Trevor Morton

2/21/2017

Project 3

MA 444

The Hungarian Method

For this problem, I chose to work with the Hungarian Method. I made this decision because the problem the method solves, the Assignment Problem, is interesting from a computer science standpoint as it deals with discrete binary assignment of certain costs to certain actions. This makes it an interesting problem to understand from a mathematical perspective, and makes it useful to know how to efficiently solve it as it comes up often and knowing a truly efficient Method to handle it gives me an advantage over those who might approach it from the brute force side of things.

To start with I will describe the assignment problem in plain English. Let's say we have n people who work for a given company and m different jobs that the company needs completed. Each of these n people can complete each of the m jobs in a certain amount of time that is different for each person, and each job they can perform. The problem is then, how can you assign your employees jobs to complete and have the total amount of time it takes to complete the jobs be minimized? From a mathematical standpoint, this problem can be described as:

$$Min: \sum_{i=1}^{n}\sum_{j=1}^{m} c_{i,j} * x_{i,j}$$

$$S.T.$$

$$\sum_{i=1}^{n} x_{i,j} = 1 \; For\ all\ j = \{1 \dots m\}$$

$$\sum_{j=1}^{m} x_{i,j} = 1 \; For\ all\ i = \{1 \dots n\}$$

$$x_{i,j} = \{0,1\} \; For\ all\ i = \{1 \dots n\}, j = \{1 \dots m\}$$

The question then becomes, how does one solve this type of problem? There are many ways to consider this type of problem, from a simple linear program to a network flow problem but for the case of the Hungarian Method we shall treat it as a linear program. As with all linear problems, this problem has a dual to it, which will be incredibly useful in solving this problem. The dual to the problem is stated below.

$$Max\colon \sum_{i=1}^{n} u_i + \sum_{j=1}^{m} v_j$$

$$S.T.$$

$$u_i + v_j \leq c_{i,j} \; for\ all\ i = \{1 \dots n\}, j = \{1 \dots m\}$$

Just by looking at it, the dual is clearly easier to solve, as instead of dealing with $n * m$ variables, we deal only with $m + n$, which even if we were to solve it from a brute force standpoint is much simpler. This problem is also simpler because in all cases of the dual, the solution of $u = \vec{0}$ and $v = \vec{0}$ is always feasible and so we always have a known starting point.

From what we already know about solving linear programs using the Simplex Method, it would appear that this would be a good case to use the Primal-Dual Simplex Method, as the dual has a trivial feasible solution and it is simple to keep complementary slackness satisfied. This is actually very similar to what we are about to do in the Hungarian Method, as the Method starts at this dual-feasible solution and keeps progressing until we have a primal feasible solution as well. Delving more into the actual history of the Primal-Dual Simplex Method this similarity turns out to not be a coincidence, as the Primal-Dual Simplex Method was actually developed to be a general form of the Hungarian Method. This makes it clear that the similarities in approach to the problem are to be expected, as the Hungarian Method is simply a specific application of the Primal-Dual Simplex Method.

Setting up our assignment problem for use with the Hungarian Method, there are several things we must establish. The first is that our variables $c_{i,j}$ can be represented in the form of a $n * m$ matrix, where the matrix value at index i, j is $c_{i,j}$. This matrix will be one of the main entities we deal with throughout this problem. We also must establish the idea of a forbidden cell in this matrix. This would be associated with a cost of infinity, or an assignment that is not allowed in our problem. Another thing we must establish is the idea of assignment. When you assign a variable $c_{i,j}$, you are saying that its corresponding variable $x_{i,j}$ is being temporarily set to 1 for the purposes of checking for feasibility in the primal. The last thing is the idea of labeling, which for the purposes of visualization can be related to crossing out a certain cell of our matrix if it is labeled, or simply marking it in some way.

