Design and Analysis of Algorithms

6.3 Dynamic Programming Exercises

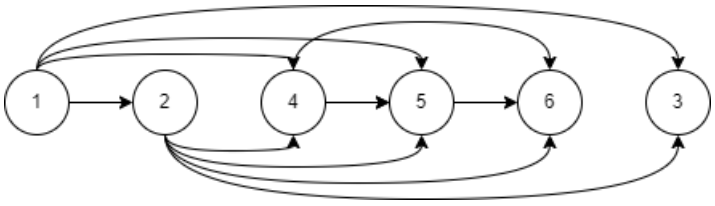
Daniel Shannon

May 16th, 2022

6.3.2

- You have a sequence of numbers $x_1, x_2, x_3, ..., x_n$
- Find the contguous subsequence $[x_i, ..., x_j]$ with the greatest sum
- Not allowed to skip elements!
- Use dynamic programming to find an $O(n)$ solution

My Answer



example linearized DAG

Similar to largest increasing subsequence, but we track the sum instead of the length.

$prev(0) = x_1$
for all $j = 1, 2, ...n$, in linearized order **do**
    sum_j+=$max\{S[j] + x_j : (i, j) \in E\}$
    if $S[j] < sum_j$ **then**
        $S[j] = sum_j$
        $prev(j) \cdot (i, j)$ concat
    end if
end for

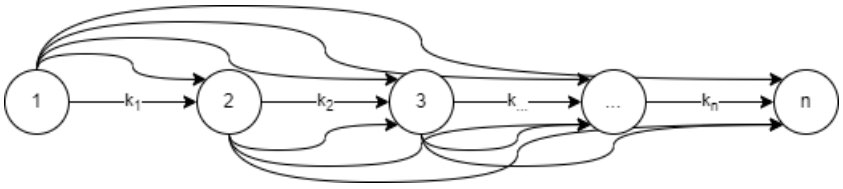
Solution

$S[j]$ maximum sum of a cont subsequence ending at index j
$S[j] = max\{A[j], S(j-1) + A[j]\}$
$S[0] = 0$
for all $j = 1, 2, ...n$, in linearized order **do**
    $S[j] = max\{A[j], S(j-1) + A[j]\}$
end for

6.3.4

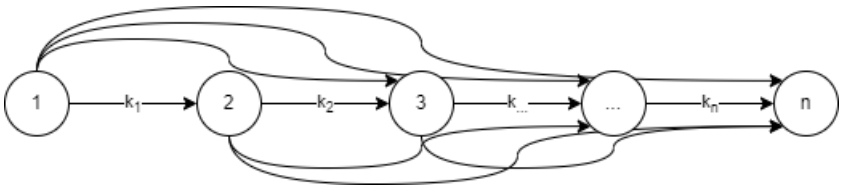
- You are trying to decide where to build a chain of restaurants along a linear highway.
- At each location i along the highway, you will make a profit of $p(i)$ that depends on the location.
- You are not allowed to put two restaurants within k miles of each other.
- Where should you place the restaurants to maximize profit?
- Hint: Draw the DAG!

My Answer



$P[j]$ max profit ending at index j
$P[j] = max\{p_j, (i,j) \in E : M(i) + p\}$
$P[0] = 0$
$dist(0) = 0$
for all $j = 1, 2, ...n$, in linearized order **do**
    $P[j] = max\{p_j - k, (i-k, j) \in E : P(i-k) + p\}$
end for

Solution



6.3.6

- Suppose you have two strings of length n and m
- What is the maximum edit distance between the two strings?

