Matchings in Weighted Bipartite Graphs – Kuhn-Munkres Algorithm

Textbook Reading:

Chapter 4, Subsection 4.1.3, pp. 104-107

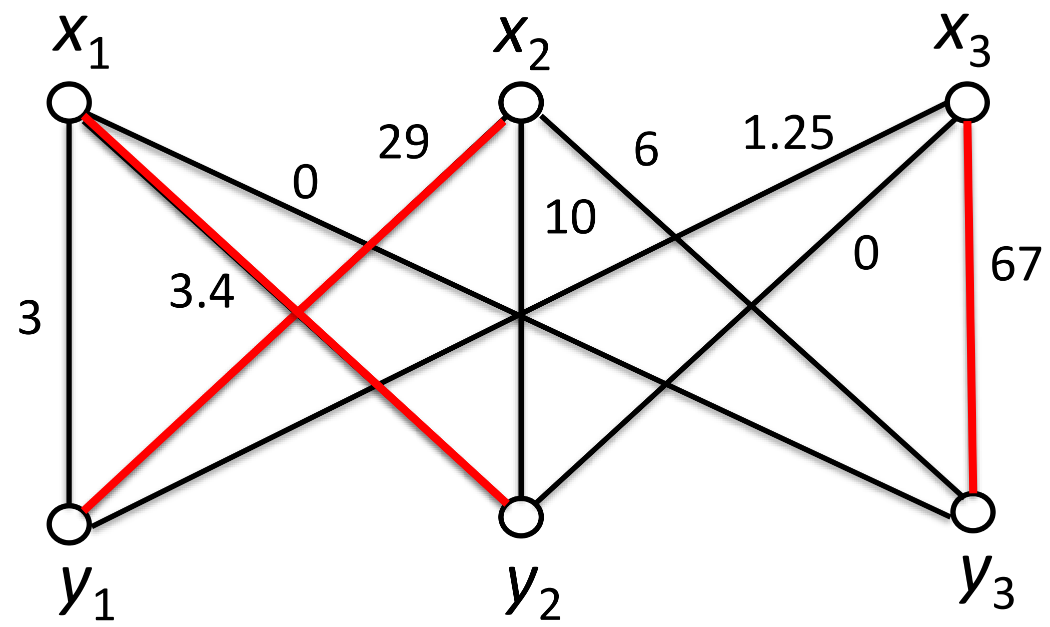
Perfect Matchings in Weighted Bipartite Graphs

Weighted complete bipartite graph:

$G = (V,E)$ with vertex bipartition $V = X \cup Y$, where $X = \{x_1, . . . , x_{n-1}\}$ and $Y = \{y_1, . . . , y_n\}$.

Each edge x_iy_j of G is assigned the a real weight ω_{ij}, $i, j \in \{1, . . . , n\}$.

Sample weighted complete bipartite graph



It can be input using the two-dimensional array (matrix):

$$\begin{pmatrix} 3 & \color{red}{3.4} & 0 \\ \color{red}{29} & 10 & 6 \\ 1.25 & 0 & \color{red}{67} \end{pmatrix}$$

A perfect matching corresponds to a traversal of the matrix.

Weight of a matching

The **weight** of a matching *M*, denoted by $\omega(M)$, is the sum of the weights of its edges, i.e.

$$\omega(M) = \sum_{e\in M} \omega(e).$$

Maximum Weight Perfect Matching

A **maximum-weight perfect matching** is one that maximizes $\omega(M)$.

There are many natural applications for finding a maximum weight perfect matching. For example, finding an assignment of workers to jobs so that the total effectiveness of the workers is optimized.

Brute force very inefficient

A brute-force algorithm that enumerates all $n!$ perfect matchings and chooses one of maximum weight is hopelessly inefficient.

There is a combinatorial explosion of perfect matchings.

Kuhn-Munkres algorithm

- We now describe an $O(n^3)$ algorithm due to Kuhn and Munkres for finding a maximum perfect matching in a weighted complete bipartite graph.
- The Kuhn-Munkres algorithm utilizes the Hungarian algorithm, together with the notion of a ***feasible vertex weighting***.