The Hungarian Method, stated plainly, is as follows:

1. If the entire matrix has a cost of infinity, or the entire problem is forbidden then stop as there is no feasible solution. Otherwise continue.
2. Create the initial feasible solution to the dual. This is done by creating two vectors $\vec{u}$ and $\vec{v}$. First set $\overrightarrow{u_i}$ to the min $c_{i,j}$ in each row of the cost matrix. Then subtract $\overrightarrow{u_i}$ from $c_{i,j}$ where the i's are equal. Next set $\overrightarrow{v_j}$ to the min of $c_{i,j}$ for each column of the cost matrix. Then subtract $\overrightarrow{v_j}$ from $c_{i,j}$ where the j's are equal. These two vectors $\vec{u}$ and $\vec{v}$ are the initial feasible solution to the dual.
3. The next step is to create a partial assignment for the problem, which will take place in two steps:

   a. First we will move across all the rows. For each row i, if there is one and only one $c_{i,j}$ equal to zero and unlabeled, label the column that the $c_{i,j}$ is in and mark that $c_{i,j}$ as assigned.
   b. Next, we will move across all the columns. For each column j, if there is one and only one $c_{i,j}$ equal to zero and unlabeled, label the row that $c_{i,j}$ is in and mark that $c_{i,j}$ as assigned.
4. If there is a $c_{i,j}$ equal to zero and still unlabeled, go back to step 3 leaving the current labels in place.
5. Now that we have our partial assignment, we can check if we are at an optimal solution. If the number of assigned $c_{i,j}$ is equal to n then we are at an optimal solution. If not, move on to the next step.
6. We will now find a δ equal to the minimum of the $c_{i,j}$'s that are unlabeled. We will then subtract that δ from each of those unlabeled $c_{i,j}$'s and add our δ to each $c_{i,j}$ that has been labeled twice. If δ is infinity, then the problem is infeasible and we stop. With our new matrix of $c_{i,j}$'s, go back to step 3.

Now to go into explanation over how the algorithm works step by step.

The first step is fairly self explanatory. When every single cost is infinity, it means that we had an assignment problem where we forbid every assignment possible. Although this seems like a silly case to include, if we were generating these problems in a computer algorithm there is a chance we could generate a problem where every assignment was forbidden, and so we must account for it.

The second step deals with creating an initial feasible solution to the dual. This is where we really start seeing how we use duality in this problem. Even though we are dealing with this matrix of $c_{i,j}$'s from the primal and apparently changing those values in some arbitrary way, we are in fact finding feasible solutions to the dual. By finding the minimum value in a row or column and subtracting that value from the entire row or column, we are really just using this as a bookkeeping method to ensure that our dual solution does not break the condition

$$u_i + v_j \leq c_{i,j} \; for \; all \; i = \{1 \dots n\}, j = \{1 \dots m\}$$

Although as we go through the algorithm we change the $c_{i,j}$'s that we deal with, the true $c_{i,j}$'s in the problem never change and this second step of the algorithm is just finding a feasible solution to the dual that is simple to find and better than the trivial solution where both vectors are equal to zero.

The third step involves creating a partial assignment and going through what is known as labeling. By going through each row to find a $c_{i,j}$ equal to zero with no other unlabeled $c_{i,j}$'s in the row basically checks to see if we can currently assign this variable to true as there is no extra cost associated with it, and if we can assign it to true we label the column that it was in to make sure that no other zero gets assigned in that column to maintain the constraints of the primal. The same thing but with the rows and columns swapped applies to going through the columns.

The fourth step is rather straightforward. If there is still an unmarked $c_{i,j}$ equal to zero when you come to step four it means you hit an edge case when assigning your variables, so you must go back to step 3 to continue with the assignment.

The fifth step take care of checking whether we are at the optimal solution. In this step we check if our primal is feasible, and if it is we call it optimal. If it is feasible then it must be optimal because throughout our algorithm we are maintaining the feasibility of the dual and maintaining complementary slackness. This means if the primal is feasible then all three of those properties are true at the same time, which only occurs when we are at our optimal solution.

The sixth step is where we improve our dual solution, and where the most subtle use of the dual comes in. In this step we take the minimum of the unlabeled $c_{i,j}$'s and subtract that from each unlabeled $c_{i,j}$ and add it to each doubly labeled $c_{i,j}$. What this is really doing is finding all the $c_{i,j}$'s whose corresponding $\overrightarrow{u_i}$ and $\overrightarrow{v_j}$ can be further increased. By subtracting this minimum value from the unlabeled $c_{i,j}$ we are really increasing the $\overrightarrow{u_i}$ or $\overrightarrow{v_j}$ corresponding to it in order to improve our dual. The reason we must add to the $c_{i,j}$'s that are double labeled is because when we increase the values of some components of $\overrightarrow{u_i}$ or $\overrightarrow{v_j}$, the rows or columns they correspond to contain labeled $c_{i,j}$'s and so if you increased $\overrightarrow{u_i}$, you must decrease $\overrightarrow{v_j}$ in one of its components in order to make sure the labeled $c_{i,j}$'s are not modified. Going through this process always results in an improved dual value because there will always be more components being increased than being decreased and so the objective value of the dual will always be increasing.

