CS 598CSC: Combinatorial Optimization	Lecture date: Feb 18, 2010
Instructor: Nitish Korula	Scribe: Abner Guzmán-Rivera

1 Maximum Weight Matching in Bipartite Graphs

In these notes we consider the following problem:

Definition 1 (Maximum Weight Bipartite Matching) *Given a bipartite graph $G = (V, E)$ with bipartition (A, B) and weight function $w : E \to \mathbb{R}$ find a matching of maximum weight where the weight of matching M is given by $w(M) = \sum_{e \in M} w(e)$.*

Note that without loss of generality, we may assume that G is a *complete* weighted bipartite graph (we may add edges of zero weight as necessary); we may also assume that G is balanced, i.e. $|A| = |B| = \frac{1}{2}|V|$, as we can add dummy vertices as necessary. Hence, some maximum weight matching is a perfect matching. Furthermore, by negating the weights of the edges we can state the problem as the following *minimization* problem:

Definition 2 (Minimum Weight Perfect Matching in Bipartite Graphs) *Given a bipartite graph $G = (V, E)$ with bipartition (A, B) and weight function $w : E \to \mathbb{R} \cup \{\infty\}$, find a perfect matching M minimizing $w(M) = \sum_{e \in M} w(e)$.*

We could also assume that no edge weights are negative as we may add a large enough constant C to all weights, but this is not required by the algorithms below.

The following is an ILP formulation of the minimum weight perfect matching problem:

$$\min \sum_{(a,b)} w(a,b)x(a,b) \text{ subject to:}$$

$$\begin{aligned} \sum_b x(a,b) &= 1 && \forall a \in A \\ \sum_a x(a,b) &= 1 && \forall b \in B \\ x(a,b) &\in \{0,1\} && \forall a \in A, b \in B \end{aligned} \tag{1}$$

Definition 3 (Primal) *This is the LP relaxation of the above ILP:*

$$\min \sum_{(a,b)} w(a,b)x(a,b) \text{ subject to:}$$

$$\begin{aligned} \sum_b x(a,b) &= 1 && \forall a \in A \\ \sum_a x(a,b) &= 1 && \forall b \in B \\ x(a,b) &\geq 0 && \forall a \in A, b \in B \end{aligned} \tag{2}$$

Recall that we saw, in an earlier lecture, a proof of the following theorem by noting that the constraint matrix of the polytope is totally unimodular.

Theorem 4 *Any extreme point of the polytope defined by the constraints in (2) is integral.*

We obtain a different proof of Theorem 4 via algorithms to find a minimum-weight perfect matching. Our algorithms are *primal-dual*; we will construct a feasible solution to the dual of LP (2) with value equal to the weight of the perfect matching output by the algorithm. By weak duality, this implies that the matching is optimal. More precisely, our algorithms will always maintain a feasible dual solution y, and will attempt to find a primal feasible solution (a perfect matching M) that satisfies complementary slackness.

(Dual) The following LP is the dual for (2):

$$\begin{aligned} &\textit{maximize} \sum_{(a\in A} y(a) + \sum_{b\in B} y(b) \text{ subject to:} \\ &y(a)+y(b) \quad \leq \quad w(a,b) \quad \forall (a,b)\in E \end{aligned} \tag{3}$$

Given a dual-feasible solution y, we say that an edge $e=(a,b)$ is *tight* if $y(a)+y(b)=w(a,b)$. Let $\hat{y}$ be dual-feasible, and let M be a perfect matching in $G(V,E)$: Then,

$$\begin{aligned} w(M) = \sum_{(a,b)\in M} w(a,b) \quad &\geq \quad \sum_{(a,b)\in M} \hat{y}(a)+\hat{y}(b) \\ &= \quad \sum_{a\in A} \hat{y}(a)\cdot(\delta(a)\cap M) + \sum_{b\in B} \hat{y}(b)\cdot(\delta(b)\cap M) \\ &= \quad \sum_{a\in A} \hat{y}(a) + \sum_{b\in B} \hat{y}(b) \end{aligned}$$

where the first inequality follows from the feasibility of $\hat{y}$, and the final equality from the fact that M is a perfect matching. That is, any feasible primal solution (a perfect matching M) has weight at least as large as the value of any feasible dual solution. (One could conclude this immediately from the principle of weak duality.) Note, though, that if M only uses edges which are *tight* under $\hat{y}$, we have equality holding throughout, and so by weak duality, M must be *optimal*. That is, given any dual feasible solution $\hat{y}$, if we can find a perfect matching M only using tight edges, M must be optimal. (Recall that this is the principle of *complementary slackness*.)

