HW6 Solution hints

6.19 Coin denominations $x_1, x_2, \cdots, x_n$

For each denomination there is an unlimited supply of coins.

We need to make change for a value v using at most k coins. Therefor, we can assume that we have total of kn coins. Let $y_1, y_2, \cdots, y_{kn}$ be the coins.

Let $C(t,j)$ be the optimal # of coins from $y_1, y_2, \cdots, y_j$ to change t, where $1 \le t \le v$ & $1 \le j \le kn$. Let u_j be the value of coin y_j.

$\therefore$ # of subproblems is $O(knv)$

We can write $C(t,j)$ recursively as

$$C(t,j) = \min\left(C(t-u_j, j-1)+1, C(t,j-1)\right)$$

Basis $C(a,j) = 0$ for any $a \le 0$

& $C(t,0) = 0$ for any $t \ge 0$

Memoized_$C(t,j)$

if $t \le 0$ return 0;

if $j = 0$ return 0;

if $C(t,j)$ is already computed then return $C(t,j)$

else return $C(t,j) = \min\{C(t-y_j, j+1)+1, C(t,j-1)\}$

4. Retriere_C(t, j)

{ if $t \le 0$ return null.

if $j = 0$ return ""

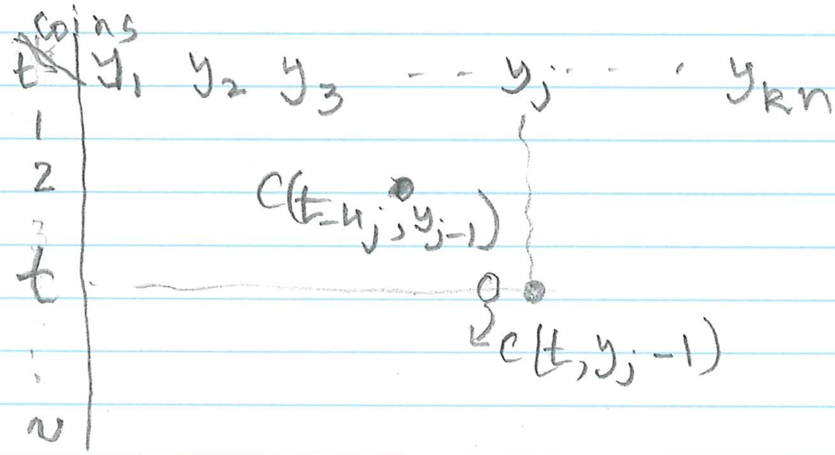
if $c(t, j) = 1 + c(t - u_j, j-1)$ then

Print y_j

}

5. <u>Iterative solution</u>

Subproblems



We evaluate row-wise

for $t = 1, 2, \cdots, n$

for $j = 1, 2, \cdots, kn$

$$c(t, j) = \min\{1 + c(t - u_j, j-1), c(t, j)\}$$

Chain Matrix Multiplication

We need to optimally multiply

$A_1 \times A_2 \times \cdots \times A_n$. Where the dimension of A_i is $m_i \times m_{i+1}$,

$i = 1, 2, \cdots, n$.

The dimension of the product matrix is

$m_1 \times m_{n+1}$.

Let $C(i,j)$ be the optimal multiplication cost of the matrices

$A_i \times A_{i+1} \times \cdots \times A_j$ $\quad 1 \le i \le n \quad 1 \le j \le n$

There are n^2 subproblems.

Basis: $C(1,2) = m_1 m_2 m_3$

$C(2,3) = m_2 m_3 m_4$

$\vdots$

$C(n-1,n) = m_{n-1} \cdot m_n \cdot m_{n+1}$

Also $C(i,i) = 0 \;\; \forall i$

Recurrence formula for $C(i,j)$

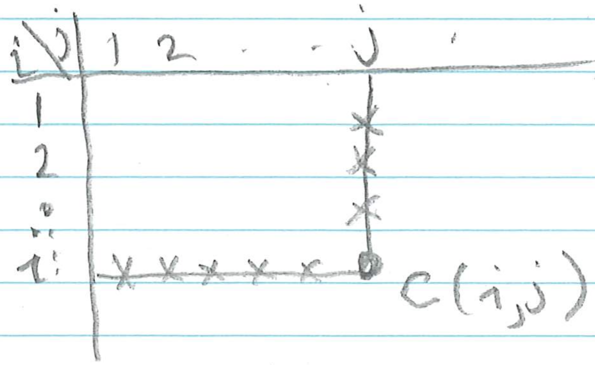
$$C(i,j) = \min_{i \le k < j} \left\{ \left(C(i,k) + C(k+1,j) + m_i \cdot m_{k+1} \cdot m_{j+1} \right) \right\}$$

Each subproblem requires $O(n)$ time once the smaller subproblem values are known.

$\therefore$ memoized cost is $O(n^3)$.

Iterative sol^{n}.

In order to compute $C(i,j)$



We note that all the values marked x must be evaluated before $C(i,j)$ can be computed.

$\therefore$ We evaluate row by row from left to right:

for $i = 1, 2, \ldots, n$

for $j = i, 2, \ldots, n$.

$$C(i,j) = \min_{i \le k < j} \left(C(i,k) + C(k+1,j) + m_i m_{k+1} m_{j+1} \right)$$

6.4 Subproblems: Define $S(i)$ to be 1 if $s[1, \ldots i]$ is a sequence of valid words. Otherwise it is 0.

Recurrence formula

$$S(i) = \max_{1 \le j < i} \{ S(j) : dict(s[j+1, \ldots, i] = true \}$$

If $S(j)$ is false for every $dict(s[j+1, \ldots, i] = true)$, $S(j)$ is returned 0.

$\therefore$ the given input sequence is a sequence of valid words if $S(n) = 1$

# of subproblems is n. Each subproblem can be solved in $O(n)$ time. Total cost is $O(n^2)$.

The basis is $S(i)$ is true for $i < 0$.

Iterative sol^n is easy

for $i = 0, \ldots n$
$S(i) = \ldots$

6.7 $L(i,j)$: length of the longest palindromic subsequence of string $s[i,\ldots,j]$.

For $1 \leq i \leq n$ & $1 \leq j \leq n$.

Basis $L(i,i) = 1 \;\; \forall\, i$

$L(i,j) = 0$ if $i > j$

We can show that

$$L(i,j) = 1 + L(i+1, j-1) \text{ if } x_i = x_j$$

$$= \min\{L(i+1,j), L(i,j-1)\} \text{ otherwise}$$

There are $O(n^2)$ subproblems. Each subproblem takes $O(1)$ time to compute.

The order with which we compute $L(\cdot,\cdot)$ is given below

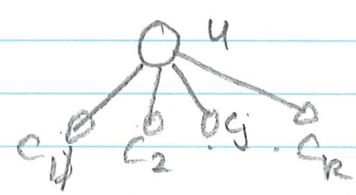
```
for j = 2, 3, ..., n
    for i = 1, 2, ..., j
        L(i,j) = ...
```

6.21 $V^+(u)$: size of the minimum vertex cover that includes u

$V^-(u)$: size of the minimum vertex cover that does not include u.

$\therefore V(u) = \min\{V^+(u), V^-(u)\}$

Consider



$$V^+(u) = 1 + \sum_{\forall c_i} V^-(c_i)$$ (all the edges (u, c_i) are covered by u)

$$V^-(u) = \sum_{\forall c_i} V^+(c_i)$$

the edge (u, c_j) is covered by c_j.

Note that

$V^+(u) = 1$ for all leaf nodes

$V^-(u) = 0$ " " "