As briefly discussed before, the Hungarian Method is very closely related to the Primal-Dual Simplex Method. This is because the Primal-Dual Simplex Method was actually created to take advantage of duality in general case linear programs in the same way that the Hungarian Method takes advantage of duality in the assignment problem. Both algorithms have numerous similarities between them, and step by step the Primal-Dual Simplex Method can be seen as just approaching linear programs from the most mathematically abstracted approach to the same idea of what the Hungarian Method does. To start with, they both start at a dual feasible solution instead of a primal dual solution taking advantage of the fact that it is often simpler to find these solutions. They also both maintain dual feasibility and complementary slackness in order to simply check for dual feasibility when checking if a solution is optimal. They both move towards optimizing the dual caring not about how feasible the primal is and just waiting for the primal to be feasible. They also both go about exposing the problem to a new way to interpret it, and if you did not know the dual you would have no idea that you could even go about solving it in this fashion.

Now that we know how the algorithm works and how to run through it, let's go through a couple examples.

Example 1:

$$c_{i,j}= \begin{bmatrix} 37.7 & 32.9 & 33.8 & 37.0 & 35.4 \\ 43.4 & 33.1 & 42.2 & 34.7 & 41.8 \\ 33.3 & 28.5 & 38.9 & 30.4 & 33.6 \\ 29.2 & 26.4 & 29.6 & 28.5 & 31.1 \end{bmatrix}$$

Step 1: There are no forbidden cells in this cost matrix, so we continue.

Step 2: Taking the minimum of the rows and columns of the matrix, we find the $\vec{u}$ of our initial feasible solution to the dual to be

$$\vec{u} = \begin{bmatrix} 32.9 \\ 33.1 \\ 28.5 \\ 26.4 \end{bmatrix}$$

The cost matrix is now

$$c_{i,j}= \begin{bmatrix} 4.8 & 0 & .9 & 4.1 & 2.5 \\ 10.3 & 0 & 9.1 & 1.6 & 8.7 \\ 4.8 & 0 & 10.4 & 1.9 & 5.1 \\ 2.8 & 0 & 3.2 & 2.1 & 4.7 \end{bmatrix}$$

We can now find the $\vec{v}$ of our initial feasible solution to the dual to be

$$\vec{v} = \begin{bmatrix} 2.8 \\ 0 \\ .9 \\ 1.6 \\ 2.5 \end{bmatrix}$$

The cost matrix is now

$$c_{i,j}=\begin{bmatrix} 2 & 0 & 0 & 2.5 & 0 \\ 7.5 & 0 & 8.2 & 0 & 6.2 \\ 2 & 0 & 9.5 & .3 & 2.6 \\ 0 & 0 & 2.3 & .5 & 2.2 \end{bmatrix}$$

Step 3: We will now create the partial assignment.

Part 1: Going through the rows and assigning based on the rules in the algorithm, our matrix looks like this

$$c_{i,j}=\begin{bmatrix} 2 & \cancel{0} & 0 & 2.5 & 0 \\ 7.5 & \cancel{0} & 8.2 & 0 & 6.2 \\ 2 & \cancel{\mathbf{0}} & 9.5 & .3 & 2.6 \\ 0 & \cancel{0} & 2.3 & .5 & 2.2 \end{bmatrix}$$

Where the bold numbers represent the assigned $c_{i,j}$'s and the numbers that are crossed out represent the labeled $c_{i,j}$'s.