Our primal-dual algorithms apply these observations as follows: We begin with an arbitrary feasible dual solution y, and find a maximum-cardinality matching M that uses only tight edges. If M is perfect, we are done; if not, we *update* our dual solution. This process continues until we find an optimal solution.

We first give a simple algorithm (Algorithm 1 in the following page) exploiting these ideas to prove Theorem 4. The existence of set S in line 6 is a consequence of Hall's theorem. Observe that the value of y increases at the end of every iteration. Also, the value of y remains feasible as tight edges remain tight and it is easy to verify that by the choice of ϵ the constraints for other edges are not violated.

Claim 5 *Algorithm 1 terminates if w is rational.*

Proof: Suppose all weights in w are integral. Then at every iteration ϵ is integral and furthermore $\epsilon \geq 1$. It follows that the number i of iterations is bounded by $i \leq \max w(a,b)\cdot|E|$. If weights are rational we may scale them appropriately so that all of them become integers. □

Algorithm 1 MinWeightPerfectMatching($G = (V, E), w$)

$y \leftarrow 0$
$E' \leftarrow$ set of tight edges
$M \leftarrow$ max cardinality matching for graph $G' = (V, E')$
while M is not a perfect matching **do**
  let $G' = (V, E')$
  let $S \subseteq A$ be such that $|S| > |N(S)|$
  let $\epsilon = \min_{a \in S, b \in B \setminus N(S)} \{w(a, b) - y(a) - y(b)\}$
  $\forall a \in S \; y(a) = y(a) + \epsilon$
  $\forall b \in N(S) \; y(b) = y(b) - \epsilon$
  update E', M
end while
return M

Proof of Theorem 4. The incidence vector of a perfect matching computed by Algorithm 1 is an extreme point of the polytope in (2). This vector is integral. Furthermore, by carefully choosing the cost function one can make any extreme point be the unique optimum solution to the primal linear program. □

Note that Algorithm 1 does not necessarily terminate in strongly polynomial time; in the rest of this section, we describe a more efficient algorithm for the minimum-weight bipartite matching problem.

As before, Algorithm 2 always maintains a feasible dual y and attempts to find a close to primal feasible solution (matching M) that satisfies complementary slackness. One key difference from Algorithm 1 is that we now carefully use the maximum cardinality matching M as a guide in constructing the updated dual solution y; this allows us to argue that we can augment M efficiently. (In contrast, Algorithm 1 effectively "starts over" with a new matching M in each iteration.)

Algorithm 2 MinWeightPerfectMatchingPD($G = (V, E), w$)

$\forall b \in B \; y(b) \leftarrow 0$
$\forall a \in A \; y(a) \leftarrow \min_b \{w(a, b)\}$
$E' \leftarrow$ set of tight edges
$M \leftarrow$ max cardinality matching for graph $G' = (V, E')$
while M is not a perfect matching **do**
  let $E_{dir} \leftarrow \{e$ directed from A to $B \mid e \in E', e \notin M\} \bigcup$
    $\{e$ directed from B to $A \mid e \in E', e \in M\}$
  let $D = (V, E_{dir})$ $\{D$ is a directed graph$\}$
  let $L \leftarrow \{v \mid v$ is reachable in D from an unmatched vertex in $A\}$
  let $\epsilon = \min_{a \in A \cap L, b \in B \setminus L} \{w(a, b) - y(a) - y(b)\}$
  $\forall a \in A \cap L \; y(a) = y(a) + \epsilon$
  $\forall b \in B \cap L \; y(b) = y(b) - \epsilon$
  update E', M
end while
return M

Claim 6 *At every iteration,* $C = (A \setminus L) \cup (B \cap L)$ *is a vertex cover for graph* $G' = (V, E')$*. Moreover,* $|C| = |M|$*.*

Proof: Assume C is not a vertex cover. Then there must be an edge $e = (a, b) \in E'$ with $a \in A \cap L$ and $b \in B \setminus L$. If e is directed from a to b, then since a is reachable from an unmatched vertex in A, so is b; this contradicts the fact that $b \in B \setminus L$. Therefore, e must be directed from b to a, and hence e is in the matching M. As a itself is matched (using edge e) and $a \in L$, it must be reachable from an unmatched vertex of A. But the only incoming edge to a is (b, a) (this is the unique edge incident to a in the matching M), and hence b is reachable from this unmatched vertex of A; again, this contradicts the fact that $b \notin L$. To show the second part of the proof we show that $|C| \leq |M|$, since the reverse inequality is true for any matching and any vertex cover. The proof follows from the following observations:

1. No vertex in $A \setminus L$ is unmatched by the definition of L.
2. No vertex in $B \cap L$ is unmatched since this would imply the existence of an augmenting path (contradicting the maximality of M).
3. There is no edge $e = (a, b) \in M$ such that $a \in A \setminus L$ and $b \in B \cap L$. Otherwise, as this edge would be directed from b to a, a would be in L.

These remarks imply that every vertex in C is matched and moreover the corresponding edges of the matching are distinct. Hence $|C| \leq |M|$, and so C is an optimum vertex cover for $G'(V, E')$. □

At every iteration where the maximum cardinality matching M output is not perfect, the algorithm will use information from the optimum vertex cover C to update the dual solution and improve its value. By the proof of claim 6 there is no tight edge between $a \in A \cap L$ and $b \in B \setminus L$, which implies $\epsilon > 0$; it is easy to check that the updated dual solution is feasible. Moreover, the difference between the new dual solution and the old dual solution is:

$$\epsilon \cdot (|A \cap L| - |B \cap L|) = \epsilon \cdot (|A \cap L| + |A \setminus L| - |A \setminus L| - |B \cap L|) = \epsilon \cdot (\frac{|V|}{2} - |C|),$$

but $|C| = |M| < \frac{|V|}{2}$, since M is not perfect, which implies the value of the dual solution strictly increases. When the algorithm terminates, we obtain a perfect matching M and a dual feasible solution which satisfy complementary slackness.

Claim 7 *Algorithm (2) terminates in* $O(|V|^2)$ *iterations.*

Proof: We first observe that after any iteration, all edges in M are still tight: The only edges (a, b) that are tight at the beginning of an iteration but not at the end are those with $a \in A \cap L$ and $b \in B \setminus L$; from observation 3 in the proof of Claim 6, there are no edges in M of this form. Thus, after any iteration, the size of a maximum cardinality matching M in $G'(V, E')$ cannot decrease.

Say that an iteration is *successful* if the size of a maximum cardinality matching using the tight edges E' increases. Clearly, after at most $|V|/2$ successful iterations, we have a perfect matching, and the algorithm terminates. We show that there are at most $|B| = |V|/2$ consecutive unsuccessful iterations between any pair of successful iterations. Hence, the total number of iterations is at most $\frac{|V|}{2} \cdot \frac{|V|}{2}$, which is $O(|V|^2)$.

To bound the number of consecutive unsuccessful iterations, we argue below that after an unsuccessful iteration, $|B \cap L|$ increases. Assume for now that this is true: After at most $|B|$ unsuccessful iterations, we have $B \cap L = B$. Once this occurs, every vertex of B (which must include at least one unmatched vertex) is reachable from an unmatched vertex of A, and so we can augment M to find a larger matching, which means that the current iteration is successful.

It remains only to prove that at every unsuccessful iteration, at least one more vertex in B must become reachable from an exposed vertex in A (i.e. $|B \cap L|$ increases). First note that no vertex of A or B becomes unreachable; the only way this could happen is if for some path P from an unmatched vertex $a \in A$ to vertex $v \in L$, an edge $e \in P$ that was previously tight is no longer tight. But the only edges that are no longer tight are between $A \setminus L$ and $B \cap L$, and by definition, no such path P visits a vertex in $A \setminus L$. To see that at least one new vertex of B becomes reachable, note that some edge $e = (a, b)$ with $a \in A \cap L$ and $b \in B \setminus L$ now has become tight by our choice of ϵ. As the edge (a, b) is directed from a to b, b is now reachable. □

It is not hard to see that each iteration takes only $O(|V|^2)$ time, and hence the overall running time of the algorithm is $O(|V|^4)$. A more careful analysis would yield a tighter running time of $O(|V|^3)$.

References

[1] A. Schrijver. *Combinatorial optimization: Polyhedra and Efficiency*, Springer, 2003. Chapter 17.

[2] Lecture notes from Michael Goemans class on Combinatorial Optimization. `http://math.mit.edu/~goemans/18433S09/matching-notes.pdf`, 2009.