Introduction to Optimization

CO 250

Kevin Shu

Lecture 18

Live (interactive) version of these slides: https://ootks.github.io/co250-lectures/lecture18.html

Lecture Outline

- Review: Integrality of the Bipartite Matching Polytope
- The Primal Minimum Cost Matching LP
- The Dual LP and An Algorithm

Matching Linear Programs

Matching Linear Programs

Last lecture we studied the **minimum cost perfect matching** problem: given a graph $G = (V, E)$ with edge costs c_e, find a perfect matching M minimizing $c(M) = \sum_{e \in M} c_e$.

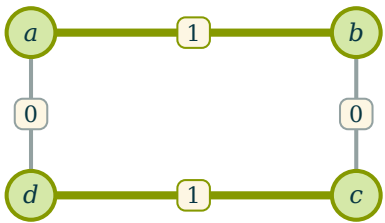
Introducing a variable x_e for each edge, the integer program is

$$\begin{aligned} \min \quad & \sum_{e \in E} c_e x_e \\ \text{s.t.} \quad & \sum_{e \in E : v \in e} x_e = 1 \quad \text{for each } v \in V \\ & x_e \geq 0 \\ & x_e \in \mathbb{Z}. \end{aligned}$$

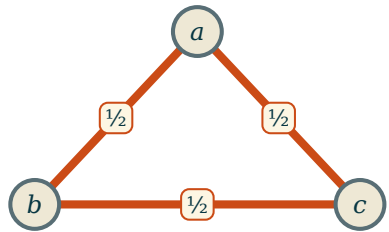
Dropping $x_e \in \mathbb{Z}$ gives the **LP relaxation**. The LP relaxation can sometimes be a genuinely different problem than the min cost perfect matching problem.

Matching Linear Programs

We saw that the LP relaxation sometimes returns an integral solution, corresponding to a matching, and sometimes a fractional solution.



Cycle on 4 vertices (bipartite)
$x^* = (1, 0, 1, 0)$ — integral

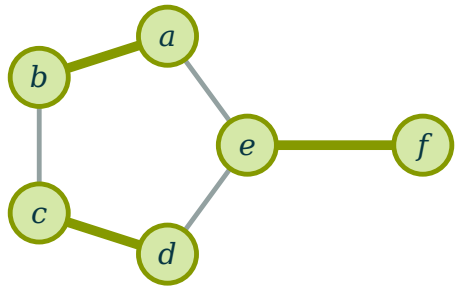


Triangle (not bipartite)
$x^* = (\frac{1}{2}, \frac{1}{2}, \frac{1}{2})$ — fractional

The key difference: the first graph is **bipartite**, the second is not.

Matching Linear Programs

A graph that is not bipartite can still have a perfect matching.



Matching Linear Programs

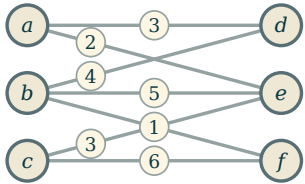
Integrality of Bipartite Perfect Matching Polytope
For any bipartite graph with n vertices and m edges, let

$$P = \left\{ x \in \mathbb{R}^m : \sum_{e \in E : v \in e} x_e = 1 \text{ for all } v \in V,\ x \geq 0 \right\}.$$

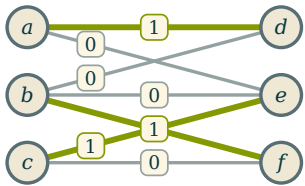
Every vertex of P has all entries equal to 0 or 1.

So for a bipartite graph, the LP relaxation always has an **integral** optimal vertex, with the same optimal value as the minimum cost perfect matching.

Matching Linear Programs



The costs.