Part 2: Going through the columns and assigning based on the rules in our algorithm, our matrix looks like this

$$c_{i,j}=\begin{bmatrix} \cancel{2} & \cancel{\cancel{0}} & \cancel{\mathbf{0}} & \cancel{2.5} & \cancel{0} \\ \cancel{7.5} & \cancel{\cancel{0}} & \cancel{8.2} & \cancel{\mathbf{0}} & \cancel{6.2} \\ 2 & \cancel{\mathbf{0}} & 9.5 & .3 & 2.6 \\ \cancel{\mathbf{0}} & \cancel{\cancel{0}} & \cancel{2.3} & \cancel{.5} & \cancel{2.2} \end{bmatrix}$$

With the same formatting rules as above, but where each number that is crossed out twice is double labeled.

Step 4: There are no $c_{i,j}$'s equal to zero that are unlabeled, so we can move on.

Step 5: There are only four assigned $c_{i,j}$'s and we want there to be four, so we are at an optimal solution and may stop.

Looking at our original $c_{i,j}$ matrix we find that the optimal cost is

$$29.2 + 28.5 + 33.8 + 34.7 = 126.2$$

We can affirm this by looking at the value of the sum of the components of the current dual feasible solution

$$\sum_{i=1}^{n} u_i + \sum_{j=1}^{m} v_j = (32.9 + 33.1 + 28.5 + 26.4) + (2.8 + 0 + .9 + 1.6 + 2.5) = 128.7$$

These values are not the same, so it might appear at first as though our algorithm has failed. It hasn't though, because the optimal cost we found in the primal is for if our matrix was a 4x4 and does not take into account the fact that we have an extra column in the matrix that is not being assigned to. We need to fix our dual vector to reflect the answer the primal is giving us, and subtract the component of the dual feasible solution that is associated with the column that was not assigned. Doing that we would get

$$\sum_{i=1}^{n} u_i + \sum_{j=1}^{m} v_j = (32.9 + 33.1 + 28.5 + 26.4) + (2.8 + 0 + .9 + 1.6 + 2.5) - 2.5 = 126.2$$

The optimal solution of the dual and the primal are now identical, and it has been confirmed that this assignment is optimal.

Example 2:

$$c_{i,j=} \begin{bmatrix} 10 & 11 & 10 & 5 & 6 & 4 & 3 \\ 5 & 26 & 14 & 18 & 15 & 10 & 10 \\ 6 & 22 & 18 & 17 & 15 & 8 & 8 \\ 2 & 14 & 16 & 16 & 24 & 25 & 12 \\ 4 & 15 & 19 & 10 & 8 & 14 & 11 \\ 10 & 22 & 22 & 15 & 28 & 24 & 12 \\ 8 & 18 & 21 & 18 & 18 & 18 & 14 \end{bmatrix}$$

Step 1: There are no forbidden cells in this cost matrix, so we continue.

Step 2: Taking the minimum of the rows and columns of the matrix, we find the $\vec{u}$ of our initial feasible solution to the dual to be

$$\vec{u} = \begin{bmatrix} 3 \\ 5 \\ 6 \\ 2 \\ 4 \\ 10 \\ 8 \end{bmatrix}$$

The cost matrix is now

$$c_{i,j=} \begin{bmatrix} 7 & 8 & 7 & 2 & 3 & 1 & 0 \\ 0 & 21 & 9 & 13 & 10 & 5 & 5 \\ 0 & 16 & 12 & 11 & 9 & 2 & 2 \\ 0 & 12 & 14 & 14 & 22 & 23 & 10 \\ 0 & 11 & 15 & 6 & 4 & 10 & 7 \\ 0 & 12 & 12 & 5 & 18 & 14 & 2 \\ 0 & 10 & 13 & 10 & 10 & 10 & 6 \end{bmatrix}$$

We can now find the $\vec{v}$ of our initial feasible solution to the dual to be

$$\vec{v} = \begin{bmatrix} 0 \\ 8 \\ 7 \\ 2 \\ 3 \\ 1 \\ 0 \end{bmatrix}$$

The cost matrix is now

$$c_{i,j=} \begin{bmatrix} 7 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 13 & 2 & 11 & 7 & 4 & 5 \\ 0 & 8 & 5 & 9 & 6 & 1 & 2 \\ 0 & 4 & 7 & 12 & 19 & 22 & 10 \\ 0 & 3 & 8 & 4 & 1 & 19 & 7 \\ 0 & 4 & 5 & 3 & 15 & 13 & 2 \\ 0 & 2 & 6 & 8 & 7 & 9 & 6 \end{bmatrix}$$

Step 3: We will now create the partial assignment.

