search for j step to put the first jar

max (adversary) — worst case

breaks: $MinT(j-1, k-1)$

not breaks: $MinT(n-j, k)$

want j with Minimum (overall)

with j { max { break / not } ... }

$q \geq \log n$ or $n \leq 2^q$ — NOT POSSIBLE

$k \leq \lg n$ or Binary Search

$q = T(n,k)$ = min # trials to search (worst case) ladder size n with k jars

$n = R(k,q)$ = max ladder size that can be searched (worst case) with k jars and q trials

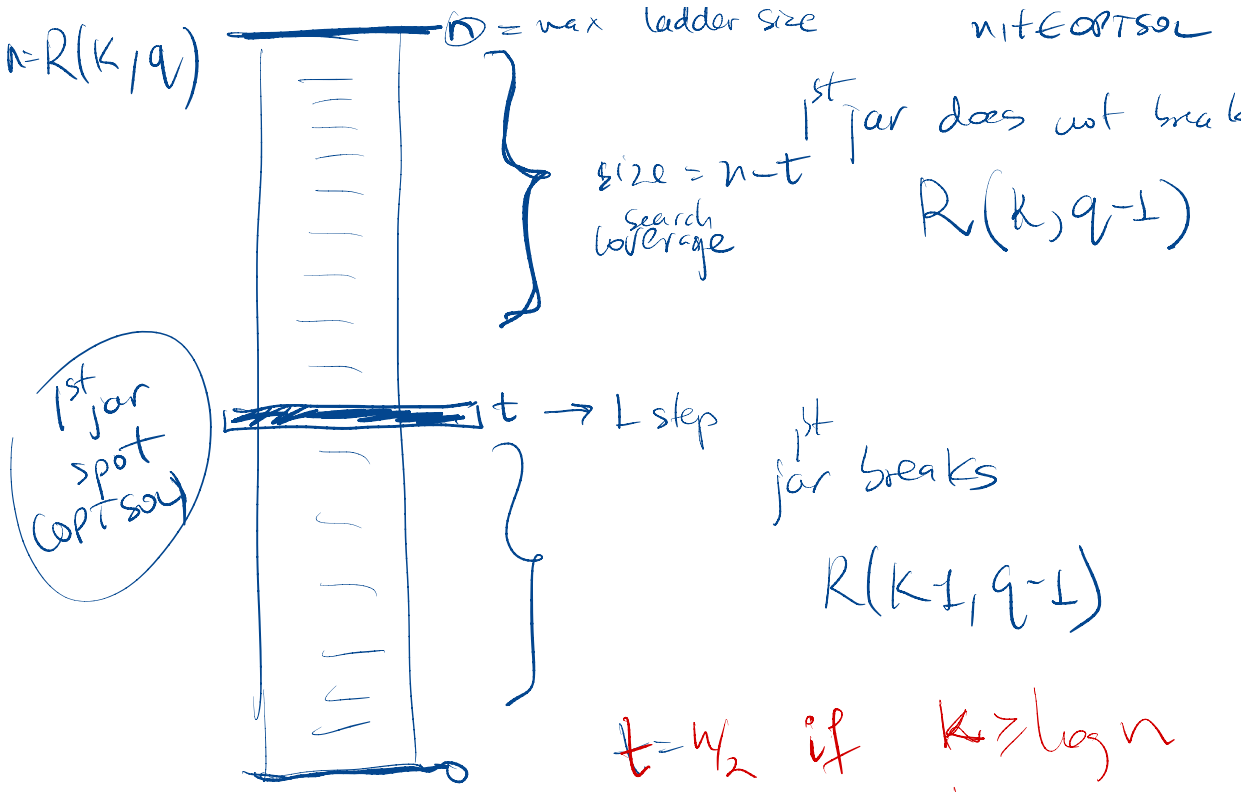
$$R(k,q) = \begin{cases} n = 2^q & k \geq q \\ \text{Binary Search} & \\ \text{TRIVIAL} & \end{cases}$$

$k < q$

- $k-1$ jars BinSearch linear
- down to 1 jar $\Rightarrow$ remain trials to be enough 2^{k-1} ($n = ?$)