Feasible Vertex Weighting

A **feasible vertex weighting** is a mapping ϕ from *V* to the real numbers such that for each edge *xy* $\in$ *E*

$$\phi(x) + \phi(y) \geq \omega(xy).$$

Let

$$\phi(V) = \sum_{v\in V} \phi(v).$$

PSN. Handle on knowing whether we have a perfect matching

a) Show that for **any** perfect matching *M* and **any** feasible vertex weight ϕ,

$$\omega(M) \leq \phi(V).$$

b) Suppose there exists a perfect matching $M*$ and a feasible vertex weighting $\phi*$, such that

$$\omega(M*) = \phi*(V).$$

Then $M*$ is a **maximum-weight perfect matching** and $\phi*$ is a **minimum-weight feasible vertex weighting**, i.e., minimizes $\phi*(V)$ over all feasible vertex weightings.

Equality Subgraph

Equality subgraph consists of all edges $xy \in E$ such that

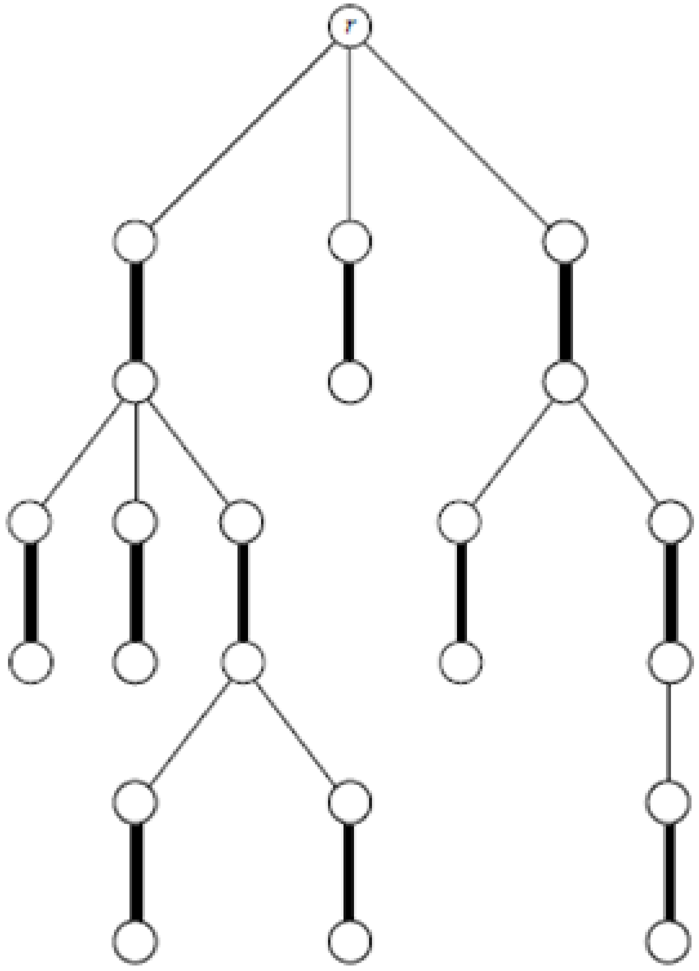
$$\phi(x) + \phi(y) = \omega(xy).$$

A perfect matching in the **equality subgraph** is necessarily a **maximum-weight** perfect matching in G.

Initial feasible vertex weighting ϕ

Choose initial feasible vertex weighting so that equality subgraph **spans** X. One such equality subgraph is obtained by choosing $\phi(x)$ for each $x \in X$ to be the **maximum** value of $\omega(xy)$ over all edges xy incident with x and setting $\phi(y)$ = 0 for all $y \in Y$.

Grow M-alternating Tree in equality subgraph



Apply Hungarian Algorithm to increase size of matching M in the equality subgraph

Keep applying Hungarian algorithm in the equality subgraph until either

1) a **perfect matching *M* is found in equality subgraph**

OR

2) it terminates with an ***M*-alternating tree in the equality subgraph**

Adjusting the Feasible Weighting

Once Hungarian algorithm terminates construct a new feasible weighting ϕ' so that its equality subgraph still contains the *M*-alternating tree *T*, but *T* it can be expanded

Let *S* be the vertices of the *M*-alternating tree *T* that lie in *X, i.e.,* $S = V(T) \cap X$.