Part 1: Going through the rows and assigning based on the rules in the algorithm, our matrix looks like this

$$c_{i,j}=\begin{bmatrix} \cancel{7} & 0 & 0 & 0 & 0 & 0 & 0 \\ \cancel{\mathbf{0}} & 13 & 2 & 11 & 7 & 4 & 5 \\ \cancel{0} & 8 & 5 & 9 & 6 & 1 & 2 \\ \cancel{0} & 4 & 7 & 12 & 19 & 22 & 10 \\ \cancel{0} & 3 & 8 & 4 & 1 & 19 & 7 \\ \cancel{0} & 4 & 5 & 3 & 15 & 13 & 2 \\ \cancel{0} & 2 & 6 & 8 & 7 & 9 & 6 \end{bmatrix}$$

Where the bold numbers represent the assigned $c_{i,j}$'s and the numbers that are crossed out represent the labeled $c_{i,j}$'s.

Part 2: Going through the columns and assigning based on the rules in our algorithm, our matrix looks like this

$$c_{i,j}=\begin{bmatrix} \cancel{7} & \cancel{\mathbf{0}} & \cancel{0} & \cancel{0} & \cancel{0} & \cancel{0} & \cancel{0} \\ \cancel{\mathbf{0}} & 13 & 2 & 11 & 7 & 4 & 5 \\ \cancel{0} & 8 & 5 & 9 & 6 & 1 & 2 \\ \cancel{0} & 4 & 7 & 12 & 19 & 22 & 10 \\ \cancel{0} & 3 & 8 & 4 & 1 & 19 & 7 \\ \cancel{0} & 4 & 5 & 3 & 15 & 13 & 2 \\ \cancel{0} & 2 & 6 & 8 & 7 & 9 & 6 \end{bmatrix}$$

With the same formatting rules as above, but where each number that is crossed out twice is double labeled.

Step 4: There are no $c_{i,j}$'s equal to zero that are unlabeled, so we can move on.

Step 5: There are only two assigned $c_{i,j}$'s and we want there to be seven, so we are not at an optimal solution. We move to the next step.

Step 6: Going through all the unlabeled $c_{i,j}$'s we find our δ to be 1. Subtracting that delta from the unmarked $c_{i,j}$'s and adding it to the doubly marked $c_{i,j}$'s we get the following cost matrix

$$c_{i,j=}\begin{bmatrix} 8 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 12 & 1 & 10 & 6 & 3 & 4 \\ 0 & 7 & 4 & 8 & 5 & 0 & 1 \\ 0 & 3 & 6 & 11 & 18 & 21 & 9 \\ 0 & 2 & 7 & 3 & 0 & 18 & 6 \\ 0 & 3 & 4 & 2 & 14 & 12 & 1 \\ 0 & 1 & 5 & 7 & 6 & 8 & 5 \end{bmatrix}$$

Although our algorithm does not ask us to do this, to better understand what is happening I will show what has happened to our dual solution at this point. The $\vec{v}$ components corresponding to the last 6 columns of the matrix will increase by one each which results in the reduction of all of the unlabeled $c_{i,j}$'s by one. In order to make sure the top row isn't increased though, we must decrease the $\vec{u}$ component corresponding to the first row of our matrix. This means our new feasible solution to the dual is

$$\vec{u} = \begin{bmatrix} 2 \\ 5 \\ 6 \\ 2 \\ 4 \\ 10 \\ 8 \end{bmatrix}, \qquad \vec{v} = \begin{bmatrix} 0 \\ 9 \\ 8 \\ 3 \\ 4 \\ 2 \\ 1 \end{bmatrix}$$

This step is now over, so we will go back to step 3.

Step 3: We will now create the partial assignment.