If all jars break $q - (k-1)$ linear coverage

NOT OPTIMAL

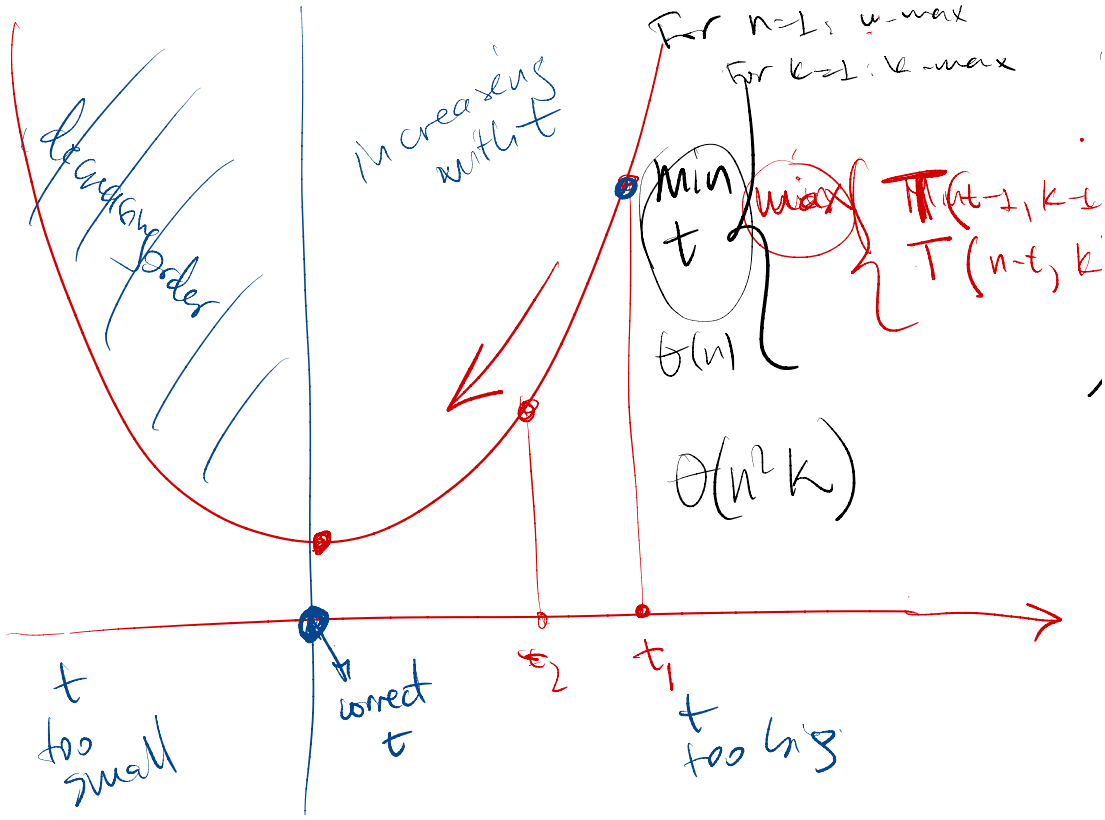


nitEOPTSOL

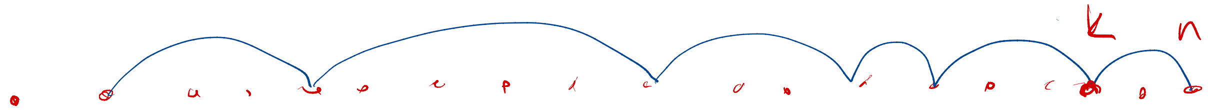
$t = n/2$ if $k \geq \log n$

$k \geq q/2$

$n = 2$



15.4-5



$c[n]$ = longest increase subseq ending in n

= search for k } previous element $A[k] \le A[n]$

$c[n] = c[k] + 1$ (max)

$S[n] = k$ previous element in longest this ending in n

linear both 15.4-5, 15.4-6

for $i = 1 : n$

search for k . . .

to compute $c[n]$

→ 15.4-5 linear search $O(n)$

→ 15.4-6 binary search $\theta(\log n)$

15-4

line: $l_i \; l_{i+1} \; l_{i+2}$ — — — l_j

words chunk i–j characters sum $L_{ij} = \sum_{t=i} l_t$

penalty$_G$ (#spaces) $(M - L_{ij} - j + i)^3$

allow 1 space between words → extras(i,j)

current last word

$c[j] =$

$= \min \left\{ \sum_{G \text{ except last}} \text{penalty}_G \right\}$

$= \min_i \{ c[i-1] + \text{penalty}(i,j) \}$

$$\text{penalty}(i,j) = \begin{cases} \infty & \text{if extras}(i,j) < 0 \text{ (words don't fit on line)} \\ 0 & \text{last line} \\ \text{extras}(i,j)^3 & \text{other lines} \end{cases}$$

i → break at first word on line