The optimal solution.

$$
\begin{aligned}
\min \quad & 3x_{ad} + 2x_{ae} + 4x_{bd} + 5x_{be} + x_{bf} \\
& + 3x_{ce} + 6x_{cf} \\
\text{s.t.} \quad & x_{ad} + x_{ae} = 1 && (a) \\
& x_{bd} + x_{be} + x_{bf} = 1 && (b) \\
& x_{ce} + x_{cf} = 1 && (c) \\
& x_{ad} + x_{bd} = 1 && (d) \\
& x_{ae} + x_{be} + x_{ce} = 1 && (e) \\
& x_{bf} + x_{cf} = 1 && (f) \\
& x \geq 0
\end{aligned}
$$

$$x^*_{ad} = x^*_{bf} = x^*_{ce} = 1, \quad x^*_e = 0 \text{ otherwise}$$

$$\sum_{e \in E} c_e x^*_e = 3 + 1 + 3 = 7$$

The Primal and Dual LPs

The Primal Matching LP

Let $G = (V, E)$ be a graph (not necessarily bipartite) with edge costs c_e. The LP relaxation of the minimum cost perfect matching problem is the **primal**:

$$\begin{aligned} \min \quad & \sum_{e \in E} c_e x_e & \\ \text{s.t.} \quad & \sum_{e \in E : v \in e} x_e = 1 & \text{for each } v \in V \\ & x_e \geq 0 & \text{for each } e \in E. \end{aligned}$$

One **variable** x_e per **edge** $e \in E$.

One **constraint** per **vertex** $v \in V$.

Taking the Dual

In the primal, we get the following relationships between vertices/constraints in the LP and edges/vertices in the graph.

Primal **constraints** $\leftrightarrow$ vertices
so dual **variables** $\leftrightarrow$ vertices.

Primal **variables** $\leftrightarrow$ edges
so dual **constraints** $\leftrightarrow$ edges.

The dual reverses these correspondences.

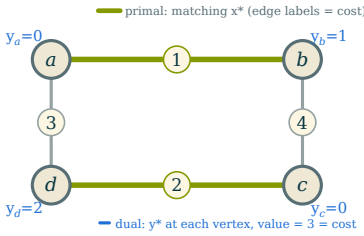
The Dual LP

The primal variable x_e for $e = uv$ appears with coefficient 1 in exactly the two vertex constraints for u and v, so its dual constraint is $y_u + y_v \leq c_e$.

$$\begin{aligned} \text{(P)} \quad \min \quad & \sum_{e \in E} c_e x_e \\ \text{s.t.} \quad & \sum_{e \in E : v \in e} x_e = 1 \quad (v \in V) \\ & x_e \geq 0 \quad (e \in E) \end{aligned}$$

$$\begin{aligned} \text{(D)} \quad \max \quad & \sum_{v \in V} y_v \\ \text{s.t.} \quad & y_u + y_v \leq c_e \quad (e = uv \in E) \end{aligned}$$

Example: Primal and Dual



$$
\begin{aligned}
\min \quad & x_{ab} + 4x_{bc} + 2x_{cd} + 3x_{da} & \\
\text{s.t.} \quad & x_{ab} + x_{da} = 1 & (a) \\
& x_{ab} + x_{bc} = 1 & (b) \\
& x_{bc} + x_{cd} = 1 & (c) \\
& x_{cd} + x_{da} = 1 & (d) \\
& x \geq 0 &
\end{aligned}
$$

$$
\begin{aligned}
\max \quad & y_a + y_b + y_c + y_d & \\
\text{s.t.} \quad & y_a + y_b \leq 1 & (ab) \\
& y_b + y_c \leq 4 & (bc) \\
& y_c + y_d \leq 2 & (cd) \\
& y_d + y_a \leq 3 & (da)
\end{aligned}
$$

Primal optimum $\{ab, cd\}$ and dual optimum $y = (0, 1, 0, 2)$ both have value 3.

Interpreting the Dual

We can think of the dual to the matching LP as a way to **reweight the costs of the edges.**

Pick a vertex $v \in V$. Even if we don't know what the optimal perfect matching is, we know that that perfect matching has one edge incident to v.

Imagine that we take all of the edges that touch the vertex v, and subtract y_v from the cost of all such edges.

