

# Implementation of Brute Force algorithm in Quadratic Assignment Problem

Darwin - 13508102  
Informatics Engineering  
School Of Electrical and Informatics Engineering  
Bandung Institute of Technology, Ganesha road No.10, Bandung 40132, Indonesia  
Email : Dawn0689@aol.com

Brute-force is a straightforward algorithm that can be used to solve almost every problem that cannot be solved using a more sophisticated algorithm. The drawback in using the brute-force algorithm is that it consumes a lot of performance related quality.

Quadratic assignment problem is a combinatorial problem that appears because of the allocation of some facilities that needed interaction between each other causes the cost and the flow of the activity becomes somewhat a problem. What we need to do is to minimize the cost as small as possible so that the cost and the flow of the activity can be reduced to minimum. It is one of the most challenging combinatorial optimization problem because it have a lot of possibilities especially when we have a lot of facilities and locations.

**Index Terms** — brute-force, combinatorial, performance, quadratic assignment problem.

## I. INTRODUCTION

Quadratic assignment problem is one of the fundamental combinatorial optimization problems in the branch of operation research in mathematics from the category of the facilities location problems. When we are placing new facilities, if the new facilities need to interact with old facilities, all it causes is only the linear assignment problem. But when the new facilities wanted to interact with other new facilities too, this causes quadratic assignment problem. When we have a lot of facilities which needed to interact with each other, there will be a lot problem when we calculate it manually because the time that is needed to do this work will be wasted a lot. The problem that we are facing usually is that we have  $n$  locations and  $n$  facilities that we need to allocate in each location so that the cost is minimized.

What considered as a problem in here is the flow between each interacting facilities times the distance of the interacting facilities will give us the cost. If the cost of the calculated distance and flow is small then it means that the location of the facilities that we put together has been optimized. To acquire this optimization we need to do a lot of combination of locations to acquire the most minimum cost between each facility.

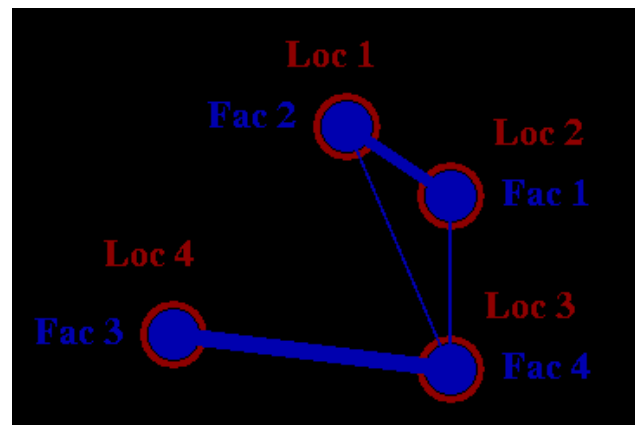


FIGURE 1 SHOWS THE LOCATION AND THE FACILITIES

## II. ALGORITHMS

Brute Force algorithm is one the most powerful algorithm. The reason why this algorithm can solve almost every problem is because it can solve what other algorithms can solve. The disadvantage of using this algorithm is that it usually has a high complexity. A high complexity will gives us a longer waiting time for the result. This will happen because it needs to find all the possibilities of what it presumes as the most optimized result if the case of using the brute force is for optimizing. Because of its powerfulness, which it could solve what other algorithms such as Breadth First Search algorithm, Dynamic Programming algorithm, in spite of its complexity, many programmers prefer to use this for problem solving.

The difference of what the brute force algorithm compared to other algorithms other than complexity, which we have mentioned before, is that it pin point every problem from a much more general view where other algorithms could only solve the given problem for a specific type of problem. This gives it the upper hand especially when people need to solve the problem when they do not know how to implement a lower complexity algorithm.

### III. QUADRATIC ASSIGNMENT PROBLEM

#### A. Introduction

Facility is one of the most important parts of an industry. Knowing that each facility has their own distance and flow when moving object from one to another, surely the allocation of the facility will make a huge difference in pressing the cost. In order to have the lowest cost possible we need to plan how to allocate each facility in their best location. Sometimes when we have industrial facility such as a mechanical one needs to interact with each other. This interaction between new facility and the old one will give us a linear assignment problem while another interaction between new facility and new facility could cause us quadratic assignment problem.