$E(i, j)$ graph of edges between letters i of $string_n$ and j of $string_m$
for all $i = 0, 1, 2, ..., m$ **do**
    $E(i, 0) = i$
end for
for all $i = 1, 2, ..., n$ **do**
    $E(0, j) = j$
end for
for all $i = 1, 2, ..., m$ **do**
    for all $j = 1, 2, ..., n$ **do**
        $E(i, j) = max\{E(i-1, j) + 1, E(i, j-1) + 1\};$
    end for
end for

[1] Dasgupta, Papadimitriou, Vazirani *Algorithms* p159

6.3.8

- You are given a string of characters without any spaces or punctuation. Itlookslikethissentence. You want to figure out whether it is possible to insert spaces to split the string into valid words.
- You are given a dictionary to check whether a sequence of characters is a valid word.
- Create a dynamic programming algorithm to determine whether the string can be split into valid words.

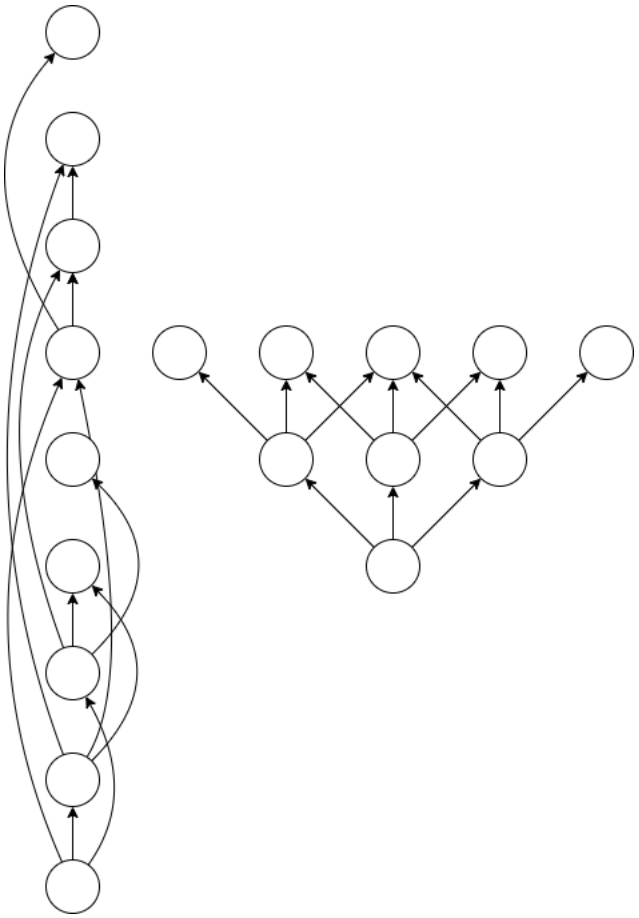
$S[(i,j)]$ a string of letters
$D(i) = W_D[(i,j)]$ a dictonary of words
$W_S[i]$ an array of words from S
for all $(i,j) \in S$ **do**
    $i = 0$
    while $minEdit(i, D(i)) \neq 0$ **do**
        $i++$
    end while
    if minEdit(i,D(i))=0 **then**
        $W_S[i] = D(i)$
    end if
end for

Solution

$W[i]$ can string $S[1...i]$ be split into words?
for all $i = 1 : n$ **do**
    $W[i] = 0$ no
    for all $j = 1 : i$ **do**
        if $W[j] = 1$ and $S[j+1...i] \in D$ **then**
            $W[i] = 1$
        end if
    end for
end for

3.3.10

- You are trying to climb a wall made of blocks. It is N blocks tall and M blocks wide.
- At each block (i, j), you can climb up to $(i + 1, j)$ or diagonally up to $(i + 1, j \pm 1)$ (unless you are on a boundary position)
- Each block has a *danger value* $d_{i,j}$
- Your job is to climb to the top of the wall, minimizing the sum of danger values along the path.



s starting node
$path(s) = s$ path vector starting at node s
$G(E, V), (u, v) \in E$
for all $i = 1 : n$ **do**
    $path(u) += min(danger(u_i))$
end for

*not my best effort, want to finish before class though!

Solution

$D(i, j)$=minimum danger value required to get to block (i, j).
$D(i, j) = min\{D(i-1, j), D(i-1, j-1), D(i-1, j+1)\} + d_{i,j}$