Homework Seven, for Fri 11/13 CSE 101

Prepare a PDF file in which your solution to each of the following problems (1–6) begins on a fresh page. Upload the file to Gradescope, using your campus email address as login. The deadline is noon on Friday.

These problems cover the following skills and concepts:

- Creative uses of Dijkstra's algorithm.
- Reducing problems to canonical graph operations.
- More practice with greedy algorithms.
- Dynamic programming in one variable, given the subproblem.

---

1. *Shortest cycle in a graph.* Textbook problem 4.11. You should be able to handle this by calling Dijkstra's algorithm repeatedly.

2. *A generalized shortest-paths problem.* Textbook problem 4.19. Solve it by invoking Dijkstra's algorithm on a suitably transformed graph. We would thus say that the generalized shortest-paths problem *reduces to* the shortest-paths problem.

3. *Which road to add?* Textbook problem 4.20. Can you do this by just calling Dijkstra's algorithm *twice*, and then doing a linear amount of computation?

4. *One-dimensional set cover.* Suppose we are given points $x_1, \ldots, x_n$ on the real line (that is, each x_i is a real number). We are also given a collection of intervals $[s_1, e_1], \ldots, [s_m, e_m]$. Show that there is an efficient greedy algorithm that finds the minimum number of intervals needed to cover all the x_i.

5. Recall the *string reconstruction* problem from class: You are given a string of n characters $x[1 \ldots n]$, which you believe to be a corrupted text document in which all punctuation has vanished (so that it looks something like "itwasthebestoftimes..."). You wish to reconstruct the document using a dictionary, which is available in the form of a Boolean function $\texttt{dict}(\cdot)$: for any string w,

$$\texttt{dict}(w) = \begin{cases} \texttt{true} & \text{if } w \text{ is a valid word} \\ \texttt{false} & \text{otherwise} \end{cases}$$

   In class, we found a dynamic programming algorithm that determines whether $x[\cdot]$ is a sequence of valid words, and runs in $O(n^2)$ time, assuming calls to `dict` take unit time. We'll now consider an alternative, graph-theoretic approach.

   (a) Consider a directed graph $G = (V, E)$ in which $V = \{1, 2, \ldots, n+1\}$, and there is an edge (i, j) whenever $i < j$ and $x[i \cdots j-1]$ is a valid word. Show that $x[\cdot]$ is a valid sequence of words if and only if node $n+1$ is reachable from node 1 in G. (In other words, *string reconstruction reduces to reachability in graphs*). How long does this graph-based algorithm take?

   (b) Often there are several alternative ways to reconstruct a string. For instance, "potato" could be reconstructed as a single word, or as a sequence of three words, "pot", "a", "to". In such situations, we would like a *minimal* reconstruction: one involving the minimum number of words. Show how to solve this problem using a graph algorithm (that is, reduce it to a standard graph problem); it should output the actual minimal sequence of words. What is the running time?

6. *Dynamic programming vs. greedy.*

(a) Solve problem 6.3 in the textbook. Use the following subproblem: for all $1 \leq j \leq n$,

$$P(j) = \text{maximum profit achievable using only the first } j \text{ locations.}$$

(b) Once you have solved the problem by dynamic programming, consider this greedy approach:

```
Repeat:
   Pick the location i with highest profit p_i
   Remove all locations within k miles of location i
```

Do you think this will work? Why or why not?