Patrick Trinkle

CMSC641

Spring 2008

Homework 5

19/30

1. Exercise 26.2-8 5/10

2. Problem 26-1 7/10

3. Problem 26-2 7/10

1. Exercise 26.2-8

Show that a maximum flow in a network G = (V, E) can always be found by a sequence of at most |E| augmenting paths. (Hint: Determine the paths *after* finding the maximum flow.)

2. Problem 26-1 Escape Problem

An n x n grid is an undirected graph consisting of n rows and n columns of vertices, as shown in Figure 26.11. We denote the vertex in the ith row and the jth column by (i,j). All vertices in a grid have exactly four neighbors, except for the boundary vertices, which are the points (i,j) for which i = 1, i = n, j = 1, or j = n. Given m ≤ n2 starting points (x1, y1), (x2, y2), …, (xm, ym) in the grid, the escape problem is to determine whether or not there are m vertex-disjoint paths from the starting points to any m different points on the boundary. For example, the grid in Figure 26.11(a) has an escape, but the grid in Figure 26.11(b) does not.

1. Consider a flow network in which vertices, as well as edges, have capacities. That is, the total positive flow entering any given vertex is subject to a capacity constraint. Show that determining the maximum flow in a network with edge and vertex capacities can be reduced to an ordinary maximum-flow problem on a flow network of comparable size.
2. Describe an efficient algorithm to solve the escape problem, and analyze its running time.

3. Problem 26-2 Minimum path cover

A path cover of a directed graph G = (V, E) is a set P of vertex-disjoint paths such that every vertex in V is included in exactly one path in P. Paths may start and end anywhere, and they may be of any length, including 0. A minimum path cover of G is a path cover containing the fewest possible paths.

1. Give an efficient algorithm to find a minimum path cover of a directed acyclic graph G = (V, E). (Hint: Assuming that V = {1, 2, …, n}, construct the graph G’ = (V’, E’), where

V’ = {x0, x1, …, xn} ∪ {y0, y1, …, yn},

E’ = {(x0, xi) : i ∈V} ∪ {(yi, y0) : i ∈ V} ∪ {(xi, yj) : (i, j) ∈ E},

and run a maximum-flow algorithm.)

1. Does your algorithm work for directed graphs that contain cycles? Explain.

Answers

1.

Either f(e) = 0 ∀e ∈E or there is remaining flow. There must be some augmenting path p else f(e) = 0. Augmenting along p results in G’ with f’ as flow. f’ ≥ f by 1 more edge. This can only be repeated at most once per edge. Therefore only |E| times => max flow f in G can be found by at most |E| augmenting paths.

“why? I am not able to see your argument.”

2a.

To make into a regular flow problem:

Set vertices to have capacity 1, so they can only be touched once. Set all edges to capacity 1, because no edges can be reused. Add a supersource and supersink. Each m vertex has an edge to every non-starting point neighbor. If flow is m, then you know escape is possible.



“Read the problem again” – Correct, I misread the problem slightly as I didn’t quite transform it into an “ordinary” maximum flow problem.

2b.

From each mij build a graph G’ = (V’, E’) s.t. V’ = V – {M} and E’ = edges not to vertices in M. M is the set of all starting point vertices. Build the graph by the following method: Build a set of graphs with roots mi ∈M, walk out in each possible direction and add edges and vertices to the graph:

if the vertex is not another starting point,

if the edge does not touch another starting point,

and if the vertex found is in another graph in the set

In the case of the vertex being found in another graph—link the current graph from at that point to the graph wherein the vertex was found. This way only each edge and vertex will be examined once. Set all capacities, on vertex and edge to 1. This all takes O(E+V) time. Attach a supersource s, to each mi source, and a supersink t to each leaf of the graph set. This takes O(V). Run a maximum flow algorithm (choose one that runs in O(V3). If max flow = number of included\* starting points—then all can escape.

\*exclude all mi on boundaries in the graph.

O(2V + E + V3) = O(V+E+V3)

“Your graph construction is not clear to me” – Essentially the “set of graphs” is a group of graphs all with roots as the starting points. These graphs are independent unless the point can take the same escape path; if this is possible then the graphs are joined at that vertex. When graphs are joined the set is reduced by 1 graph. It is possible that the set of graphs will only contain 1 graph.



3a.

The hint describes a method of transforming the graph G into a network flow problem. Here is an algorithm for doing this to find a min path cover of G = (V, E).

Define G’ = (V’, E’) s.t. V’ = {x0, x1, …, xn} ∪ {y0, y1, …, yn} and E’ = {(x0, xi) : i ∈ V} ∪{(yi, y0) : i ∈V} ∪ {(xi, yj) : (i, j) ∈ E}.

Create a vertex s and add an edge from s to every xi ∈ V’. Give this edge capacity 1. Create another vertex t, linking all yi ∈ V’ to t, with capacity 1. Then add all edges in E’ s.t. they connect xi and yi with capacity 1. Run a maximum flow algorithm. Take each edge between xi and yi which is saturated and make a new G’’ = (V, E’’) where there is only 1 edge entering or exiting a vertex. The minimum path cover is comprised of the edges of G’’. E’’ = saturated edges in G’.

“Proof of correctness” – Stuff… I just need to write a proof that including the saturated edges will be a minimum path cover.

3b.

Doesn’t work with cycles because we might choose them for E’’; and they would be unnecessary for a minimum path cover.

“Explain your answer clearly. An example here would greatly help.”