#### B.1 Linear Assignment Problem

The linear assignment problem consists of finding a maximum weight matching in a weighted bipartite graph. In its most general form, the problem is as follows :

There are a number of agents and a number of tasks. Any agent can be assigned to perform any task, incurring some cost that may vary depending on the agent-task assignment. It is required to perform all tasks by assigning exactly one agent to each task in such a way that the total cost of the assignment is minimized.

If the numbers of agents and tasks are equal and the total cost of the assignment for all tasks is equal to the sum of the costs for each agent (or the sum of the costs for each task, which is the same thing in this case), then the problem is called the linear assignment problem. Commonly, when speaking of the Assignment problem without any additional qualification, then the linear assignment problem is meant.

#### B.1 Quadratic Assignment Problem

The quadratic assignment problem is one of the most fundamental combinatorial optimization problems in the branch of operations research in mathematics, from the category of the facilities location problems. The problem is that when we have new facility that needed to interact with old facilities, it can be drawn as a linear assignment problem but when we have the new facilities that also needed to interact with each other, we cannot draw the problem as the linear assignment problem anymore. This kind of problem is what makes the quadratic assignment problem as a problem that is challenging.

The following will be given a mode of the problem about quadratic assignment problem :

There are a set of  $n$  facilities and a set of  $n$  locations. For each pair of locations, a distance is specified and for each pair of facilities a weight or flow is specified (e.g., the amount of supplies transported between the two facilities). The problem is to assign all facilities to

different locations with the goal of minimizing the sum of the distances multiplied by the corresponding flows.

The problem statement resembles that of the assignment problem, only the cost function is expressed in terms of quadratic inequalities, hence the name.

In addition to the original facility location formulation, quadratic assignment problem is a mathematical mode for the problem of placement of interconnected electronic components onto a printed circuit board or on a microchip, which is part of the place and route stage of computer aided design in the electronics industry.

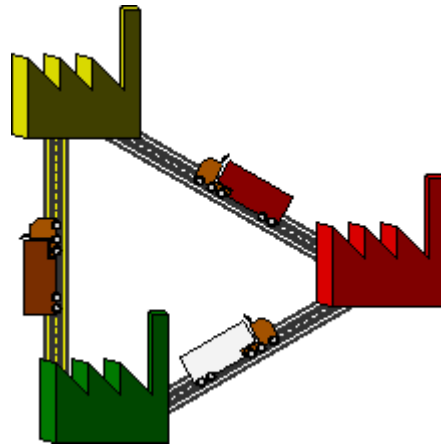


FIGURE 2 QUADRATIC ASSIGNMENT PROBLEM ILLUSTRATION

		Machines		
		M1	M2	M3
Jobs	J1	0	1	6
	J2	0	9	14
	J3	0	0	0

FIGURE 3 LINEAR ASSIGNMENT PROBLEM ILLUSTRATION

#### B. Problem Descriptions

##### B.1 Linear Assignment Problem

A commonly used intuitive introduction to the assignment problem as used, involves the assignment of  $n$  people to  $n$  jobs. For each job assignment, there is a related cost,  $c_{ij}$ , of assigning person  $i$  to job  $j$ . The objective is to assign each person to one and only one job

in such a manner that minimizes the sum of each assignment cost, i.e., the total cost. Mathematically, the problem can be formulated as follows :

$$\min \sum_{i=1}^n C_i \pi(i)$$

### B.2 Quadratic Assignment Problem

Mathematically we can formulate the problem by defining two  $n$  by  $n$  matrices : a flow matrix  $F$  whose  $(i,j)$ -th element represents the flow between facilities  $i$  and  $j$ , and a distance matrix  $D$  whose  $(i,j)$ -th element represents the distance between locations  $I$  and  $j$ . We represents an assignment by the vector  $p$ , which is a permutation of the number  $\{ 1, 2, \dots, n \}$ .  $P(j)$  is the location to which facility  $j$  is assigned. With these definitions, the quadratic assignment problem can be written as :

$$\min_{p \in \Pi} \sum_{i=1}^n \sum_{j=1}^n f_{ij} d_{p(i)p(j)}$$

### C. Other Applications

Before, we have discussed the application of quadratic assignment problem in the use of industrial facility allocation. But there are also other applications in the quadratic assignment problem.

#### C.1 Steinberg Wiring Problem

The Steinberg wiring problem is a major research in the topic of science computer, electrical engineers, and operations research analyst over the past 40 years. The reason why that it become such a big problem is because engineer use this to place the components on the computer backboards. The point of placing it in the computer backboard accordingly is so that the total length of the wiring that was needed will become less so that the computational time will increase. Not only in terms of speeding up the computational time, but it also gives the manufacturer a more cost effective for the backboard.

