Hungarian Method

The Hungarian Method for the Assignment Problem

TL;DR

Theorem

> If a number is added to or subtracted from all the entries of any one row or column of a cost matrix, then an optimal assignment for the resulting cost matrix is also an optimal assignment for the original cost matrix.

Hungarian Algorithm

1. Subtract the smallest entry in each row from all the entires of this row
2. Subtract the small entry in each column from all the entries of its column
3. Draw lines through appropriate rows and columns so that all the zero entries of the cost matrix are covered and the minimum number of lines is used
4. *Test For Optimality*:
   - If the minimum number of covering lines is n, an optimal assignment of zeros is possible and we are finished
   - If the minimum number of covering lines is less than n, an optimal assignment of zeros is not yet possible. Proceed to step 5
5. Determine the smallest entry not covered by any line. Subtract this entry from each uncovered row, and then add it to each covered column. Return to Step 3.

The Simple Assignment Problem

Suppose there are four individuals (1-4) are available for four jobs (1-4). This information can be represented well by a **qualification** (cost) matrix:

$$Q = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

An assignment is said to be **complete** if it is impossible to improve an assignment by placing an unassigned individual a job for which they are unqualified for (complete but not optimal). If an assignment is complete, it is natural to attempt an improvement by means of a **transfer**. An **incomplete** assignment is an assignment for which you can select another individual for a qualified position as an example:

Complete

$$Q = \begin{bmatrix} 1 & 1 & 1* & 0 \\ 0 & 0 & 1 & 1* \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Incomplete

$$Q = \begin{bmatrix} 1* & 1 & 1 & 0 \\ 0 & 0 & 1* & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

We can say that an optimal assignment can be formulated by a sequence of transfers followed by additional assignments until this is no longer possible.

Every assigned individual involved in a transfer is labeled as **essential individual** and every job assignment to an inessential individual an **essential job**. With said definition we can then state:

> ***Lemma 1***. For an given assignment, if an individual is assigned to a job, then either the individual or the job is essential, not both.

Corollary 1. For all assignments, the number of individuals assigned to a jobs equals the number of essential individuals and jobs.

Lemma 2. For a given assignment, if an individual is assigned to a job and qualifies for another, unassigned, ob then the individual is essential.

Lemma 3. For a given assignment, if every transfer leaves a job assigned then the job is essential.

Combining the lemmas we find:

Theorem 1: For a given assignment, if every transfer leads to a complete assignment then, for every individual qualified for a job, either the individual or the job is essential, and possibly both.

Theorem 2: There is an assignment which is complete after every possible transfer.

Now that we can guarantee complete assignments based on transfers, lets shift toward allowing **budgets** to account for the value of an individual assigned to a job for which they are qualified. A budget is said to be **adequate** if, for every individual qualified for a job, either the individual or the job is alloted one unit, and possibly both.

Theorem 4: There is an adequate budget and assignment such that the total allotment of the budget equals the number of jobs assigned to qualified individuals.

We can then state: The largest number of jobs that can be assigned to qualified individuals is equal to the smallest total allotment of any adequate budget. Any assignment is optimal if and only if it is complete after every possible transfer.

The General Assignment Problem

Suppose n individuals $(i = 1 \cdots n)$ are available for n jobs $(j = 1 \cdots n)$ and that a rating matrix R is given, where r_{ij} are positive integers, for all i and j. An assignment consists of the choice of one job j_i for each individual i such that no job is assigned to two different individuals. Therefore the General Assignment Problem is a permutation of the sum:

$$r_{1j_1} + r_{2j_2} + \cdots + r_{nj_n}$$

that produces the largest sum.

The dual problem considers **adequate budgets**, that is, allotments of non-negative amounts u_i to each individual and v_i to each job in such a manner that the sum of the allotments to the i^{th} individual and the j^{th} job is not less than his rating in that job.

$$u_i + v_j \geq r_{ij} \;\; (i, j = 1, \cdots, n)$$

Therefore the dual to the General Assignment Problem is minimizing the sum for adequate allotment

$$u_1 + \cdots + u_n + v1 + \cdots + v_n$$

Theorem 5: The total allotment of any adequate budget is not less than the rating sum of any assignment

It is then an immediate consequence if an adequate budget and assignment can be exhibited such that the total allotment equals the rating sum, then they must be simultaneously a solution of the assignment problem and its dual.

Centralized Hungarian Method Dissertation

Background

- **Bipartite Graph**: A Graph $G = (V, E)$, where the vertex set V is decomposed into two disjoint sets of vertices A and B respectively, such that no two vertices in the same set are adjacent. In general, it is said that G has a bipartition $(A, , B)$.
- **Matching**: A set of edges without common vertices without common edges
- **Maximum Cardinality Matching**: A matching that contains the large possible number of edges.
- **Vertex Cover**: A set of vertices such that each edge is incident on at least one vertex of the set.
- **Minimum Vertex Cover**: A vertex cover that contains the smallest possible number of vertices.

Remark 1

In a bipartite graph, the number of edges in a maximum cardinality matching equals the number of vertecies in a minimum vertex cover. In fact, due to this inter-relation between a matching and a vertex cover, algorithms used for finding a maximum cardinality M, can be extended to finding a corresponding minimum vertex cover $V_c \subset V$.

