

|  |  |  |
| --- | --- | --- |
| **Kingdom of Saudi Arabia**  **Ministry of Education**  **Islamic University of Madinah**  **Faculty of Computer and**  **Information Systems** | **IU_ALL_COLOR_RGB** | **المملكة العربيـة السـعودية**  **وزارة التعليم**  **الجامعة الإسلامية بالمدينة المنورة**  **كلية الحاسب الالي ونظم المعلومات** |

**CCS 3013/ Design and Analysis of Algorithms**

**Assignment problem**

**2019-2020**

**Semester 2**

**Prepare by: Subject's instructor:**

Saif Alshamrani 361003663 Dr. Hatem El-Boghdadi

Mohmmad Jamal 362049583 Mr. Majed Alhariry

1. INTRODUCTION
   1. PROBLEM STATEMENT

Nowadays getting a job became something difficult to have which you have always dreamed about it since the first day for you at elementary school.

These difficulties are due to several factors, perhaps the most important of which is that a person may obtain two jobs at the same time if he has a high qualification.

This project has been taken four people as a sample from random population and focus on hiring them into four available jobs by using Hungarian algorithm.

Table1.1 (The number inside the table refers to the job's complexity).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Job 1 | Job 2 | Job 3 | Job 4 |
| Person 1 | 10 | 9 | 8 | 7 |
| Person 2 | 4 | 3 | 2 | 2 |
| Person 3 | 40 | 3 | 2 | 1 |
| Person 4 | 20 | 19 | 18 | 17 |

* 1. OBJECTIVES OR GOALS

The main goal of this project is to provide the right job for the right person which means the smallest complexity that each person has is the right job for him/her.

In order to ensure that we hiring one person per a job we will apply Hungarian method to solve the problem by finding the minimum total cost and come up with a modification of the Hungarian method that run faster (less complexity) than the original version and can handle the described situation.

1. IMPLEMENTATION
   1. INTRODUCTION

This chapter discusses the implementation of Hungarian algorithm. Firstly, we will mention the general information about Hungarian algorithm in section 2.2, then we will come up with the first version of our solution and that will be in section 2.3, after that we will try as much as we can to optimize the solution in section 2.4.

* 1. ABOUT HUNGARIAN ALGORITHM

Hungarian algorithm is a combinatorial optimization algorithm that gives us an efficient method to find the optimal solution without having to make a direct comparison of every solution. It was developed and published in 1955 by Harold Kuhn.

An assignment problem is a particular case of transportation problem where it can solve be Hungarian algorithm which goal is to minimize the cost or maximize it as well.

In order to implementing the Hungarian method, you must go through four steps:

* **Step One:** subtract the smallest entry in a row to all entry row.
* **Step Two:** subtract the smallest entry in a column to all entry column.
* **Step three:** draw lines through the row and columns to cover all zeros.
* **Step four:** find the smallest entry that is not covered by any line and subtract it to all other elements that is not covered as well. Then, add the same entry to the intersection of the row in the column, if any.

Step one and two is going to be once, while three and four will be repeated until it optimizes the problem. In more details the section 2.3 will cover that with our problem.

The time complexity of the original algorithm is O (). And later on, Edmonds and Karp came up with a studied in 1972 to improve this complexity and they achieve to O ().

* 1. ORIGINAL VERSION OF SOLUTION

In order to find the first version of the problem's solution we have to preformed four steps to assign these four people into only one job for each.

Let's begin with the **first step** which is subtracting the smallest entry in row from all the other entries in the row and repeating this process to all rows that you have. This will give you at least one zero per row.

Table 2.1 (The table before preforming the first step).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Job 1 | Job 2 | Job 3 | Job 4 |
| Person 1 | 10 | 9 | 8 | 7 |
| Person 2 | 4 | 3 | 2 | 2 |
| Person 3 | 40 | 3 | 2 | 1 |
| Person 4 | 20 | 19 | 18 | 17 |

Table 2.2 (The table after preforming the first step).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Job 1 | Job 2 | Job 3 | Job 4 |
| Person 1 | 3 | 2 | 1 | 0 |
| Person 2 | 2 | 1 | 0 | 0 |
| Person 3 | 39 | 2 | 1 | 0 |
| Person 4 | 3 | 2 | 1 | 0 |

Then, after we have done from the first step, we perform the **second step** which is as same as we have done earlier in first step but instant of subtracting the smallest entry in each row, we will subtract it form the columns now.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Job 1 | Job 2 | Job 3 | Job 4 |
| Person 1 | 1 | 1 | 1 | 0 |
| Person 2 | 0 | 0 | 0 | 0 |
| Person 3 | 37 | 1 | 1 | 0 |
| Person 4 | 1 | 1 | 1 | 0 |

Table 2.3 (The table after preforming the second step).

