CSCE 629-601 Analysis of Algorithms

Fall 2022

Instructor: Dr. Jianer Chen
Office: PETR 428
Phone: (979) 845-4259
Email: `chen@cse.tamu.edu`
Office Hours: MWF 3:50 pm - 5:00 pm

Teaching Assistant: Vaibhav Bajaj
Office: EABC 107B
Phone: (979) 739-2707
Email: `vaibhavbajaj@tamu.edu`
Office Hours: T,TR 2:00 pm - 3:00 pm

Solutions to Assignment # 5

1. A vertex v in an undirected graph G is an *odd cycle transversal* if every cycle of odd length in G contains the vertex v. Develop a linear-time algorithm for the following problem: given a graph G and a vertex v in G, decide if v is an odd cycle transversal.

Solutions. A graph is bipartite if and only if it does not contain an odd cycle. We have shown in class that using DFS, we can decide in linear-time whether a given graph is bipartite.

To solve the problem given in the question on a graph G, we first decide in linear time whether G is bipartite. If G is bipartite, then v is not an odd cycle transversal because G does not contain any odd cycles. Otherwise, remove v from the graph G to obtain a graph G', which can be done in linear time. Now decide in linear-time whether the graph G' is bipartite. If G' is bipartite, then the vertex v is an odd cycle transversal in the graph G; otherwise, v is not because there are odd cycles in the graph G' that does not contain the vertex v. Combining the above steps, we have a linear-time algorithm that solves the problem given in the question.

2. Suppose that each class C_i has an enrollment r_i while each classroom R_j has a capacity c_j. A classroom R_j is "feasible" for a class C_i if $c_j/2 \leq r_i \leq c_j$. Develop an efficient algorithm that, on a set of classes (with enrollments given) and a set of classrooms (with capacities given), make a feasible assignment of the classes to the classrooms such that as many classes as possible can get held starting at 9am on Monday.

Solutions. Let $C = \{C_1, C_2, \ldots, C_m\}$ be the given set of m classes, where each class C_i is associated with an enrollment r_i, and let $R = \{R_1, R_2, \ldots, R_n\}$ be the given set of n classrooms, where each classroom R_j is associated with a capacity c_j.

First, we construct a bipartite graph $G = (U, V, E)$ as follows: let $U = \{u_1, u_2, \ldots, u_m\}$ and $V = \{v_1, v_2, \ldots, v_n\}$, where for each i, u_i corresponds to the class C_i, and for each j, v_j corresponds to the classroom R_j. There is an edge between u_i and v_j if $c_j/2 \leq r_i \leq c_j$. Thus, the graph G has at most mn edges, and constructing the graph G takes time $O(mn)$.

Now use the algorithm discussed in class to construct the maximum matching M in the graph G. Obviously, M is the desired optimal feasible assignment.

The maximum matching algorithm runs in time $O(n_1 m_1)$ on a graph of n_1 vertices and m_1 edges. Since the graph G has $n_1 = |U| + |V| = n + m$ vertices and $m_1 \leq mn$ edges, the above algorithm solving the problem runs in time $O(n_1 m_1) = O(n^2 m + nm^2)$.

3. Suppose that in addition to edge capacities, a flow network also has *vertex capacities*, i.e., each vertex v has a limit $c(v)$ on how much flow can pass through v. Show how to transform a flow network $G = (V, E)$ with vertex capacities into a flow network $G' = (V', E')$ without vertex capacities, such that a maximum flow in G' has the same value as a maximum flow in G.

Solutions. Let $G = (V, E)$ be a flow network with vertex capacities (in addition to edge capacities). Let s and t be the source and sink in the flow network G, respectively. We construct an "equivalent" flow network $G' = (V', E')$ without vertex capacities such that a maximum flow in G' has the same value as that of a maximum flow in G.

The flow network G' is constructed from the flow network G, as follows. For every vertex v with vertex capacity $c(v)$ in G, "split" v into two vertices v_1 and v_2, and add an edge $[v_1, v_2]$ with edge capacity $c'(v_1, v_2) = c(v)$. For every edge $[u, v]$ in G, where the vertex u is split into $[u_1, u_2]$ and v is split into $[v_1, v_2]$, replace $[u, v]$ with the edge $[u_2, v_1]$ with capacity $c'(u_2, v_1) = c(u, v)$. This gives the flow network $G' = (V', E')$ without vertex capacities. The source of G' is s_1 while the sink of G' is t_2. It is easy to see that $|E'| = |V| + |E|$ and $|V'| = 2|V|$.

Let f be a flow from s to t in the flow network G. We create a flow f' in the flow network G' as follows: for each edge $[u, v]$ in G, $f'(u_2, v_1) = f(u, v)$, and for each vertex $v \neq s$ in G, let $f'(v_1, v_2) = \sum_{[u,v] \in E} f(u, v)$. Moreover, let $f'(s_1, s_2) = \sum_{[s,u] \in E} f(s, u)$. It is easy to verify that f' is a valid flow in G' and its value is equal to that of the flow f in G.

Conversely, given a flow f' in the flow network G', we can construct a flow f in the flow network G as follows: for each edge $[u, v]$ in G, let $f(u, v) = f'(u_2, v_1)$. Since f' satisfies the capacity constraint, in particular the capacity constraints on the edges $[v_1, v_2]$ where v_1 and v_2 correspond to a vertex v in G, f satisfies both edge capacity constraints and the vertex capacity constraints in G. Thus, f is a valid flow in the flow network G with the same value as f'.

Therefore, there is a one-to-one correspondence between flows in G and flows in G' where the corresponding flows have the same value. Thus, the maximum flow value in G equals the maximum flow value in G'.

4. (Textbook, page 731, Question 26.2-10) Show how to find a maximum flow in a flow network $G = (V, E)$ by a sequence of at most $|E|$ augmenting paths. (*Hint*: determine the paths *after* finding the maximum flow.)

Solutions. First, find a maximum flow f in G. There are no more than $|E|$ edges with flow value larger than 0. Let $\mathcal{P} = \emptyset$. Repeat the following procedures until $f(u, v) = 0$ for all edges $[u, v]$:

1. construct a subgraph $G' = (V', E')$ of G that contains the edges $[u, v] \in E$ with $f(u, v) > 0$;
2. select an edge $[u', v']$ in G' with the minimum flow value $f(u', v')$. There must be an s-t path in G' that contains $[u', v']$ because there is a positive flow going through the edge $[u', v']$;
3. let P be a path from s to t in G' that contains the edge $[u', v']$. Since $[u', v']$ has the minimum f value, we can push a flow along the path P that saturates the edge $[u', v']$. Thus, on each edge $[x, y]$ on the path P, we push a flow of value $f(u', v')$ so that the remaining flow value on the edge $[x, y]$ becomes $f(x, y) - f(u', v')$. The path P is an augmenting path. Add P into $\mathcal{P}$. In particular, the flow value along the edge $[u', v']$ becomes 0.

In each iteration, at least one edge has its flow value become 0. Since there are at most $|E|$ edges whose flow value are greater than 0, the number of iterations is $|E|$. It follows that $\mathcal{P}$ contains at most $|E|$ augmenting paths.