Minimum Weight Matchings

Jan 14, 2009

1 Review

1.1 Definition: LP-duality

$$\min\{cx : Ax = b, x \geq 0\} = \max\{by : yA = c\}$$

1.2 Definition: Complementary Slackness

If x, y satisfy $Ax = b, x \geq 0, yA \leq c$ and $(yA)_i \leq c_i \Rightarrow x_i = 0$ then x, y are optima.

2 Problem

Given a graph $G(V, E)$ with weights c_e assigned to each edge $e \in E$ which has a perfect matching, find a perfect matching of minimum weight.

2.1 Integer Programming Formulation

$$\begin{aligned} &\min \sum x_e c_e \\ &\text{s.t. } x(\delta(v)) = 1 \forall v \in V, \text{ and} \\ &x_e \in \{0, 1\} \forall e \in E \end{aligned}$$

where $x(\delta(v))$ is defined as $\sum_{e \in \delta(v)} x_e$ and $\delta(v)$ is the set of edges incident to v.

2.2 Linear Program Relaxation

$$\begin{aligned} &\min \sum_{e \in E(G)} x_e c_e \\ &\text{s.t. } x(\delta(v)) = 1 \forall v \in V(G), \text{ and} \\ &x_e \geq 0 \forall e \in E(G) \end{aligned}$$

2.3 LP dual

$$\max \sum_{v \in V(G)} y_v, \text{ where}$$
$$y_u + y_v \leq c_e \forall uv \in E(G)$$

3 Hungarian Method

This algorithm, attributed to Kuhn or Jacobi, finds a minimum weight perfect matching for all bipartite graphs, and, more generally, all graphs not containing blossoms. The basic idea is to find a perfect matching M and a vector y s.t. we get equality for $x =$(the characteristic vector of M) and y. We start with some matching M, for instance $M = \varnothing$, and some vector y, for instance $y = -\infty \forall v \in V$.

3.1 Equality Edges and Equality Graph

$e = uv \in E$ is an equality edge if $y_u + y_v = c_e$. We want the number of equality edges to be non-decreasing. We want M to be a matching in the equality graph (to satisfy complementary slackness). How? First, we try to find an M-augmenting path in the equality graph. If we succeed, it means we can make a bigger matching. If there isn't one, we try to increase y to acquire a new equality edge. Doing this means we make progress by increasing the size of the equality graph. The following lemma means we can always do one or the other.

Lemma 1. *If G is bipartite (and contains no blossoms), and there is no M-augmenting path in the equality graph, then y can be increased so that a new edge becomes an equality edge.*

Proof. We may assume that every vertex is incident to an edge in the equality graph (for if any vertex is not, we increase its y until one of its incident edges is equality). We may also assume there is an unsaturated vertex r (or else no larger matching is possible). Take a maximal M-alternating tree in the equality graph rooted at r.

Now we want to increase y by some amount at vertices in $B(T)$ (those at even distance from r) and decrease it by same amount at vertices in $A(T)$ (those at odd distance from r. There is an edge in the graph that goes from a vertex in $B(T)$ to an edge not in T, because we assumed there was a perfect matching and no blossoms. We choose the amount to add and substract so that such an edge becomes an equality edge. □

4 Minimum weight matching for non-bipartite graphs

4.1 New LP relaxation

If G is non-bipartite (and blossoms exist), the LP relaxation can be rewritten as:

$$\begin{aligned}
&\min \sum_{e \in E(G)} x_e c_e \\
&\text{s.t.} x(\delta(v)) = 1 \forall v \in V(G), \\
&x_e \geq 0 \forall e \in E(G) \\
&X(C) \geq 1 \forall \text{ odd cuts } C.
\end{aligned}$$

An odd cut is a cut of the form $\delta(X)$, where X contains an odd number of vertices.

4.2 New LP dual

$$\begin{aligned}
&\max \sum_{v \in V(G)} y_v + \sum_{C \in \text{odd cuts}} y_c, \text{ where} \\
&y_u + y_v + \sum_{C \ni e} y_c \leq c_e \forall e = uv \in E(G), \text{ and} \\
&y_C >= 0 \forall \text{ odd cuts } C.
\end{aligned}$$

5 Method

Start with a perfect matching in equality graph. Let $y_v = -\infty$ and $y_C = 0 \forall v, C$. Apply the same strategy as in the Hungarian Method.

5.1 Handling Blossoms

Whenever a blossom B is encountered, construct $E(B)$ and modify c for edges in $d(V(B))$ (edges leaving $V(B)$) in the following way:

If $vw \in E(G), v \in V(B)$, $w \notin V(B)$ define

$$\begin{cases} c'_{vw} := c_{vw} - y_v & vw \in E(G), v \in V(B), w \notin V(B) \\ c'_e := c_e & \text{o.w.} \end{cases}$$

All parallel edges are retained after contraction.

As for the ys, $y'_v := 0$ for the new contracted vertex, and no others change.

This creates a new graph G' with $M' := M - E(B), c', y'$ to which we apply the algorithm recursively. We obtain a perfect matching M'' and vector y'' s.t.

M'' is a perfect matching in the equality graph w.r.t. y'' in G'. We construct M and y in G as follows:

M is obvious (same as Edmonds).

$y_v = y''v \forall v \in V(G') - \{n\}$, or just the old y_v otherwise.

$y_C = y''_n if C = \delta(V(B))$ or y''_C otherwise.

5.2 Properties of cuts

All cuts with y nonzero form a laminar family, so there are a linear number of nonzero y_Cs. (Only edges leaving blossoms are nonzero.)

6 Summary of algorithm

Start with M empty, $y_v = -\infty, y_c = 0$, and at least one equality edge incident to every vertex. Find a maximal M-alternating tree in equality graph. If there is a blossom, contract, apply recursively to a smaller graph. Obtain perfect matching in the equality graph of G' and use it to construct a perfect matching in the equality graph of G. This is a minimum weight perfect matching. We may assume no blossom. If there is an M-augmenting path, find a bigger matching. Otherwise y can be increased to acquire an equality edge.

Theorem 1 (Edmonds)**.** *The convex hull of incidence vectors of perfect matchings of a graph G is given by*

$$x(\delta(v)) = 1 \forall v \in V(G)$$
$$x_e \geq 0 \forall e \in E(G)$$
$$x(C) \geq 1 \forall \text{ odd cuts } C$$