The cost of every perfect matching goes down by y_v, so the optimal perfect matchings don't change! Their costs just go down by y_v.

Interpreting the Dual

Interactive demo — open in your browser

Dual Reweight Game

An interactive game in which you subtract dual values y_v from the edges at each vertex and watch the reduced costs change. It is embedded in the live slide and cannot run inside a PDF.

`https://ootks.github.io/co250-lectures/demos/DualReweightGame.html`

https://ootks.github.io/co250-lectures/lecture18.html

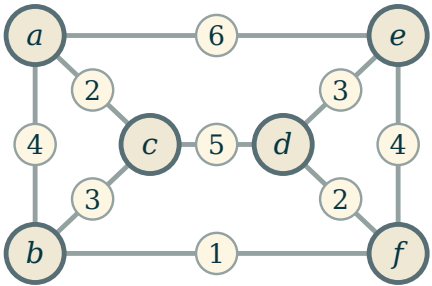
Interpreting the Dual

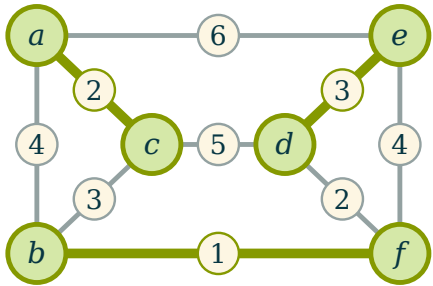
For an edge $e = uv$, you can imagine the 'updated' weight of an edge, given a dual vector y, to be $c_e - y_u - y_v$.

An **equality edge** of a given dual feasible y is an edge e whose updated cost is 0, that is, $c_e = y_u + y_v$.

Complementary slackness says that if e is in an optimal matching, then it is an equality edge of every dual optimal y.

Certifying Optimality





Is this the best perfect matching?

Certifying Optimality

We can use this to certify that a given matching is optimal.

Interactive demo — open in your browser

Dual Reweight Game (non-bipartite)

The same reweighting game on the graph shown at top right; the minimum cost perfect matching $\{ac, bf, de\}$ is highlighted there. It is embedded in the live slide and cannot run inside a PDF.

https://ootks.github.io/co250-lectures/demos/DualReweightGameNonBipartite.html

Certifying Optimality

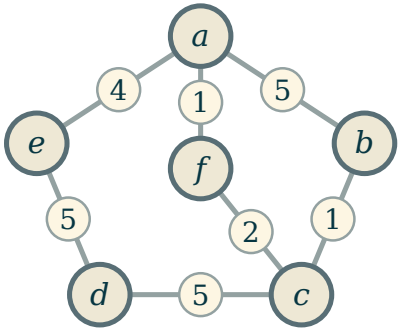
$$
\begin{aligned}
(\mathrm{P}) \quad \min \quad & 4x_{ab} + 3x_{bc} + 2x_{ac} + 3x_{de} + 4x_{ef} \\
& + 2x_{df} + 5x_{cd} + 6x_{ae} + x_{bf} \\
\text{s.t.} \quad & x_{ab} + x_{ac} + x_{ae} = 1 && (a) \\
& x_{ab} + x_{bc} + x_{bf} = 1 && (b) \\
& x_{bc} + x_{ac} + x_{cd} = 1 && (c) \\
& x_{de} + x_{df} + x_{cd} = 1 && (d) \\
& x_{de} + x_{ef} + x_{ae} = 1 && (e) \\
& x_{ef} + x_{df} + x_{bf} = 1 && (f) \\
& x \geq 0
\end{aligned}
$$