Part 1: Going through the rows and assigning based on the rules in the algorithm, our matrix looks like this

$$c_{i,j=}\begin{bmatrix} \cancel{8} & 0 & 0 & 0 & \cancel{0} & \cancel{0} & 0 \\ \cancel{\mathbf{0}} & 12 & 1 & 10 & \cancel{6} & \cancel{3} & 4 \\ \cancel{0} & 7 & 4 & 8 & \cancel{5} & \cancel{\mathbf{0}} & 1 \\ \cancel{0} & 3 & 6 & 11 & \cancel{18} & \cancel{21} & 9 \\ \cancel{0} & 2 & 7 & 3 & \cancel{\mathbf{0}} & \cancel{18} & 6 \\ \cancel{0} & 3 & 4 & 2 & \cancel{14} & \cancel{12} & 1 \\ \cancel{0} & 1 & 5 & 7 & \cancel{6} & \cancel{8} & 5 \end{bmatrix}$$

Part 2: Going through the columns and assigning based on the rules in our algorithm, our matrix looks like this

$$c_{i,j}=\begin{bmatrix} \xcancel{8} & \cancel{\mathbf{0}} & \cancel{0} & \cancel{0} & \xcancel{0} & \xcancel{0} & \cancel{0} \\ \cancel{\mathbf{0}} & 12 & 1 & 10 & \cancel{6} & \cancel{3} & 4 \\ \cancel{0} & 7 & 4 & 8 & \cancel{5} & \cancel{\mathbf{0}} & 1 \\ \cancel{0} & 3 & 6 & 11 & \cancel{18} & \cancel{21} & 9 \\ \cancel{0} & 2 & 7 & 3 & \cancel{\mathbf{0}} & \cancel{18} & 6 \\ \cancel{0} & 3 & 4 & 2 & \cancel{14} & \cancel{12} & 1 \\ \cancel{0} & 1 & 5 & 7 & \cancel{6} & \cancel{8} & 5 \end{bmatrix}$$

Step 4: There are no $c_{i,j}$'s equal to zero that are unlabeled, so we can move on.

Step 5: There are only four assigned $c_{i,j}$'s and we want there to be seven, so we are not at an optimal solution. We move to the next step.

Step 6: Going through all the unlabeled $c_{i,j}$'s we find our δ to be 1. Subtracting that delta from the unmarked $c_{i,j}$'s and adding it to the doubly marked $c_{i,j}$'s we get the following cost matrix

$$c_{i,j}=\begin{bmatrix} 9 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 11 & 0 & 9 & 6 & 3 & 3 \\ 0 & 6 & 3 & 7 & 5 & 0 & 0 \\ 0 & 2 & 5 & 10 & 18 & 21 & 8 \\ 0 & 1 & 6 & 2 & 0 & 18 & 5 \\ 0 & 2 & 3 & 1 & 14 & 12 & 0 \\ 0 & 0 & 4 & 6 & 6 & 8 & 4 \end{bmatrix}$$

From this step, we can find that the new feasible solution to the dual is

$$\vec{u}=\begin{bmatrix} 1 \\ 5 \\ 6 \\ 2 \\ 4 \\ 10 \\ 8 \end{bmatrix}, \qquad \vec{v}=\begin{bmatrix} 0 \\ 10 \\ 9 \\ 4 \\ 4 \\ 2 \\ 2 \end{bmatrix}$$

We now move back to step 3

Step 3: We will now create the partial assignment.

Part 1: Going through the rows and assigning based on the rules in the algorithm, our matrix looks like this

$$c_{i,j}= \begin{bmatrix} \cancel{9} & \cancel{0} & 0 & 0 & \cancel{1} & 1 & \cancel{0} \\ \cancel{0} & \cancel{11} & 0 & 9 & \cancel{6} & 3 & \cancel{3} \\ \cancel{0} & \cancel{6} & 3 & 7 & \cancel{5} & 0 & \cancel{0} \\ \cancel{\mathbf{0}} & \cancel{2} & 5 & 10 & \cancel{18} & 21 & \cancel{8} \\ \cancel{0} & \cancel{1} & 6 & 2 & \cancel{\mathbf{0}} & 18 & \cancel{5} \\ \cancel{0} & \cancel{2} & 3 & 1 & \cancel{14} & 12 & \cancel{\mathbf{0}} \\ \cancel{0} & \cancel{\mathbf{0}} & 4 & 6 & \cancel{6} & 8 & \cancel{4} \end{bmatrix}$$