#### C.2 Hospital Layout

Designing a hospital environment is a formidable task to undertake. Why is it become such a big problem ? This happens because a lot of lives are at stake while designing this hospital. It is very important to design the hospital that it is both beneficial to both the patients and the care providers. For example, we know that the emergency room at nearly every hospital in the world is located at the front of the facility, this happens so that the total distance that a patient needed in urgent to travel is minimized. The idea itself to place emergency room in any other place is self-defeating and illogical.

## IV. IMPLEMENTATION

There are a lot of algorithms that could be implemented in quadratic assignment problem, but in this paper we will only discuss the brute force algorithm. The brute force algorithm will always give us the best result, but considering the computation time, sometimes it is almost impractical to use. The following is the brute force algorithm implementation in the quadratic assignment problem :

### PROCEDURE AND FUNCTION

Array of combination GenerateCombination(input Amount : Integer)

{This function is used to generate every possible combination from the total amount of the object that the problem has}

Path And Cost FindPath(input Combination : Array of combination, DistanceMatrix :  $N \times N$ , FlowMatrix :  $N \times N$ )

{This function is used to compute every combination that has been generated before. After it is computed, it will be compared with each other and the lowest cost will be taken and the combination of the allocation will also be stored.}

### MAIN

#### VARIABLE

AMOUNT : Integer

DISTANCE\_MATRIX :  $N \times N$

FLOW\_MATRIX :  $N \times N$

OPTIMIZED\_RESULT : Path And Cost

COMBINATION : Array of combination

#### ALGORITHM

Combination <= GenerateCombination(AMOUNT)

RESULT\_PATH <= FindPath(COMBINATION, DISTANCE\_MATRIX, FLOW\_MATRIX)

#### A. Variable Explanation

$N \times N$  : A well-defined array that could contain every distance or flow between each facility

Path And Cost : A data structure that contains the cost and the path of the final computation

Array of combination : Array that is used to expressed every possible combination

#### B. Computational Time

Because the algorithm uses brute force, it will generate every possible combination whether it is bad or good. The result from generating such combination will impact the computational time especially if the amount of the allocation rises. We could acquire that the computational time for time is  $O(N!)$ . For example if we have 3

allocations, then the possibilities for the allocation is as following :

- 1.1, 2, 3
- 2.1, 3, 2
- 3.2, 1, 3
- 4.2, 3, 1
- 5.3, 2, 1
- 6.3, 1, 2

From the following we could see that for 3 allocations we would have 6 possibilities :  $3! = 1 \times 2 \times 3 = 6$ .

### C. Implementation Example

In this section, we will use a test case to show how the quadratic assignment problem works. In this test case, we will use 4 machines with 4 allocations. The machine and the locations have the corresponding data :

$$V = \begin{bmatrix} 0 & 2 & 8 & 3 \\ 2 & 0 & 4 & 9 \\ 8 & 4 & 0 & 5 \\ 3 & 9 & 5 & 0 \end{bmatrix}$$

$$D = \begin{bmatrix} 0 & 8 & 10 & 2 \\ 8 & 0 & 4 & 7 \\ 10 & 4 & 0 & 9 \\ 2 & 7 & 9 & 0 \end{bmatrix}$$

The V variable denotes the flow between each facility while the D variable represents the distance between each facility. Now that we have the data, we need to input the data into the program for the program to process.

By using the brute force algorithm described before, we can compute the most optimized cost that we can get from those data. The computation may took some time because of the complexity that it has ( $O(n!) = 4! = 24$ ). A few of the computation is as following :

#### 1. Possibility route : 1-2-3-4

From the route that has been generated we need to calculate the possible cost whether it is the most minimized one or not.

$$\text{Cost} = V_{12} \times D_{12} + V_{13} \times D_{13} + V_{14} \times D_{14} + V_{23} \times D_{23} + V_{24} \times D_{24} + V_{34} \times D_{34} = 2(8) + 8(10) + 3(2) + 4(4) + 9(7) + 5(9) = 226$$

From the computation we acquire that the cost from allocation the facility 1 in location 1, facility 2 in location 2, facility 3 in location 3 and facility 4 in location 4 is 226.