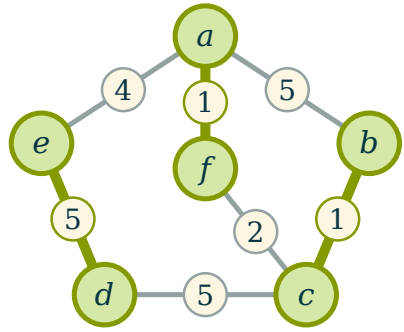
$$
\begin{aligned}
(\mathrm{D}) \quad \max \quad & y_a + y_b + y_c + y_d + y_e + y_f \\
\text{s.t.} \quad & y_a + y_b \leq 4 && (ab) \\
& y_b + y_c \leq 3 && (bc) \\
& y_a + y_c \leq 2 && (ac) \\
& y_d + y_e \leq 3 && (de) \\
& y_e + y_f \leq 4 && (ef) \\
& y_d + y_f \leq 2 && (df) \\
& y_c + y_d \leq 5 && (cd) \\
& y_a + y_e \leq 6 && (ae) \\
& y_b + y_f \leq 1 && (bf) \\
& y \text{ free}
\end{aligned}
$$

Check that this is feasible: $y^* = (y_a^*, y_b^*, y_c^*, y_d^*, y_e^*, y_f^*) = (1, 1, 1, 1, 2, 0)$

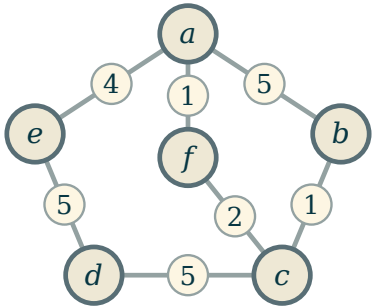
Complementary slackness means that $x_{uv} = 0$ except for x_{ac}, x_{de} and x_{bf}. Note that $\{ac, de, bf\}$ is a perfect matching!

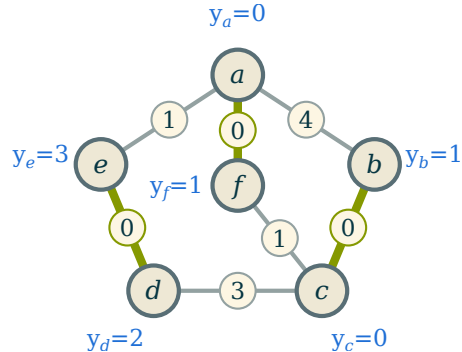
Certifying Optimality





Certifying Optimality





Certifying Optimality

$$
\begin{aligned}
(\mathrm{P}) \quad \min \quad & 5x_{ab} + x_{bc} + 5x_{cd} + 5x_{de} \\
& + 4x_{ae} + x_{af} + 2x_{cf} \\
\text{s.t.} \quad & x_{ab} + x_{ae} + x_{af} = 1 && (a) \\
& x_{ab} + x_{bc} = 1 && (b) \\
& x_{bc} + x_{cd} + x_{cf} = 1 && (c) \\
& x_{cd} + x_{de} = 1 && (d) \\
& x_{de} + x_{ae} = 1 && (e) \\
& x_{af} + x_{cf} = 1 && (f) \\
& x \geq 0
\end{aligned}
$$

$$
\begin{aligned}
(\mathrm{D}) \quad \max \quad & y_a + y_b + y_c + y_d + y_e + y_f \\
\text{s.t.} \quad & y_a + y_b \leq 5 && (ab) \\
& y_b + y_c \leq 1 && (bc) \\
& y_c + y_d \leq 5 && (cd) \\
& y_d + y_e \leq 5 && (de) \\
& y_a + y_e \leq 4 && (ae) \\
& y_a + y_f \leq 1 && (af) \\
& y_c + y_f \leq 2 && (cf) \\
& y \text{ free}
\end{aligned}
$$

Check that this is feasible: $y^* = (y_a^*, y_b^*, y_c^*, y_d^*, y_e^*, y_f^*) = (0, 1, 0, 2, 3, 1)$

$x_{ij} = 1$ when ij is green

The Primal-Dual Algorithm

The Primal-Dual Algorithm

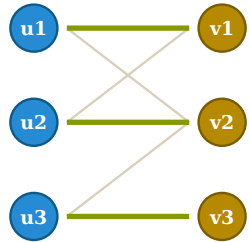
Goal: solve the minimum cost perfect matching problem on a bipartite graph using the dual LP.

