CSCI 314 – Fall 2012 Handout: asgn08
Algorithms
Michael H. Goldwasser
Saint Louis University Tuesday, 6 November 2012

Homework #8: Network Flow
Due Date: Friday, 16 November 2012

Guidelines

Please make sure you adhere to the policies on collaboration and academic honesty as outlined in the syllabus.

Reading

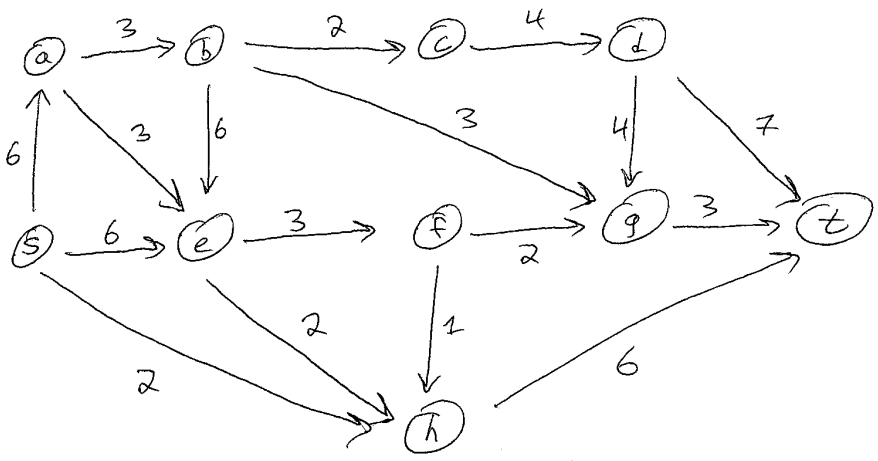
Chapter 26

Problems

Problem A (25 points)

"Work entirely on your own."

Consider the following graph:



You are to simulate the Edmonds-Karp algorithm for computing the maximum flow from s to t. To succinctly describe the process, list each augmenting path that is used, in the order considered by the algorithm, including the sequence of nodes on the path (e.g., `sefgt`) and the amount of flow that is augmented along that path.

For problems B, C, and D, we wish to explore the use of network flow as a model to solve a variety of practical problems. For each of the following, you are to clearly describe how the given problem can be reduced to an s-t network flow computation. Your submitted solutions must specifically detail:

- a general way to transform an instance of the given problem into a relevant directed graph G, including:
  - how the vertices of G are defined,
  - what vertex serves as source s and as sink t,
  - what edges exist in G,
  - how the capacity for each edge is defined.
- how knowledge of the maximum flow from s to t in G can be used to compute the desired result for the original problem instance.

So long as your provide a clear explanation of the reduction, you are not expected to provide any formal proofs in your responses.

Problem B (25 points)

"You may discuss ideas with other students."
In the traditional formulation of the network flow problem, there are capacity constraints on *edges* of the graph. In this problem, we wish to consider the goal of maximizing the flow from selected s to t, for directed graph G with *vertex* constraints. That is, for each vertex $v \notin \{s, t\}$, we have a capacity $c(v)$ such that at most $c(v)$ units of flow pass through vertex v. However, there are no explicit capacity constraints on the edges. The goal is to maximize the flow from s to t.

Problem C (25 points)

"You may discuss ideas with other students."
Whenever the President has an event in Washington D.C., the secret service is very concerned in making sure, not just that there is a clear path from that event back to the white house, but that there are many independent such paths. We define independence as follows.

We represent the map of the city as an *undirected* graph G, with nodes representing intersections and edges representing portions of streets that connect those intersections. An "escape" path is one that goes from the event location back to the White House. A set of escape paths are considered independent if no two of them share any road segment. Note well that we consider two paths to be in conflict even if travel across a shared segment takes place in opposite directions. However, independent paths are allowed to pass through the same intersection point.

Problem D (25 points)

"You may discuss ideas with other students."

You are in charge of creating a schedule for the employees of your store. You have k employees who work at the store, and you have s shifts that must be covered during the week, with a requirement that there be two employees for each shift. Furthermore, you have created a survey in which each employee has indicated which shifts he or she would be able to cover, and an absolute limit on the maximum number of shifts that he or she is willing to work.

Your goal is to produce a feasible assignment of employees to shift respecting the employee's constraints (or to determine that it is impossible).

Problem E **(EXTRA CREDIT – 10 points)**

"You may discuss ideas with other students."

Assume that you have a network that is designed to move k units of flow from source s to destination t, with each edge having capacity 1. An attacker is able to destroy the $s - t$ connectivity of your network by destroying precisely k edges of the network (by destroying any minimum $s - t$ cut of the graph).

After the attack, your job is to report back to your boss the complete set of all nodes that are unreachable from s. As a tool, you are allowed to call PING(V) for any vertex v, to determine if that vertex is currently reachable from s. So clearly, you could use $|V|$ ping calls to analyze the connectivity. However, we wish for you to develop a more efficient algorithm.

Give an algorithm that determines all nodes that are unreachable from s using only $O(k \log |V|)$ pings, based on the assumption that precisely k edges of the network have been destroyed. Your algorithm is allowed to decide which node to ping next based on the outcome of earlier ping operations.