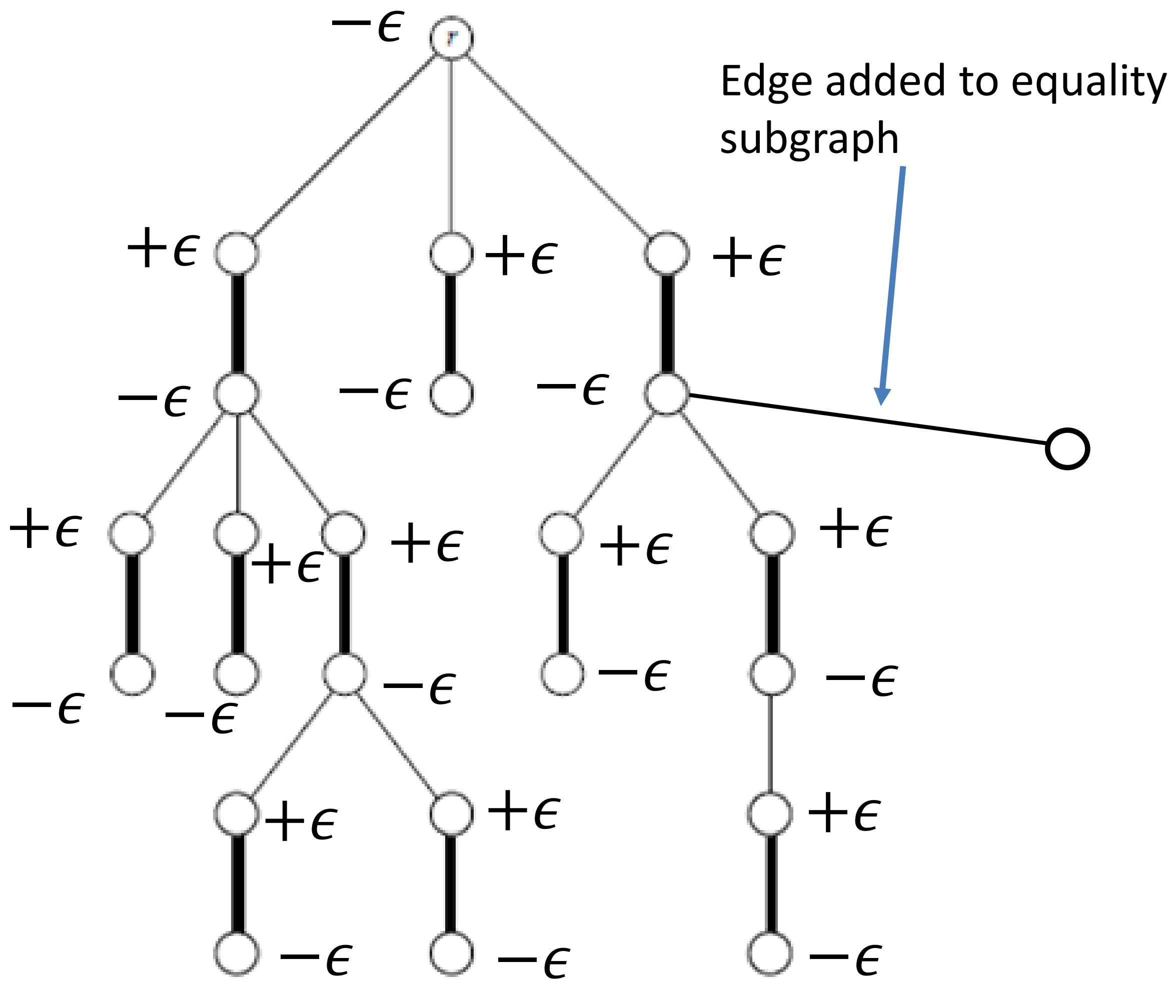
The new feasible weighting ϕ' is constructed as follows. Set

$$\varepsilon = \min\{\phi(x) + \phi(y) - \omega(xy) \mid x \in S, y \in Y - \Gamma_\phi(S)\}. \qquad (4.1.10)$$

For each $v \in V(G)$, the weighting ϕ' is defined by

$$\phi'(v) = \begin{cases} \phi(v) - \varepsilon & v \in S, \\ \phi(v) + \varepsilon & v \in \Gamma_\phi(S), \\ \phi(v) & \text{otherwise.} \end{cases} \qquad (4.1.11)$$

Expand *M*-alternating Tree in new equality subgraph



Overview of Kuhn-Munkres Algorithm

1) Use Hungarian algorithm to either find a perfect matching in equality subgraph in which case $\omega(M) = \phi(V)$ and we are done, i.e., M is a maximum-weight perfect matching,

OR

2) Change feasible vertex weighting, so that M-alternating tree can be expanded in new equality subgraph.