Now the **third step** is to draw lines through the rows and columns that have the 0 entries such that the fewest lines possible are drawn.

Table 2.4 (The table after preforming the third step).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Job 1 | Job 2 | Job 3 | Job 4 |
| Person 1 | 1 | 1 | 1 | 0 |
| Person 2 | 0 | 0 | 0 | 0 |
| Person 3 | 37 | 1 | 1 | 0 |
| Person 4 | 1 | 1 | 1 | 0 |

As we can notes here, we got two lines which is the fewest possible lines but the problem is that we have four people and four jobs but we got only two lines which means we only assign two person into two job so we have tocheck the number of lines must be not less than the number of person and job. Since the optimal number of zeros is not yet reached, we will continue optimizations and moves to the **fourth step** which is to find the smallest entry that is not covered by any line and subtract it from each row that is not crossed out, and then add it to where the row intersects the column. Then, go back to **third step.**

Table 2.5 (The table after preforming the fourth & fifth step).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Job 1 | Job 2 | Job 3 | Job 4 |
| Person 1 | 0 | 0 | 0 | 0 |
| Person 2 | 0 | 0 | 0 | 1 |
| Person 3 | 36 | 0 | 0 | 0 |
| Person 4 | 0 | 0 | 0 | 0 |

After we make the lines equal the number of people, we will assign each person into one zero because one person has more than zero, provided that this place (row and column) is reserved and can not be used anymore. Acutely, we can perform many solutions with the same cost. Table 2.6 below is also one of the solutions.

Table 2.6 (another solution by applying the fourth & fifth step).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Job 1 | Job 2 | Job 3 | Job 4 |
| Person 1 | 0 | 0 | 0 | 0 |
| Person 2 | 0 | 0 | 0 | 1 |
| Person 3 | 36 | 0 | 0 | 0 |
| Person 4 | 0 | 0 | 0 | 0 |

By drawing this line, we will ensure that a person has been assign to only one job and that job can be only for that person. Now let's take a look how much the cost will this line be in original table 2.1.

Table 2.5 (The table after preforming the fourth & fifth step).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Job 1 | Job 2 | Job 3 | Job 4 |
| Person 1 | 10 | 9 | 8 | 7 |
| Person 2 | 4 | 3 | 2 | 2 |
| Person 3 | 40 | 3 | 2 | 1 |
| Person 4 | 20 | 19 | 18 | 17 |

The optimal value (cost) equals = 10 + 3 + 2 + 17 = 32.

In fact, after we have analyzed we already found the order of growth is () according to the basic operation which is comparisons because the comparisons are bigger than the subtraction operation in term of complexity. Then, we look at the table and take the first row and compare each element in this row to find the minimum value and repeat this situation in all rows which means to be the order of growth is [] . Moreover, we take the first column and compare each element in this column to find the minimum value which means also to be the order of growth is [(] . In the worst case after all comparison and subtract the minimum value for all rows and columns it will produce at least one line of zeros. So, we must to find the rest of lines () by comparisons between all the remaining element which means to be the order of growth is [ (( ) × () ]

* 1. OPTIMIZATION OF THE SOLUTION

According to the previous solution by using the concept of Hungarian method we note that the elements is assigned by decreasing order in the rows which makes it a special case and it could help us to optimize our solution because we don't have to compare the elements in the rows because we always find the smallest element for each row in the last column. We will take the last element in each row and subtract it from each row. Thus, the last column will be zeros. Then, we take the first column and compare each element in this column to find the minimum value except the last column because it is already will be consisting from zeroes which mean to be the order of growth is [( ] . In the worst case after all comparison and subtract the minimum value for all rows and columns it will produce at least one line of zeros. So, we must to find the rest of lines (). We do not need to compare between the remaining elements because the smallest elements will be in columns where is located before the last column and repeat this situation to all columns which means to be the order of growth is [ ) ] and this is the optimization.

1. Reference:
2. Mansi, Angela (2019) Personality at work: an empirical investigation into higher order factors underpinning the socioanalytic theory of personality. Doctoral thesis, Birkbeck, University of London.
3. <https://www.hungarianalgorithm.com/>
4. <https://shodhganga.inflibnet.ac.in/bitstream/10603/3970/10/10_chapter%205.pdf>