#### 2. Possibility route : 2-4-3-1

From the route that has been generated we need to calculate the possible cost whether it is the most minimized one or not.

$$\text{Cost} = V_{12} \times D_{24} + V_{13} \times D_{23} + V_{14} \times D_{21} + V_{23} \times D_{43} + V_{24} \times D_{41} + V_{34} \times D_{31} = 2(7) + 8(4) + 3(8) + 4(9) + 9(2) + 5(10) = 174$$

From the computation we acquire that the cost from allocation the facility 1 in location 2, facility 2 in location 4, facility 3 in location 3 and facility 4 in location 1 is 174.

#### 3. Possibility route : 3-1-2-4

From the route that has been generated we need to calculate the possible cost whether it is the most minimized one or not.

$$\text{Cost} = V_{12} \times D_{31} + V_{13} \times D_{32} + V_{14} \times D_{34} + V_{23} \times D_{12} + V_{24} \times D_{14} + V_{34} \times D_{24} = 2(10) + 8(4) + 3(9) + 4(8) + 9(2) + 5(7) = 164$$

From the computation we acquire that the cost from allocation the facility 1 in location 3, facility 2 in location 1, facility 3 in location 2 and facility 4 in location 4 is 164.

From the example above could see that different combination of facility allocation will causes the cost of flow to be different from one another. In order to acquire the most optimized result we need scour the entire possibility. When generating the possibility, the larger the amount of machines and locations the larger it will be. Although in this example we actually have 16 possible combinations but I only mentions a few of them and the best result. In conclusion, the most optimized cost from allocating 4 facilities in 4 locations with the data given above is 164.

## V. CONCLUSION

When using the brute force to deal with the quadratic assignment problem, it is only capable of dealing up to 8 locations with 8 facilities which means having 2 8x8 matrices. When we are dealing with problems higher than that, the time that is needed to compute the process is nearly impractical when using brute force. When dealing with problems higher than 8 locations and facilities, we need to use other kind of algorithms such as dynamic programming.

Quadratic assignment problem is NP-hard. Arguably, it is the most difficult NP-hard combinatorial optimization problem. Solving problems of size greater than 30 (i.e. More than 900 0-1 variables) is computationally impractical, among the exact algorithms used to solve quadratic assignment problem, branch and bound has been the most successful. However, the lack of a sharp lower bound is one the major difficulties. Indeed, either the bound is too loose, or the time needed to compute the bound is nearly impossible.

Other ways of solving this quadratic assignment problem is by using heuristic algorithm such as viral systems, construction procedure, improvement procedure.

These algorithms can calculate and gives us the result that we needed but the results may not always get the most optimized result. Usually we always gets the result that is nearest to the optimized result, in other word, by using these algorithms we can have a better processing time complexity but the cost of having a better time processing will be paid by having not the not the most optimized result but the result that is closest to it.

## VII. ACKNOWLEDGMENT

I would like to thanks Suwandi for helping in creating this paper. This paper would not have been finished without his help.

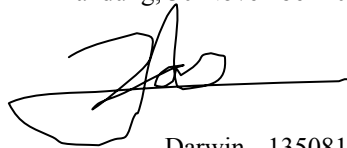
## REFERENCES

- [1] <http://www.cs.berkeley.edu/~cjr/GSI/cs267-s04/homework-0/results/sonesh/> accessed 27 November 2010
- [2] [http://en.wikipedia.org/wiki/Quadratic\\_assignment\\_problem](http://en.wikipedia.org/wiki/Quadratic_assignment_problem) accessed 27 November 2010
- [3] <http://plaza.ufl.edu/clayton8/article.pdf> accessed 27 November 2010
- [4] Munir, Rinaldi, "Diktat Kuliah IF2251 Strategi Algoritmik", Program Studi Teknik Informatika, ITB, 2007.
- [5] <http://www.ludd.luth.se/~proximus/ltu/pts4qap.pdf> accessed 27 November 2010
- [6] [http://en.wikipedia.org/wiki/Linear\\_assignment\\_problem](http://en.wikipedia.org/wiki/Linear_assignment_problem) accessed 27 November 2010
- [7] [http://www.transtutors.com/Uploadfile/CMS\\_Images/13408\\_Cheek%20for%20optimality.JPG](http://www.transtutors.com/Uploadfile/CMS_Images/13408_Cheek%20for%20optimality.JPG) accessed 27 November 2010
- [8] <http://reactive-search.com/images/qap4dummies.png> accessed 27 November 2010
- [9] Tompkins, J.A., White, A.W., Bozer, Y.A., Frazelle, E.H., Tanchoco, J.M.A., dan Trevino, T. 1996, *Facility Planning 2nd ed.*, John Wiley & Sons, New York.

## PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 30 November 2010



Darwin - 13508102