The matching LP has an integral optimal vertex, so the simplex method works. But it ignores the graph. We can get a faster algorithm if we make use of the graph structure.

One thing is that it is often practically much easier to find **A** perfect matching, than a minimum cost one.

The Primal-Dual Algorithm

For this class, we'll mostly be dealing with graphs where we can find a perfect matching by eye.



a perfect matching (size 3)

The Primal-Dual Algorithm

In each iteration, we will have a dual feasible y.

Write $E^=$ for the set of **equality edges** of a dual feasible y, the edges so that $c_e = y_u + y_v$.

If $E^=$ contains a perfect matching M, then M is a minimum cost perfect matching by complementary slackness!

The Primal-Dual Algorithm

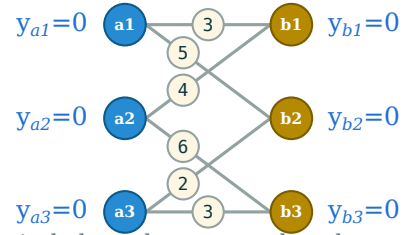
At a high level, the algorithm does the following:

- At each stage, we will have a dual feasible solution y in mind.
- If this dual y is optimal, then there will be an optimal primal solution satisfying complementary slackness. This means there is a perfect matching in the graph induced by $E^{=}$.
- If this dual y is not optimal, then there will be no perfect matching in $E^{=}$; we will need to update y to a new value.

The example below shows how this works.

The Primal-Dual Algorithm

Example. An input to the algorithm

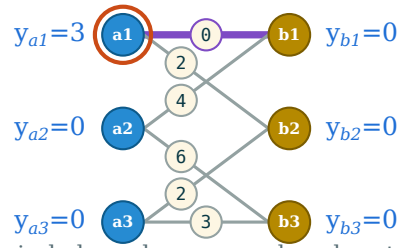


circled numbers are reduced costs

$c_{uv} - y_u - y_v$

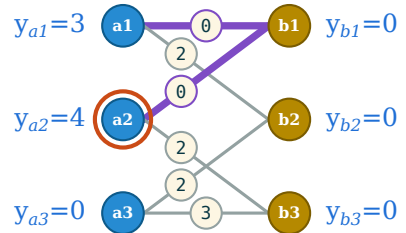
Start with $y = 0$.
This is dual feasible, since every cost is nonnegative.
There are currently no equality edges, so $E^=$ is empty and there is no perfect matching amongst the edges in $E^=$.
We need to increase some of the entries of y.

The Primal-Dual Algorithm



Increase y_{a_1} until an edge at a_1 has reduced cost 0.
Now, $E^{=} = \{a_1 b_1\}$, and there is still no perfect matching.
Let's increase y some more.

The Primal-Dual Algorithm

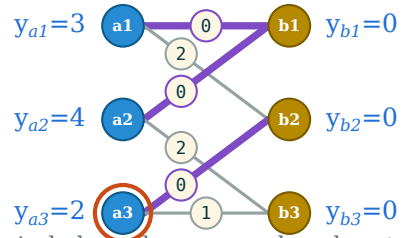


circled numbers are reduced costs

$c_{uv} - y_u - y_v$

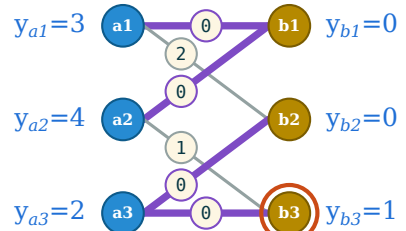
Increase y_{a_2}: $\min\{4, 6\} = 4$.
Now $E^{=} = \{a_1b_1, a_2b_1\}$, but still no perfect matching.

The Primal-Dual Algorithm



Increase y_{a_3}: $\min\{2, 3\} = 2$.
Now $E^{=} = \{a_1b_1, a_2b_1, a_3b_2\}$, but still no perfect matching.

