Algorithms – handout 3

Matchings

April 28, 2014

Bipartite matchings are intersections of two matroids, but are not matroids themselves. We show how to find maximum cardinality matchings in bipartite graphs. The algorithm uses the notion of *alternating paths*. Similar to greedy algorithms, it constructs a sequence of partial solutions, each larger than the previous one. However, unlike greedy algorithms, when extending from one solution to another, it retracts some of the decisions of the former solution.

More generally, we show an algorithm for finding maximum cardinality matchings in general graphs. This will lead to the Gallai-Edmonds decomposition of a graph.

We show an algorithm for finding minimum weight perfect matchings in complete bipartite graphs. It is a *primal-dual* algorithm, based on the following theorem of Egevary (from 1931).

Theorem: Let $G(U \cup V, E)$ be a complete bipartite graph with $|U| = |V|$, and with edge weights $w : E \longrightarrow R$. A cost function $c : U \cup V \longrightarrow R$ for the vertices is *feasible* if for every edge (u, v), $c(u) + c(v) \leq w((u, v))$. Then the weight of the minimum weight perfect matching is equal to the maximum of $\sum_{v \in U \cup V} c(v)$ over all feasible cost functions.

Homework: (hand in by May 19, 2014)

1. Hall's theorem states that a bipartite graph $G(U \cup V, E)$ has a perfect matching iff for every set H of vertices entirely contained in one side (namely, either $H \subset U$ or $H \subset V$) one has $|N(H)| \geq |H|$, where $N(H)$ is the set of neighbors of H. Use the Gallai-Edmonds decomposition for bipartite graphs to derive Hall's theorem.

2. Show that every bipartite d-regular graph is the disjoint union of d perfect matchings.

3. Vizing's theorem says that the edges of every simple graph (no parallel edges or self loops) can be partitioned into $\Delta+1$ matchings, where Δ denotes the maximum degree. Show that for bipartite graphs, Δ matchings suffice.

4. Show that the problem of finding a maximum weight matching in a bipartite graph can be reduced in polynomial time to the problem of finding a minimum weight perfect matching in a complete bipartite graph. (Hint: change the original graph by adding new vertices and edges, and by modifying weights of edges.)