Part 2: Going through the columns and assigning based on the rules in the algorithm, our matrix looks like this

$$c_{i,j}= \begin{bmatrix} \cancel{9} & \cancel{0} & \cancel{0} & \cancel{\mathbf{0}} & \cancel{1} & \cancel{1} & \cancel{0} \\ \cancel{0} & \cancel{11} & 0 & 9 & \cancel{6} & 3 & \cancel{3} \\ \cancel{0} & \cancel{6} & \cancel{3} & \cancel{7} & \cancel{5} & \cancel{\mathbf{0}} & \cancel{0} \\ \cancel{\mathbf{0}} & \cancel{2} & 5 & 10 & \cancel{18} & 21 & \cancel{8} \\ \cancel{0} & \cancel{1} & 6 & 2 & \cancel{\mathbf{0}} & 18 & \cancel{5} \\ \cancel{0} & \cancel{2} & 3 & 1 & \cancel{14} & 12 & \cancel{\mathbf{0}} \\ \cancel{0} & \cancel{\mathbf{0}} & 4 & 6 & \cancel{6} & 8 & \cancel{4} \end{bmatrix}$$

Step 4: There are $c_{i,j}$'s that are unmarked and equal to zero, so we go back to Step 3

Step 3: We will now create the partial assignment.

Part 1: Going through the rows and assigning based on the rules in the algorithm, our matrix looks like this

$$c_{i,j}= \begin{bmatrix} \cancel{9} & \cancel{0} & \cancel{0} & \cancel{\mathbf{0}} & \cancel{1} & \cancel{1} & \cancel{0} \\ \cancel{0} & \cancel{11} & \cancel{\mathbf{0}} & 9 & \cancel{6} & 3 & \cancel{3} \\ \cancel{0} & \cancel{6} & \cancel{3} & \cancel{7} & \cancel{5} & \cancel{\mathbf{0}} & \cancel{0} \\ \cancel{\mathbf{0}} & \cancel{2} & \cancel{5} & 10 & \cancel{18} & 21 & \cancel{8} \\ \cancel{0} & \cancel{1} & \cancel{6} & 2 & \cancel{\mathbf{0}} & 18 & \cancel{5} \\ \cancel{0} & \cancel{2} & \cancel{3} & 1 & \cancel{14} & 12 & \cancel{\mathbf{0}} \\ \cancel{0} & \cancel{\mathbf{0}} & 4 & 6 & \cancel{6} & 8 & \cancel{4} \end{bmatrix}$$

Part 2: Going through the columns and assigning based on the rules in the algorithm, our matrix looks like this

$$c_{i,j}= \begin{bmatrix} \cancel{9} & \cancel{0} & \cancel{0} & \cancel{\mathbf{0}} & \cancel{1} & \cancel{1} & \cancel{0} \\ \cancel{0} & \cancel{11} & \cancel{\mathbf{0}} & 9 & \cancel{6} & 3 & \cancel{3} \\ \cancel{0} & \cancel{6} & \cancel{3} & \cancel{7} & \cancel{5} & \cancel{\mathbf{0}} & \cancel{0} \\ \cancel{\mathbf{0}} & \cancel{2} & \cancel{5} & 10 & \cancel{18} & 21 & \cancel{8} \\ \cancel{0} & \cancel{1} & \cancel{6} & 2 & \cancel{\mathbf{0}} & 18 & \cancel{5} \\ \cancel{0} & \cancel{2} & \cancel{3} & 1 & \cancel{14} & 12 & \cancel{\mathbf{0}} \\ \cancel{0} & \cancel{\mathbf{0}} & 4 & 6 & \cancel{6} & 8 & \cancel{4} \end{bmatrix}$$

Step 4: We have no $c_{i,j}$'s that are unmarked and equal to zero, so we move on.

Step 5: There are seven assigned $c_{i,j}$'s and we needed 7, so we are at the optimal solution.

Looking at our original $c_{i,j}$ matrix we find that the optimal cost is

$$2 + 18 + 14 + 5 + 8 + 8 + 12 = 67$$

We can affirm this by looking at the value of the sum of the components of the current dual feasible solution

$$\sum_{i=1}^{n} u_i + \sum_{j=1}^{m} v_j = (1 + 5 + 6 + 2 + 4 + 10 + 8) + (0 + 10 + 9 + 4 + 4 + 2 + 2) = 67$$

These values are identical, affirming that we must be at an optimal solution and that the algorithm was successful.