The Primal-Dual Algorithm

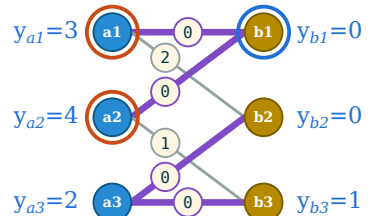


circled numbers are reduced costs

$c_{uv} - y_u - y_v$

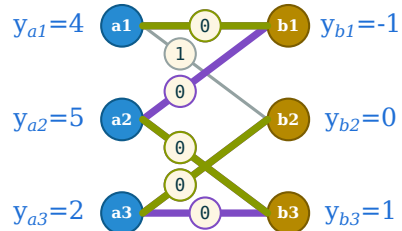
Every a_i has an equality edge, but b_3 does not. Increase y_{b_3}: $\min\{2, 1\} = 1$.
Now $E^= = \{a_1b_1, a_2b_1, a_3b_2, a_3b_3\}$, but still no perfect matching.
We can't increase any individual y_v any more!
To make more progress, we will need to also decrease the weights on some vertices.

The Primal-Dual Algorithm



$y = (3, 4, 2; 0, 0, 1)$, $\sum_v y_v = 10$.
$E^= = \{a_1b_1,\ a_2b_1,\ a_3b_2,\ a_3b_3\}$.
But there is no perfect matching: a_1 and a_2 both need b_1.
We can use this structure though: imagine increasing **both** y_{a_1} and y_{a_2} by 1, and also **decreasing** y_{b_1} by 1. The edges a_1b_1 and a_2b_1 keep the same total weight $y_u + y_v$, so they remain equality edges.
We can do this until another edge from a_1, a_2 reaches reduced cost 0: $\min\{2, 1\} = 1$.

The Primal-Dual Algorithm



circled numbers are reduced costs

$c_{uv} - y_u - y_v$

$y = (4, 5, 2; -1, 0, 1)$, $\sum_v y_v = 11$.
$E^=$ has the perfect matching
$M = \{a_1b_1,\ a_2b_3,\ a_3b_2\}$.
$c(M) = 3 + 6 + 2 = 11 = \sum_v y_v$, so M is a minimum cost perfect matching.

The Primal-Dual Algorithm

If at some point, we have found some y so that $E^{=}$ contains a perfect matching, then we are done; that perfect matching is minimum cost.

The tricky thing is finding a way to update y when $E^{=}$ has no perfect matching. To do this, we will make use of a combinatorics theorem.

The Primal-Dual Algorithm (Hall's Theorem)

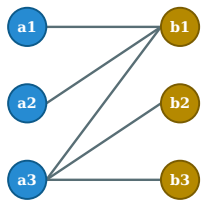
For a set of vertices, $S \subseteq V$, the **neighborhood** of S is the set $\mathcal{N}(S) \subseteq V$ of vertices that share an edge with some vertex of S.

For a bipartite graph G whose vertices are separated into partitions V_1 and V_2, a **deficient set** is a set $S \subseteq V$ so that

- S is either contained in V_1 or V_2
- $\mathcal{N}(S)$ has fewer elements than S does.

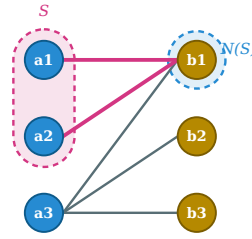
Hall's Theorem
Suppose that G is a bipartite graph. The graph G has no perfect matching if and only if there exists some deficient set of vertices.

The Primal-Dual Algorithm (Hall's Theorem)



no perfect matching

This graph has no perfect matching.



|N(S)| = 1 < 2 = |S|

$S = \{a_1, a_2\}$ is deficient: $|\mathcal{N}(S)| = 1 < 2 = |S|$.

a_1 and a_2 must both be matched into $\mathcal{N}(S) = \{b_1\}$, but there is only one vertex there.

The Primal-Dual Algorithm

We can prove Hall's theorem using Farkas' Lemma!

