CMPT307: Dynamic Programming

Week 7-3

Xian Qiu

Simon Fraser University

xianq@sfu.ca

Outline

- longest common subsequence
- optimal binary search tree

Subsequence

- $X = \langle x_1, \ldots, x_m \rangle$ is a sequence of characters
- $Z = \langle z_1, \ldots, z_k \rangle$ is a subseqence of X if each $z_i \in X$ and Z "preserves" the order of X

Example

$Z = \langle B, C, C, A \rangle$ is a subsequence of $X = \langle A, B, C, D, C, B, A \rangle$.

- Z is a common subsequence of X, Y if Z is a subsequence of both X and Y

Example

$Z = \langle B, D, A \rangle$ is a common subsequence of

$$X = \langle A, B, C, D, C, B, A \rangle , \quad Y = \langle D, B, C, A, C, D, A \rangle .$$

Longest common subsequence

- input: sequences X and Y, of lengths m and n respectively
- output: longest common subsequence of X and Y

- application in computational biology: X, Y stand for DNA strings over a finite set $\{$A,C,G,T$\}$

$$X = \text{ACCGGTCGAGTGCGCGGAAGCCGGCCGAA}$$
$$Y = \text{GTCGTTCGGAATGCCGTTGCTCTGTAAA}$$

- use longest common subsequence to determine the similarity of X and Y

Optimal Substructure

$X_m = \langle x_1, \ldots, x_m \rangle$, $Y_n = \langle y_1, \ldots, y_n \rangle$ OPT $= Z_k = \langle z_1, \ldots, z_k \rangle$

idea: compare z_k with x_m and y_n respectively

case 1: $z_k = x_m = y_n$

- Z_{k-1} is an LCS of X_{m-1} and Y_{n-1}

case 2: $z_k \neq x_m$

- Z_k is an LCS of X_{m-1} and Y_n

case 3: $z_k \neq y_n$

- Z_k is an LCS of X_m and Y_{n-1}

Recursive Formula

define $c[i, j] =$ length of LCS of X_i and Y_j

assume $i, j > 0$ and recall the three cases:

case 1: $x_i = x_j$

- ▷ $c[i, j] = c[i - 1, j - 1] + 1$

case 2: $x_i \neq x_j$

- ▷ $c[i, j] = \max\{c[i - 1, j], c[i, j - 1]\}$

if $i = 0$ or $j = 0$, then $c[i, j] = 0$

Running Time

$$c[i,j] = \begin{cases} 0 & \text{if } i = 0 \text{ or } j = 0 \\ c[i-1, j-1] + 1 & \text{if } i, j > 0 \text{ and } x_i = y_j \\ \max\{c[i, j-1], c[i-1, j]\} & \text{if } i, j > 0 \text{ and } x_i \neq y_j \end{cases}$$

- ▷ depth: $O(mn)$ subproblems
- ▷ width: $O(1)$ time to compute $c[i,j]$
- ▷ running time $O(mn)$

how to construct the optimal solution?

- ▷ use $b[i,j]$ to record the three cases

LCS-LENGTH(X,Y)

$c[i,0] = c[0,j] = 0$, $i = 1,\ldots,m$ and $j = 1,\ldots,n$;
for $i = 1$ *to* m **do**
    for $j = 1$ *to* n **do**
        if $x_i == y_j$ **then**
            $c[i,j] = c[i-1,j-1] + 1$;
            $b[i,j] = \text{``} \nwarrow \text{''}$;
        else if $c[i-1,j] \geq c[i,j-1]$ **then**
            $c[i,j] = c[i-1,j]$;
            $b[i,j] = \text{``} \uparrow \text{''}$;
        else
            $c[i,j] = c[i,j-1]$;
            $b[i,j] = \text{``} \leftarrow \text{''}$;
return m and s;

Example

i	j	0	1	2	3	4	5	6
	y_j		B	D	C	A	B	A
0	x_i	0	0	0	0	0	0	0
1	A	0	↑0	↑0	↑0	↖1	←1	↖1
2	B	0	↖1	←1	←1	↑1	↖2	←2
3	C	0	↑1	↑1	↖2	←2	↑2	↑2
4	B	0	↖1	↑1	↑2	↑2	↖3	←3
5	D	0	↑1	↖2	↑2	↑2	↑3	↑3
6	A	0	↑1	↑2	↑2	↖3	↑3	↖4
7	B	0	↖1	↑2	↑2	↑3	↖4	↑4

