CS 473: Fundamental Algorithms, Fall 2011

Discussion 12

November 8, 2011

12.1 Maximum Flow By Scaling

Let $G = (V, E)$ be a flow network with source s, sink t, and an integer capacity $c(u, v)$ on each edge $(u, v) \in E$. Let $C = max_{(u,v)\in E} c(u, v)$.

(A) Argue that a minimum cut of G has capacity at most $C|E|$.

(B) For a given number K, show that an augmenting path of capacity at least K can be found in $O(E)$ time, if such a path exists.
The following modification of Ford-Fulkerson-Method can be used to compute a maximum flow in G.

```
Max-Flow-By-Scaling(G, s, t)
1     C ← max_{(u,v)∈E} c(u, v)
2     initialize flow f to 0
3     K ← 2^{⌊lg C⌋}
4     while K ≥ 1 do {
5          while (there exists an augmenting path p of
                              capacity at least K) do {
6               augment flow f along p
           }
7           K ← K/2
      }

8     return f
```

(C) Argue that **Max-Flow-By-Scaling** returns a maximum flow.

(D) Show that the capacity of a minimum cut of the residual graph G_f is at most $2K|E|$ each time line 4 is executed.

(E) Argue that the inner **while** loop of lines 5-6 is executed $O(E)$ times for each value of K.

(F) Conclude that **Max-Flow-By-Scaling** can be implemented so that it runs in $O(E^2 \lg C)$ time.

12.2 k-Regular Bipartite Graphs.

A k-Regular graph is an undirected graph where every vertex has degree k. We will prove that if a bipartite graph is k-Regular, then it has a perfect matching. First, recall the following definitions:

- ***Bipartite Graph***: a graph whose vertices are partitioned into two independent sets, L and R.
- ***Matching***: A matching in a graph G is a set of edges such that no two edges share a common vertex.
- ***Neighbors***: Let v be a vertex. The neighbors of v, denoted by $N(v)$ are the set of vertices connected to v.
- ***Hall's theorem***: Let $G = (L \cup R, E)$ be a bipartite graph where $|L| = |R|$. Then G has a perfect matching if and only if for every subset $X \subseteq L$, $|N(X)| \geq |X|$.

For the following problems, let $G = (L \cup R, E)$ be a k-regular bipartite graph where $|L| = |R|$.

(A) Show that G has a perfect matching via Hall's theorem.

(B) Now, construct a flow network G' from G such that the value of the maximum flow in G' is equal to the size of the perfect matching in G.

(Note, that one can argue here that there is a fractional flow of value $|L|$ and thus implying (A) in this case.)

12.3 Dinner Scheduling.

Consider a group of n people who are trying to figure out a dinner schedule over the next n nights where each person needs to cook exactly once. Everyone has scheduling conflicts with some of the nights, so deciding who should cook on which night becomes tricky.

Label the people $\{p_1, \ldots, p_n\}$ and the nights $\{d_1, \ldots, d_n\}$. For each person p_i, there's a set of nights $S_i \subset \{d_1, \ldots, d_n\}$ when they are *not* able to cook.

A *feasible dinner schedule* is an assignment of each person to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j, then $d_j \notin S_i$.

(A) Describe a bipartite graph G so that G has a perfect matching if and only if there is a feasible dinner schedule for the group. What is the running time of your algorithm in this case?

(B) After generating a schedule, they realize there is a problem. $n - 2$ of the people are assigned to different nights on which they are available: no problem there. However, for the other two people p_i and p_j, and the other two days, d_k and d_l, both p_i and p_j are assigned to cook on night d_l. Show that it's possible to fix this bad assigned and get a good assigned faster than just computing a solution from scratch. Namely, decide in $O(n^2)$ time, given this bad solution, whether there exists a feasible dinner schedule. How does the running time of your algorithm compares to (A).