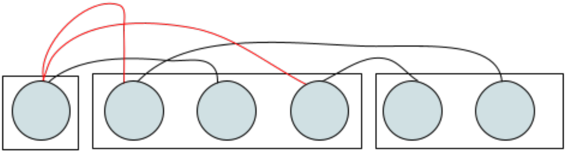
Bread First Search – Editorial

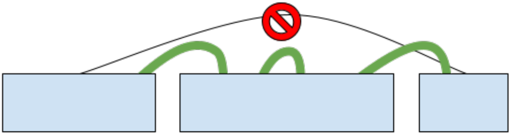
Robert Cummings / Eliden

May 13, 2021

Sample Input



Good and Bad Edges



Formal Definitions

We will zero-index the vertices. Let d be the distance array after adding edges. We need $d[0] = 0$, $d[1] = 1$, and

$$d[i] \leq d[i+1] \leq d[i] + 1$$

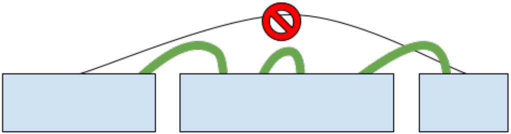
Formal Definitions

We will zero-index the vertices. Let d be the distance array after adding edges. We need $d[0] = 0$, $d[1] = 1$, and

$$d[i] \leq d[i+1] \leq d[i] + 1$$

Let us direct edges so that $i \to j$ when $i < j$. The only edge going into j that matters is the earliest one. That is, define $back[j] = \min\{k : k \to j\}$, or $back[j] = j$ if the set is empty.

$$d[i] \leq d[back[i]] + 1$$



Dynamic Programming

Let $dp[i]$ be the minimum number of edges added so that the prefix $0, 1, \ldots, i$ is valid, and i is the end of a block. There can be no edges skipping over i's block.

Dynamic Programming

Let $dp[i]$ be the minimum number of edges added so that the prefix $0, 1, \ldots, i$ is valid, and i is the end of a block. There can be no edges skipping over i's block.

We have $dp[0] = 0$. For $i > 0$:

$$dp[i] = \min_{\substack{0 \le j < i \\ \forall k > i:\ back[k] > j}} \left(dp[j] + |\{k \in \{j+1, \ldots, i\} : back[k] > j\}| \right)$$

Key Observations

$$dp[i] = \min_{\substack{0 \le j < i \\ \forall k > i:\ back[k] > j}} \left(dp[j] + |\{k \in \{j+1, \ldots, i\} : back[k] > j\}| \right)$$

Naively $O(n^3)$, but we can make observations:

- As i increases, the valid js only increase

Key Observations

$$dp[i] = \min_{\substack{0 \le j < i \\ \forall k > i:\ back[k] > j}} \left(dp[j] + |\{k \in \{j+1, \ldots, i\} : back[k] > j\}|\right)$$

Naively $O(n^3)$, but we can make observations:

- As i increases, the valid js only increase
- As j increases, the set $\{k \in [j+1, i] : back[k] > j\}$ shrinks

Key Observations

$$dp[i] = \min_{\substack{0 \le j < i \\ \forall k > i:\ back[k] > j}} \left(dp[j] + |\{k \in \{j+1, \ldots, i\} : back[k] > j\}| \right)$$

Naively $O(n^3)$, but we can make observations:

- As i increases, the valid js only increase
- As j increases, the set $\{k \in [j+1, i] : back[k] > j\}$ shrinks
- For $i > 0$, $dp[i+1] \le dp[i] + 1$, by using the same j

A Two Pointers Solution

$$dp[i] = \min_{\substack{0 \le j < i \\ \forall k > i:\, back[k] > j}} \left(dp[j] + |\{k \in \{j+1, \ldots, i\} : back[k] > j\}| \right)$$

To implement this, we increase $j < i$ together, and use

- $dp[i+1] \le dp[i] + 1$ (for $i > 0$)
- $dp[i] \le dp[j] + |\{k \in [j+1, i] : back[k] > j\}$

The set $|\{k \in [j+1, i] : back[k] > j\}$ and its size are efficiently maintained as i and j increase using a boolean array.
The solution runs in $O(n+m)$ time.

Miscellaneous

- There is also an $O(n \log n)$ solution with a lazy segment tree
- I came up with the problem by asking the question before knowing the solution
- It is a good contest strategy to try all the problems