CS 170 Algorithms Fall 2014 David Wagner HW 5

Due Oct. 3, 6:00pm

Instructions. This homework is due Friday, October 3rd, at 6:00pm electronically. Same rules as for prior homeworks. See `http://www-inst.cs.berkeley.edu/~cs170/fa14/hws/instruct.pdf` for the required format for writing up algorithms.

1. (10 pts.) Super-long path in a DAG
Design a linear-time algorithm for the following task:

Input: A directed acyclic graph G

Question: Does G contain a directed path that touches every vertex exactly once?

2. (20 pts.) Number of shortest paths
Often there are multiple shortest paths (all of the same length) between two nodes of the same graph. Design an efficient algorithm to count the number of shortest paths from vertex s to vertex t in a directed graph G (possibly containing cycles; all edges are of length 1).

Hint: throw away some of the edges, then you have a...

3. (25 pts.) The farmer and the river
A farmer has a wolf, sheep, and a cabbage. Initially they are all on the west bank of a river; he wants to transport them all to the east bank of the river. He can carry at most one of them in his boat at a time. Unfortunately, if he leaves the wolf and the sheep alone on the same bank, the wolf will eat the sheep; if he leaves the sheep and the cabbage alone, the sheep will eat the cabbage. In each step, the farmer can cross the river in his boat from west to east or from east to west.

(a) Find a way to get them all to the east bank, in the least number of crossings, by forming a graph and then doing something with the graph. Draw your graph. How many steps does the shortest solution require?

(b) How many different such solutions (with that number of crossings) are there?

4. (20 pts.) Equality contraints
Here's a problem that shows up in program analysis. Suppose we have a list of variables $x_1, x_2, \ldots, x_n$ (over the real numbers). We are given some equality constraints, of the form $x_i = x_j$, and some disequality constraints, of the form $x_i \neq x_j$. Design an efficient algorithm to determine whether it is possible to satisfy them all, when we are given m constraints over n variables.

For instance, the constraints $x_1 = x_2$, $x_2 = x_3$, $x_1 \neq x_3$ cannot be simultaneously satisfied, so your algorithm should output "No" on that input.

5. (25 pts.) Telephone keypad entry
Amalgamated Systems Inc. is building flip-phones (so 80's!) and wants to develop a system to make it easy for people to enter in text messages via the numeric keypad. The usual solution is to map letters to sequences

of numbers like this: A=2, B=22, C=222, D=3, E=33, F=333, G=4, etc. However it's annoying to have to type such a long sequence of numbers. Their marketing folks have come up with a great idea[1]: on their phone, they'll use the mapping A=2, B=2, C=2, D=3, E=3, F=3, G=4, etc.

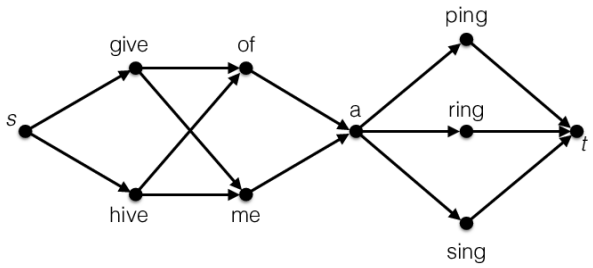
The problem, of course, is that some words are ambiguous. For example, 22737 could represent CARDS, CAPER, or a number of other English words. The company is counting on you to invent some software that somehow uses context to infer what the user was typing. Assume that the user will be typing English sentences, where all of the words are properly spelled. We'll use the * key to represent a space.

You are given a dictionary of all English words and the frequency of each of them: for each English word w, we are given $p(w)$, which represents the probability that any particular word will be w (i.e., the relative frequency of w in English text). You are also given frequency information about which words appear next to each other: for each pair w,x of English words, we are given $p(x|w)$, which represents the conditional probability that the next word is x, given that the current word is w. For instance, $p(\text{THE})$ is large (close to 1), because THE is a common word, while $p(\text{MALAPROPISM})$ is small (close to 0), because MALAPROPISM is rarely used in normal English. $p(\text{SKELTER}|\text{HELTER})$ might be very large (close to 1), while $p(\text{GIRAFFE}|\text{SMIRKING})$ might be very small (close to 0).

You are given a number sequence, e.g., 4483*63*2*7464. Your goal is to output the sequence of English words that has the highest total probability. The total probability of a sequence $w_1, w_2, \ldots, w_k$ of words is

$$p(w_k|w_{k-1}) \cdots p(w_3|w_2)p(w_2|w_1)p(w_1).$$

For any particular number sequence, we can form a directed graph; below we show the graph corresponding to 4483*63*2*7464. The graph has one source node s and one sink node t. The ith level of the graph has one vertex for each possibility for the ith word, and there is an edge from each vertex at level i to each vertex at level $i+1$. Each path from s to t in the graph below provides a possible decoding of 4483*63*2*7464.



We want to find the path that corresponds to a word sequence whose total probability is highest. In this case, based upon word statistics, the decoding with the highest probability is GIVE ME A RING.

Design an efficient algorithm to solve the following task:

Input: The directed graph G obtained from the number sequence (e.g., the graph shown above), and the probabilities $p(\cdot)$ and $p(\cdot|\cdot)$

Output: The sequence of words $w_1, w_2, \ldots, w_k$ that maximizes $p(w_k|w_{k-1}) \cdots p(w_3|w_2)p(w_2|w_1)p(w_1)$, out of all possible decodings consistent with the graph.

Try to find an algorithm whose running time is $O(|V| + |E|)$. If you cannot find a linear-time algorithm, we will accept one that is slower by a log-factor.

[1] It's such a great idea that it's been deployed in practice, under the name T9.

6. (5 pts.) Optional bonus problem: Golden graphs

(This is an *optional* bonus challenge problem. Only solve it if you want an extra challenge.)

Let $G = (V, E)$ be a graph where each edge is labelled with a non-negative length. Suppose we are given a set $E_g \subseteq E$ of "golden" edges. We call a path "golden" if at least k of its edges are golden. Let $d_g(s,t)$ denote the length of the shortest golden path from s to t. Describe an efficient algorithm to solve the following problem:

Input: A graph $G = (V, E)$, with non-negative edge lengths $\ell(u,v)$; a set $E_g \subseteq E$; and an integer $k \geq 0$

Output: $d_g(s,v)$ for each $v \in V$