Chapter 2

Zachary Campbell

March 28, 2018

1 The Hungarian algorithm

In this section we use the motivation of the linear programs to develop an algorithm for simultaneously solving the maximum weight matching and minimum weight vertex cover problems. Our algorithm works by taking every exposed node on the left, and from each such node building a collection of alternating paths. We define this collection formally.

Definition (Alternating tree). Let $G = (U, V, E)$ be a bipartite graph, and M a matching on G. An *alternating tree* with respect to M is a tree which satisfies two conditions:

- the tree contains exactly one node $u \in U$. We call u the *root* of the tree.
- all paths between the root and any other node in the tree are alternating paths.

Let's remind ourselves what the primal-dual linear programs motivate. We want to minimize $\sum w_{uv}$, and maximize $\sum(y_u + y_v)$. Moreover, we want optimal solutions such that $\sum w_{uv} = \sum(y_u + y_v)$. For our primal, we are keeping track of edge weights. For the dual, we will be keeping track of a "labeling" on vertices, given by the y_u and y_v values. We define what a labeling is now.

Definition (Labeling). A *vertex labeling* on a weighted bipartite graph $G = (U, V, E)$ is a function $l : U \cup V \to \mathbb{N}$. We call the labeling *feasible* if for all $u \in U$ and $v \in V$, $l(u) + l(v) \geq w_{uv}$.

This labeling corresponds to our dual variables; i.e. a feasible labeling is a feasible dual solution. It will be helpful us to look at a certain subset of our graph where the labeling is exact ($l(u) + l(v) - w_{uv} = 0$).

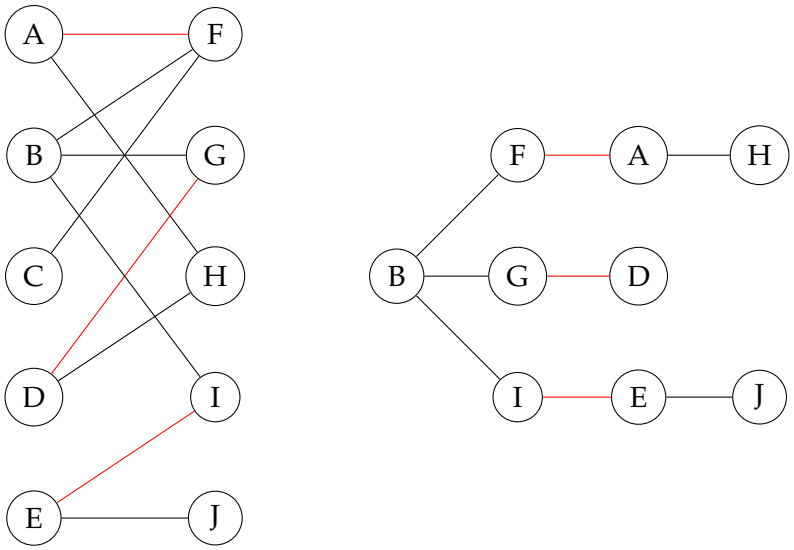

Figure 1: Bipartite graph with matching, corresponding alternating tree rooted at vertex B.

Definition (Equality subgraph). The *equality subgraph* of $G = (U, V, E)$ is the graph $G_l = (U, V, E_l)$, where

$$E_l = \{(u, v) \ : \ l(u) + l(v) = w_{uv}\}.$$

In Figure 5 we show a bipartite graph along with its corresponding equality graph.