Since bipartite graph *G* is complete, eventually case 1) will occur!

Maximum-Weight Perfect Matching in General Bipartite Graph

PSN. Show that the problem of finding a maximum-weight **matching** in **any** weighted bipartite graph can be transformed (reduced) to the problem we just solved of finding a maximum-weight **perfect matching** in a **complete** bipartite graph.

Important Special Cases

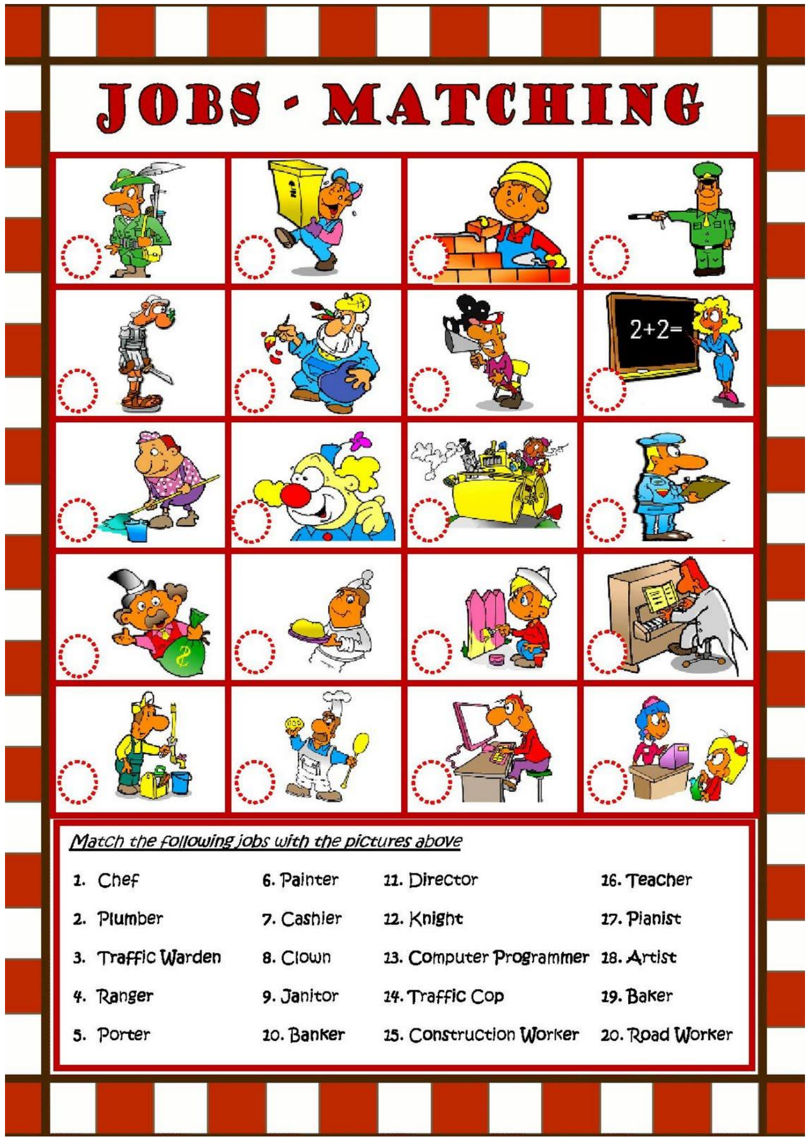
- Thus, we can use the Kuhn-Munkres algorithm to compute a maximum-weight matching in a weighted bipartite graph.
- As a special case we can compute a maximum cardinality matching in a bipartite graph.
- Computing a perfect matching in a bipartite graph is a special case of computing a maximum cardinality matching.

Complexity Analysis

The worst-case complexity is $O(mn)$

In terms of just *n* worst-case complexity is $O(n^3)$

Surprise Quiz!



Match the following jobs with the pictures above

1. Chef	6. Painter	11. Director	16. Teacher
2. Plumber	7. Cashier	12. Knight	17. Pianist
3. Traffic Warden	8. Clown	13. Computer Programmer	18. Artist
4. Ranger	9. Janitor	14. Traffic Cop	19. Baker
5. Porter	10. Banker	15. Construction Worker	20. Road Worker