

Design and Analysis of Algorithms

6.3 Dynamic Programming Exercises

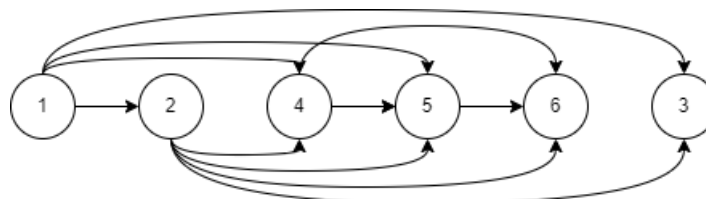
Daniel Shannon

May 16th, 2022

6.3.2

- You have a sequence of numbers $x_1, x_2, x_3, \dots, x_n$
- Find the contiguous subsequence $[x_i, \dots, x_j]$ 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) = x1
for all j = 1, 2, ...n, in linearized order do
  sumj += max{S[j] + xj : (i, j) ∈ E}
  if S[j] < sumj then
    S[j] = sumj
    prev(j) · (i, j) concat
  end if
end for

```

Solution

```

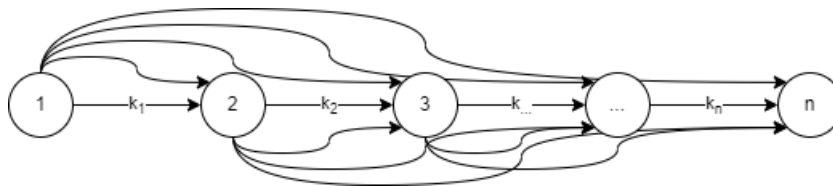
S[j] maximum sum of a cont subsequence ending at index j
S[j] = max{A[j], S(j - 1) + A[j]}
S[0] = 0
for all j = 1, 2, ...n, in linearized order do
  S[j] = max{A[j], S(j - 1) + A[j]}
end for

```

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

$P[j] = \max\{p_j, (i, j) \in E : M(i) + p\}$

$P[0] = 0$

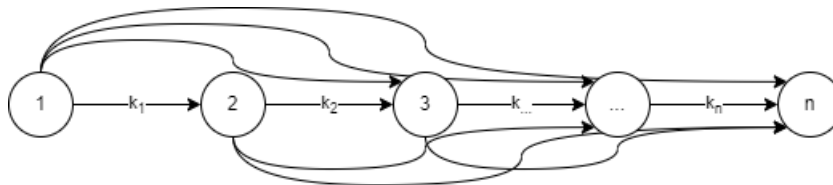
$dist(0) = 0$

for all $j = 1, 2, \dots, n$, 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 $i = 0, 1, 2, \dots, m$ **do**
 $E(i, 0) = i$
end for
for all $i = 1, 2, \dots, n$ **do**
 $E(0, j) = j$
end for
for all $i = 1, 2, \dots, m$ **do**
 for all $j = 1, 2, \dots, n$ **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. It looks like this sentence. 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 dictionary of words
 $W_S[i]$ an array of words from S
for all $(i, j) \in S$ **do**
 $i = 0$
 while $\text{minEdit}(i, D(i)) \neq 0$ **do**
 $i++$
 end while
 if $\text{minEdit}(i, D(i)) = 0$ **then**
 $W_S[i] = D(i)$
 end if
end for