Language Translation

English	hello	good	bad	morning	molecule	$\cdots$
French	bonjour	bien	mal	matin	molécule	$\cdots$

- English = key, French = value and build a binary search tree
- some English words may have no French translation (so they do not appear in binary search tree)
- search time = $O(\log n)$ for red-black tree
- words have different frequencies
- goal: minimize the running time of all searches

Optimal BST

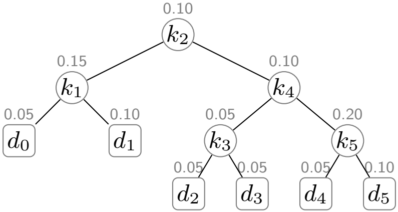
- a sequence $K = \langle k_1, \ldots, k_n \rangle$ of n distinct keys in sorted order, *i.e.* $k_1 < \ldots < k_n$
- p_i is the probability of searching k_i
- introduce dummy keys $d_0, \ldots, d_k$ for searches not in K, where $d_{i-1} < k_i < d_i$ for $i = 1, \ldots, k$
- q_i is the probability for a dummy key i $(i = 0, \ldots, n)$
- the total probability is one, *i.e.*, $\sum_{i=1}^{n} p_i + \sum_{i=0}^{n} q_i = 1$
- search cost of T

$$C_T := \sum_{v \in T} C_v, \text{ where } C_v = \text{ search cost of } v$$

- goal: $\min_T \ \mathbb{E}[C_T]$

Example

i	0	1	2	3	4	5
p_i		0.15	0.10	0.05	0.10	0.20
q_i	0.05	0.10	0.05	0.05	0.05	0.10

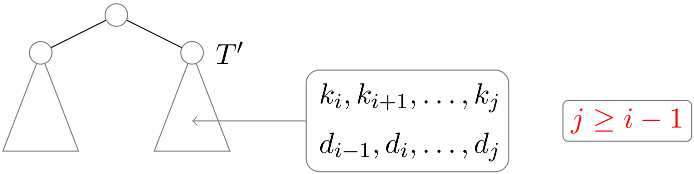


- ▷ all leaves are dummy keys and vice versa
- ▷ $\mathbb{E}(C_T) = 2.8$ not optimal
- ▷ T is an optimal binary search tree if $\mathbb{E}(C_T)$ is minimum

Optimal substructure

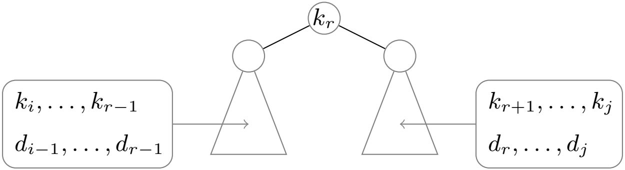
Lemma

Let T' be a subtree of OPT, with keys $k_i, \ldots, k_j$. Then T' is optimal for the subproblem with keys $k_i, \ldots, k_j$ and dummy keys $d_{i-1}, \ldots, d_j$.



Recursive Solution

$f(i,j)$:= optimal value for the instance with keys $k_i, \ldots, k_j$ and dummy keys $d_{i-1}, \ldots, d_j$



let $w(i,j) = \sum_{l=i}^{j} p_l + \sum_{i-1}^{j} q_l$ = total probability

$$\begin{aligned} f(i,j) &= p_r + f(i,r-1) + w(i,r-1) + f(r+1,j) + w(r+1,j) \\ &= f(i,r-1) + f(r+1,j) + w(i,j) \end{aligned}$$

find an optimal r by brute force!

Recursive Formula

if $j \geq i$:

$$f(i,j) = \min_{1 \leq r \leq j} \{f(i, r-1) + f(r+1, j) + w(i,j)\}$$

if $j = i - 1$: $f(i,j) = q_{i-1}$

- $f(1,n)$ returns the optimum cost
- running time?
- how to construct an optimal BST?