Using our integrality theorem, G has no perfect matching if and only if the following polytope is empty

$$P = \{x \in \mathbb{R}^E : \sum_{e \in E : v \in e} x_e = 1 \text{ for all } v \in V,\ x \geq 0\}.$$

Farkas' lemma implies then that there must be some w so that $w_u + w_v \geq 0$ for each $uv \in E$ and so that $\sum_{v \in V} w_v < 0$.

The Primal-Dual Algorithm

If there is a set S which is deficient, then define $w_u = -1$ if $u \in S$, $w_v = 1$ if $v \in \mathcal{N}(S)$ and $w_v = 0$ in all other cases.

Let's see that w satisfies these conditions. Let $uv \in E$. If neither u nor v is in S, then w_u and w_v are both at least 0, and so $w_u + w_v \geq 0$.

If u or v is in S, then the other is in $\mathcal{N}(S)$ by definition. So, $w_u + w_v = 1 - 1 = 0$. Hence $w_u + w_v \geq 0$ for every edge.

Finally, $\sum_{v \in V} w_v = |\mathcal{N}(S)| - |S| < 0$. So w certifies that P is empty, and G has no perfect matching.

The Primal-Dual Algorithm

This shows the direction that if there is a deficient set, then there are no perfect matchings. The other direction can be shown with a more careful argument.

The Primal-Dual Algorithm

Returning to the algorithm, we are at a stage where we have a dual feasible y, but the equality edges $E^=$ contains no perfect matching.

By Hall's theorem applied to the graph with edge set $E^=$, there is a deficient subset $S \subseteq V_1$ (or V_2) of this graph.

We will update y by increasing y_v by some number $t > 0$ for every $v \in S$ and decreasing y_v by t for every $v \in \mathcal{N}(S)$, where $\mathcal{N}$ is taken in $E^=$. We choose t as large as possible while y is still feasible.

Note that this is the same as updating to $y - tw$, where w is the vector we constructed in the proof of one direction of Hall's theorem.

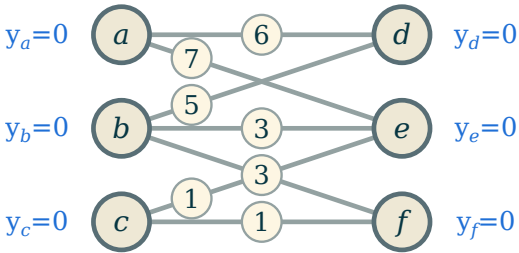
The Primal-Dual Algorithm

We have already seen that $w_u + w_v \geq 0$ for all $uv \in E^=$ and that $\sum_{v \in V} w_v < 0$. So the constraints of the equality edges stay satisfied under $y - tw$, and the constraints of all other edges are satisfied for small enough $t > 0$. Hence $y - tw$ is dual feasible for small $t > 0$.

The objective $\sum_v (y - tw)_v = \sum_v y_v - t \sum_v w_v$ increases as a function of t.

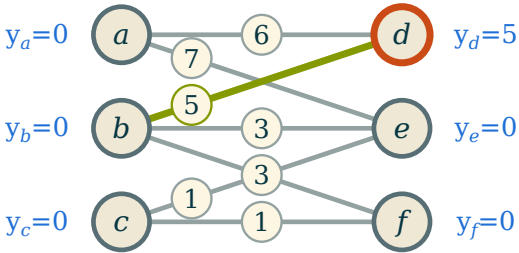
Therefore, we are making positive progress each time we update y.

The Primal-Dual Algorithm



Start with $y = 0$. This is dual feasible, since every cost is nonnegative.
$E^=$ is empty, so there is no perfect matching.
Initialization: increase the weights one vertex at a time, each until an edge at that vertex has reduced cost 0.