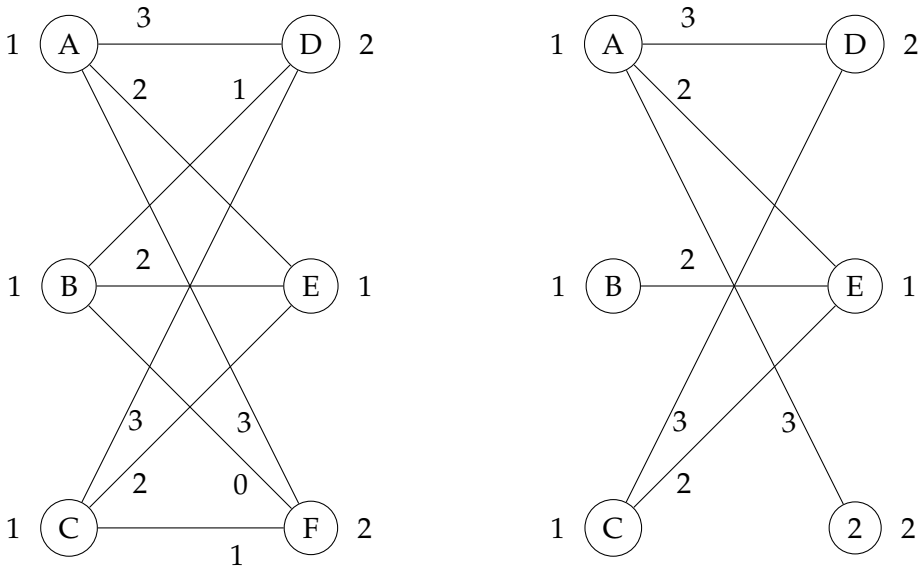

Figure 2: A weighted bipartite graph and its corresponding equality subgraph.

Theorem (Kuhn-Munkres). If l is a feasible labeling and M is a perfect matching in G_l then M is

a max-weight matching.

Proof. Let $M^{'}$ be a perfect matching in G. Since every $u \in U \cup V$ is matched by exactly one edge in $M^{'}$, then

$$\sum_{(u,v)\in M'} w_{uv} \leq \sum_{(u,v)\in M'} (l(u) + l(v)) = \sum_{w\in U\cup V} l(w).$$

This says that the sum of our label values is an upper bound on the weight of any perfect matching. Now suppose that M is a perfect matching in G_l. Then

$$\sum_{(u,v)\in M} w_{uv} = \sum_{w\in U\cup V} l(w).$$

So $\sum_{(u,v)\in M'} w_{uv} \leq \sum_{(u,v)\in M}$, meaning M must be maximum weight. □

Lemma 2. Let $S \subseteq U$ and $T = N_l(S) \neq V$. Set

$$\alpha_l = \min_{u\in S,\ v\notin T} \{l(u) + l(v) - w_{uv}\}$$

and

$$l^{'}(w) = \begin{cases} l(w) - \alpha_l & \text{if } w \in S \\ l(w) + \alpha_l & \text{if } w \in T \\ l(w) & \text{otherwise.} \end{cases}$$

Then $l^{'}$ is a feasible labeling, and

1. If $(u, v) \in E_l$ for $u \in S$ and $v \in T$ then $(u, v) \in E_{l'}$.
2. If $(u, v) \in E_l$ for $u \notin S$, and $v \notin T$, then $(u, v) \in E_{l'}$.
3. There is some edge $(u, v) \in E_{l'}$ for $u \in S$, $v \notin T$.

Proof. First, we show that $l^{'}$ is feasible. For $u \in U$, $v \in V$, there are four possibilities:

- if $u \in S$ and $v \in T$ then $l^{'}(u) + l^{'}(v) = l(u) + l(v)$.
- if $u \notin S$ and $v \notin T$, $l^{'}(u) = l(u)$ and $l^{'}(v) = l(v)$, so $l^{'}(u) + l^{'}(v) = l(u) + l(v)$.

- if $u \in S$ and $v \notin T$, $l'(u) - \alpha$ and $l'(v) = l(v)$. We know $\alpha = \min_{u \in S, v \notin T}\{l(u) + l(v) - w_{uv}\}$, which means $\alpha \leq l(u) + l(v) - w_{uv}$, and thus $l'(u) + l'(v) \geq w_{uv}$.
- if $u \notin S$ and $v \in T$, $l'(u) = l(u)$ and $l'(v) = l(v) + \alpha$, which is clearly feasible.

(1) and (2) follow from above. To see (3), note that there is some (u, v) with $u \in S$, $v \in T$ such that $\alpha = l(u) + l(v) - w_{uv}$, so when we take $l'(u) = l(u) - \alpha$ and $l'(v) = l(v)$, we get

$$\begin{aligned} l'(u) + l'(v) - w_{uv} &= l(u) - \alpha + l(v) - w_{uv} \\ &= l(u) + l(v) - w_{uv} - \alpha \\ &= \alpha - \alpha \\ &= 0. \end{aligned}$$

This is exactly what it means for an edge (u, v) to be in E_l. □

We now look at the Hungarian method for finding maximum-weight matchings on bipartite graphs. This method was originally developed by Kuhn and Munkres, who named it in honor of the Hungarian mathematicians Kőnig and Egervary.

