

This and other notes are available from <http://www.cs.berkeley.edu/~yozo/cs170.fa05/>

Midterm Questions

- Problem 3a

Note $e^{cx^2} \neq e^c e^{x^2}$.

- Problem 5 (Visiting Each Vertex Exactly Once in DAGs)

Suppose there is a path $v_1 - v_2 - \dots - v_n$. What does this imply about the topological sort of G ? Converse?

- Problem 7 (Maximum Path-Bandwidth Algorithm)

For any path p , the path-bandwidth of p was defined to be the weight of the minimum edge in the path. The question asked to devise an algorithm that computes the maximum path-bandwidth between two (given) vertices s and t .

Here is a similar problem for comparison:

Shortest Bottleneck Problem. For any path p , define the *bottleneck* of the path to be the weight of the maximum edge in path p . Devise an algorithm that computes the shortest bottleneck between two given vertices s and v .

Few things to note:

1. One is a maximization of a minimum, while the other is a minimization of a maximum. Which is which?
 2. How are all these common to Dijkstra's Algorithm?
What property of these problems allows us to use Dijkstra?
 3. What about negative edges?
 4. What about cycles? Negative cycles?
 5. A lot of you gave a solution based on DFS. Explain why a simple DFS 'solution' does not work.
- Problem 8 (Shortest S - T Path).

We want to find the shortest path from any node in S to any node in T .

1. Consider all possible path from S to T . They all start from some node in S . How can we change this so that it start from a single node? (*Hint*: the single node need not be in G).
2. Consider the following modification: change all edges entirely inside S to have weight 0. Does this work? What if there was no negative weights?

Lempel-Ziv Codes

```
1 Start with a table with just null string in it (with
  index 0).
2 while INPUT is not empty do
3   Find the longest prefix  $\sigma$  matching the table entry.
4   Read the next character  $c$ .
5   Output  $I(\sigma), c$ .
6   Add  $\sigma c$  to the table.
7 end
```

Algorithm 1: LEMPEL-ZIV

How do you decode a Lempel-Ziv encoded string? Do you need the dictionary table? In practice, various optimization is incorporated, such as variable table index size.

Questions

1. Run Lempel-Ziv on the bit string 010010001001110. Use 5-bit indices.
2. What's wrong with the following argument?
Since all the $post[v]$ are unique in a DFS, topological sort is unique.
3. What's missing with the following argument?
If DFS gives the same $post[v]$ every time, the topological sort must be unique.
4. Is $\lceil \log n \rceil!$ polynomially bounded? Is $\lceil \log \log n \rceil!$ polynomially bounded? What about $\lceil \log^k n \rceil!$ for some integer k ?