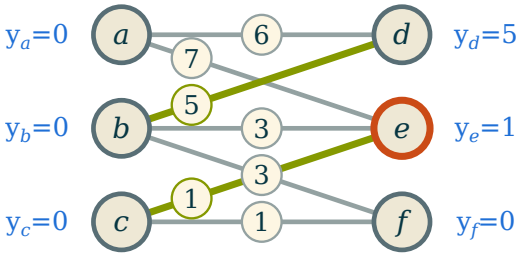
The Primal-Dual Algorithm



$$y_d = \min\{6, 5\} = 5$$

$$E^{=} = \{bd\}$$

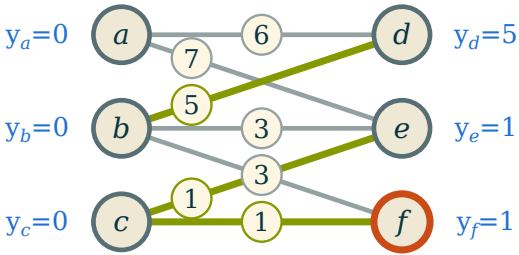
The Primal-Dual Algorithm



$$y_e = \min\{7, 3, 1\} = 1$$

$$E^{=} = \{bd, ce\}$$

The Primal-Dual Algorithm

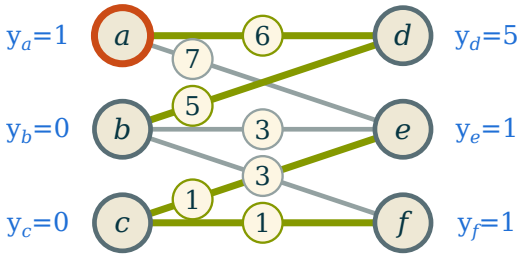


$$y_f = \min\{3, 1\} = 1$$

$$E^{=} = \{bd, ce, cf\}$$

a has no equality edge, so we can still increase y_a.

The Primal-Dual Algorithm

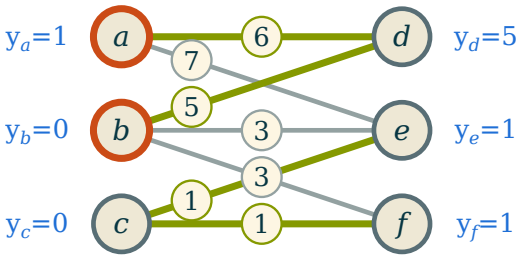


$$y_a = \min\{6 - 0 - 5,\ 7 - 0 - 1\} = 1$$

$$E^{=} = \{ad, bd, ce, cf\}$$

Every vertex now has an equality edge, so no single y_v can be increased. But there is no perfect matching, since a and b both need d.

The Primal-Dual Algorithm



$$E^{=} = \{ad, bd, ce, cf\}$$

$$S = \{a, b\}, \quad N_{E^{=}}(S) = \{d\}$$

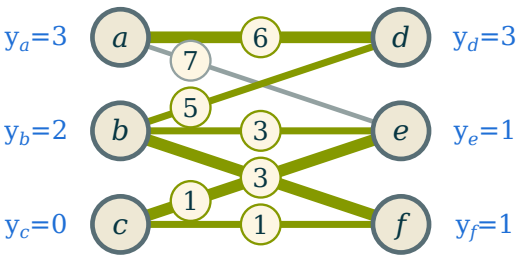
$$c_{ae} - y_a - y_e = 7 - 1 - 1 = 5$$

$$c_{be} - y_b - y_e = 3 - 0 - 1 = 2$$

$$c_{bf} - y_b - y_f = 3 - 0 - 1 = 2$$

$$\varepsilon = 2$$

The Primal-Dual Algorithm



$$y_a \leftarrow 1 + 2 = 3$$
$$y_b \leftarrow 0 + 2 = 2$$
$$y_d \leftarrow 5 - 2 = 3$$
$$E^{=} = \{ad, bd, be, bf, ce, cf\}$$
$$M = \{ad, bf, ce\}$$
$$c(M) = 6 + 3 + 1 = 10$$
$$\sum_w y_w = 3 + 2 + 0 + 3 + 1 + 1 = 10$$