The Hungarian Method

1. Choose initial feasible labeling l and matching M in G_l.
2. If M is perfect in G_l, we are done. Otherwise, pick exposed vertex $u \in U$. Set $S = \{u\}$, $T = \emptyset$.
3. If $N_l(S) = T$, update labels as in lemma (this forces $N_l(S) \neq T$).
4. If $N_l(S) \neq T$, pick $v \in N_l(S) \setminus T$
   - If v is exposed, $p = u \to v$ is an augmenting path. Set $M := M \oplus p$. Go to 2.
   - If v is matched to some w, expand our alternating tree. $S := S \cup \{w\}$, $T := T \cup \{v\}$. Go to 3.

We now provide an example of this algorithm. Our initial matching is $M = \{(B, E), (C, D)\}$ (see

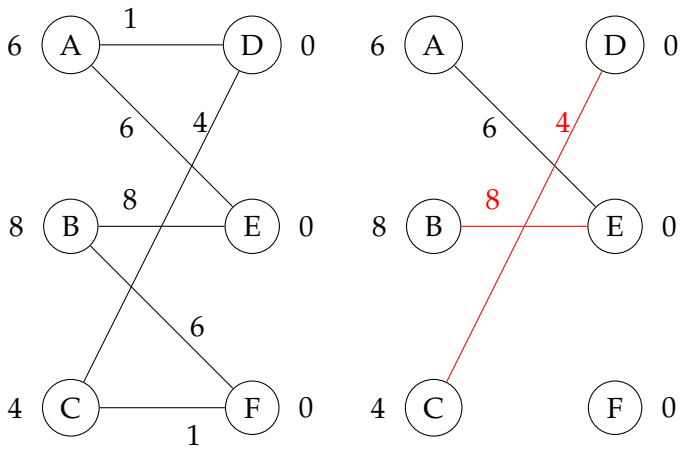

Figure 3: Bipartite graph (left) and corresponding equality graph (right) with initial matching

Figure 6). Note that the current state of the graph is primal-dual feasible. Our algorithm chooses an exposed vertex in U, say A. So we have $S = \{A\}$ and $T = \emptyset$. We have that $N_l(S) \neq T$, so we find $E \in N_l(S) \setminus T$. E is matched, so we grow our alternating tree as follows: $S := S \cup \{B\} = \{A, B\}$, $T := T \cup \{E\} = \{E\}$. At this point $N_l(S) = T$, so we adjust our dual variables. Calculate $\alpha = \min_{u \in S, v \notin T}\{l(u) + l(v) - w_{uv}\} = 2$ from edge (B, F). Our new equality graph is shown in Figure 7.

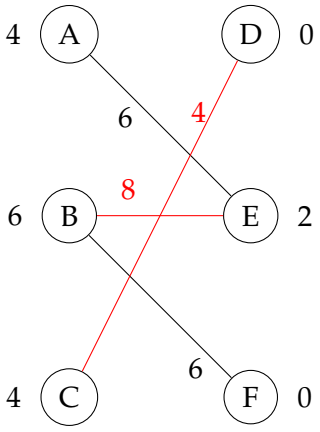

Figure 4: Second equality graph.

Now, $S = \{A, B\}$ is the same, but $N_l(S) = \{E, F\}$ has changed. $T = \{E\}$, so $N_l(S) \neq T$. So we choose $F \in N_l(S) \setminus T$. F is unmatched, meaning it is an endpoint of an augmenting path. In particular, $p = A, E, B, F$ is an augmenting path. Thus we improve our matching with $M := M \oplus p = \{(A, E), (B, F), (C, D)\}$. Our equality graph with the new matching is given in Figure 8.

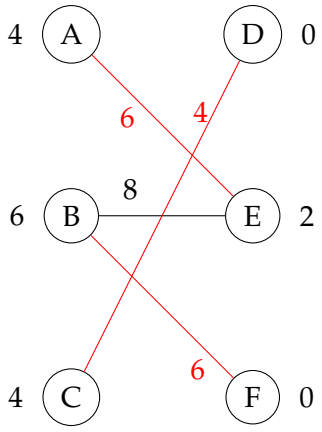

Figure 5: Equality graph after augmenting.

This is a perfect matching on the equality graph, so this matching must be a maximum weight matching on the graph. We can check that the values of the primal and dual solutions agree. The sum of weights in the matching is $6 + 6 + 4 = 16$, and the sum of the values of our dual variables is $4 + 6 + 4 + 2 = 16$.

Note that if we want to just find a maximum cardinality matching on a bipartite graph, we can just give all edges weight 1 and run this algorithm.

This algorithm was one of the first primal-dual algorithms developed, and it anticipated many later variations on the same theme. It displays the surprising connection between combinatorial optimization and linear programming, which we explore in the next section.