Using the definitions above (as well as the remark) the **Minimum Weight Bipartite Matching Problem** can be stated as:

Given a graph $G = (V, E)$ with bipartition (A, B) and weight function $w : E \to \mathbb{R}$, the objective is to find a maximum cardinality M of minimum cost, where the cost of matching M is given by $c(M) = \sum_{e \in M} w(e)$.

Next we can state the dual of the Minimum Weight Bipartite Matching Problem:

Given a graph $G = (V, E)$ with bipartition (A, B) and weight function $w : E \to \mathbb{R}$ and a vertex labeling function $y : V \to R$, the objective is to find a feasible labeling of maximum cost, where a feasible labeling is a choice of labels y such that $w(a, b) \geq y(a)+y(b) \forall (a, b) \in E$, and the cost of labeling is given by $c(y) = \sum_{a \in M} w(a)+\sum_{b \in M} w(b)$.

Moreover, given a feasible labeling y, and **equality graph** $G_y = (V, E_y)$ is a sub-graph of G such that $E_y = \{(a, b) \mid y(a) + y(b) = w(a, b)\}$

Theorem (Kuhn-Munkres)

Given a bipartite graph $G = (V, E)$ with bipartition (A, B), a weight function $w : E \to R$, and a vertex labeling function $y : V \to R$, let M and y be feasible (M is a perfect matching and y is a feasible labeling). Then M and y are optimal if and only if $M \subseteq E_y$, i.e. each edge in M is also in the set of equality subgraph edges E_y, given by (68).

Initialization: Given a graph $G = (V, E)$ with bipartition (A, B), and a weight function w (Figure 29a), the Hungarian Method begins with an arbitrary feasible labeling y (Figure 29b), generates the corresponding equality subgraph edges Ey using (68) (Figure 15c), finds a maximum cardinality matching $M_y \subseteq E_y$, and a corresponding minimum vertex cover $V_{c_y} \subset V$, with bipartition (A_{c_y}, B_{c_y}) as per Remark 1 (Figure 15d). The algorithm then performs the following two-step iterations repeatedly, until My is a perfect matching:

1. The algorithm uses V_{c_y} to isolate a set of candidate edges $E_{cand} \subseteq E \backslash E_y$ as per Remark 2, and calculates the following (Figure 16a):

$$\delta = min(a, b) \in E_{cand} slack(w, y, a, b)$$

where slack(w, y, a, b) = w((a, b)) − (y(a) + y(b))

- Using δ and V_{c_y}, the algorithm updates y as follows (Figure 16b):

$$y(a) = y(a) - \delta, \forall a \in A_{c_y} y(b) = y(b) + \delta, \forall b \in B \backslash B c_y$$

2. For the updated y, the algorithm finds the corresponding equality subgraph edges E_y using (68) (Figure 16c), to find a maximum cardinality matching $M_y \subseteq E_y$, and a corresponding minimum vertex cover $C_{c_y} \subset V$ as per Remark 1.

Lemma

Given $G = (V, E)$ with bipartition (A, B), a weight function w, a feasible vertex labeling function y, and a corresponding matching M_y, every two-step iteration (steps (a) and (b)) of the Hungarian Method results in the following: (i) An updated y that remains feasible. (ii) An increase in the matching size $|M_y|$, or no change in the matching M_y, but an increase in $|A_{c_y}|$ (and corresponding decrease in $|B_{c_y}|$, such that $|A_{c_y}| + |B_{c_y}| = |M_y|$).

Distributed Hungarian Method

Similar to the centralized Hungarian Method, the idea behind the distributed version is to execute in a finite amount of steps. However, execution of these steps are distributed across the robots. In particular, each robot shares an operates on limited information.

All robots begin by running their individual copies of the algorithm at some initial time $t_0 \in \mathbb{R}_{\geq 0}$, and continue to run it synchronously at intervals of T_s seconds ($T_s \in \mathbb{R}_{>0}$). The robot maintains and updates a global iteration counter $\alpha \in \mathbb{N}$

Each robot starts with initial information (R, P, c^i) and creates a bipartite graph $G^i_{orig} = (V, E^i_{orig})$ with bipartition (R, P), and weight function $w^i_{orig} : E \to \mathbb{R}$. Moreover, throughout the execution of the algorithm, robot i shares, updates, and operates on a bipartite graph $G^i_{lean} = (V, E^i_{lean})$ with bipartition (R, P), a corresponding weight function $w^i_{lean} =: E^i_{lean} \to \mathbb{R}$, and a corresponding vertex labeling function $y^i : V \to \mathbb{R}$.

On every iteration of the algorithm, the following two actions are performed:

1. **Send**: Robot i sends a message msg^i, to each of its outgoing neighbors, where $msg^i = (G^i_{lean}, w^i_{lean}, y^i, \gamma^i)$ where γ^i denotes the cumulative number of completed stages (two-step iterations).
2. **Receive and Compute**: Robot i receives the message from its incoming neighbors, and through computations involving the received information, its own message msg^i and its original information (G^i_{orig}, w^i_{orig}) robot i prepares its new message for sending.

Every robot extracts specific information from the messages it receives from its neighbor with the largest γ value (including its own). Robot i then combines the information into an updated $(G^i_{lean}, w^i_{lean}, y^i, \gamma^i)$ and continues with the Hungarian Method. The caveat is that robot i must have sufficient candidate edges in in E^i_{cand}, for it to be able to execute Step 2. If this is not the case, robot i gathers whatever pertinent candidate edges it has (drawing an edge if required, from its original information (G^i_{orig}, w^i_{orig})) and updates E^i_{cand} accordingly.