14 10 1 9 11 7 15**

**Best Solution cost**

**497846**

**Iteration Number**

**2000**

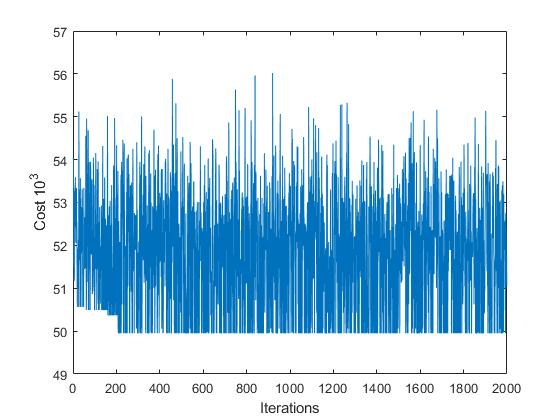
****

Image 6.2 Image 6.3 Tai17a.dat Dataset(Attempt4)

**Attempt 3**

**sprime = 12 2 6 7 4 8 14 5 11 3 16 13 17 9 1 10 15**

**Best Solution cost**

**491812 (optimal)**

**Iteration Number**

**2000**

It find better near optimal solution.

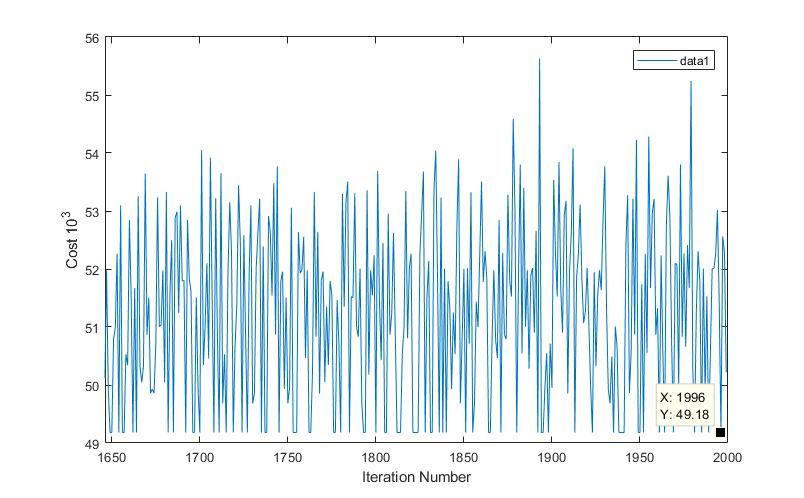
****

Image 6.3 Tai17a.dat Dataset(Attempt3)

**Attempt 4**

**sprime = 2 12 1 16 17 8 10 4 14 5 9 7 11 13 6 3 15**

**Best Solution cost**

**493662**

**Iteration Number**

**2000**

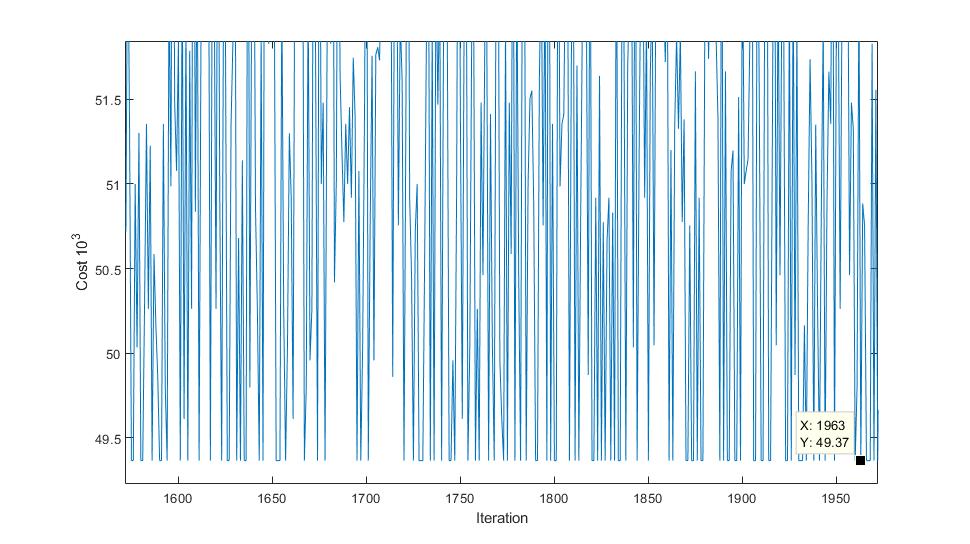


Image 6.4 Tai17a.dat Dataset(Attempt 4)

1. **Conclusion**

While the Iterated Local Search works well when the size of facilities and locations are 12, it can’t find every time the best optimum when the size is 17. Although, when the size is 17 and even iteration number is 10,000 and perturbation swap elements is 5, it may not find the best possible solution every time. Even if instead of perturbation, I used random solutions and found worse solutions. In conclusion, iterated local search works well with the data size of 12 but not well for 17 or upper for Quadratic Assignment Problem.

1. **References**

* Neos Guide, Quadratic Assignment Problem, avaible at : <https://neos-guide.org/content/quadratic-assignment-problem>
* Wikipedia, Iterated Local Search, avaible at : <https://en.wikipedia.org/wiki/Iterated_local_search>
* Wikipedia, Auadratic Assignment Problem, avaible at : <https://en.wikipedia.org/wiki/Quadratic_assignment_problem>