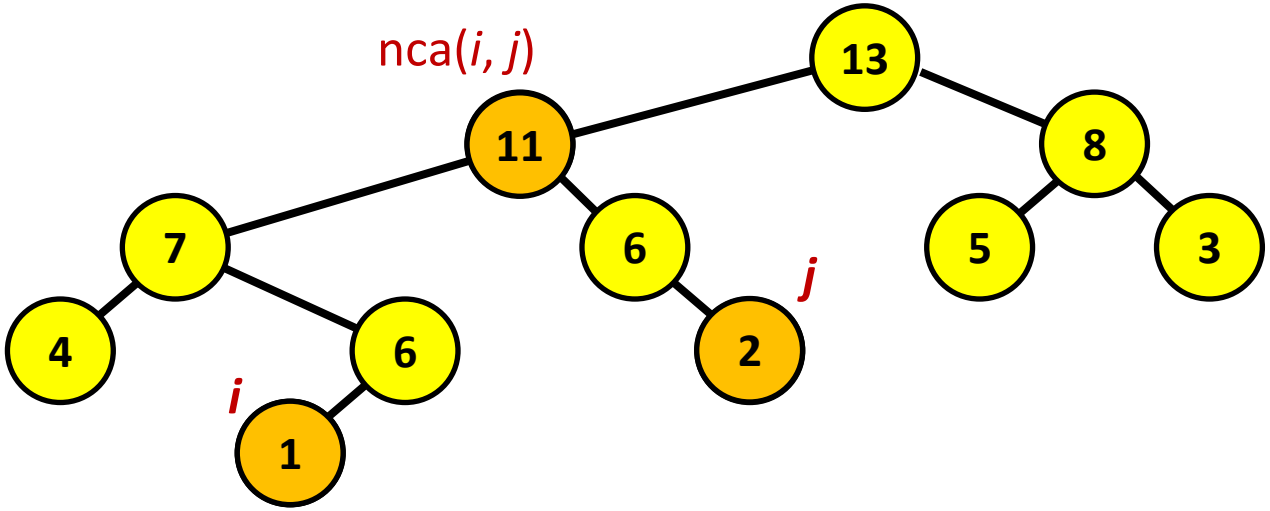
Nearest Common Ancestors (NCA)

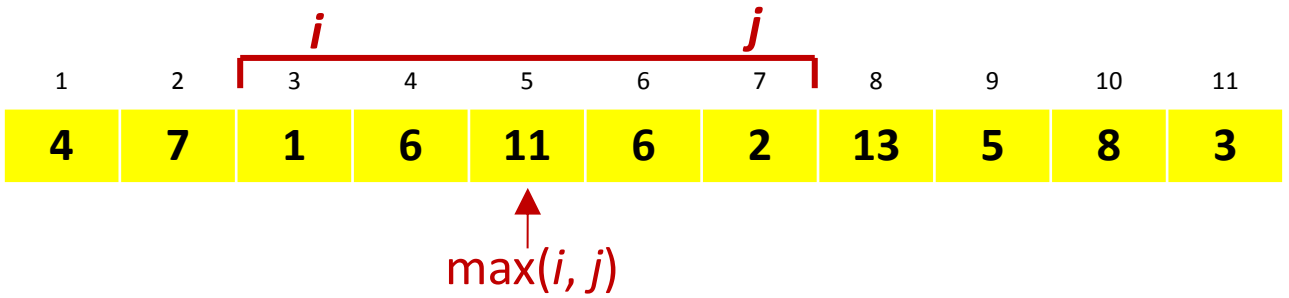
Org. [D. Harel, R.E. Tarjan, *Fast algorithms for finding nearest common ancestors*, SIAM J. on Comp. 13 (2): 338–355, 1984]

Preprocessing Time vs Query Time ?

Cartesian Tree [Vuillemin 1980]

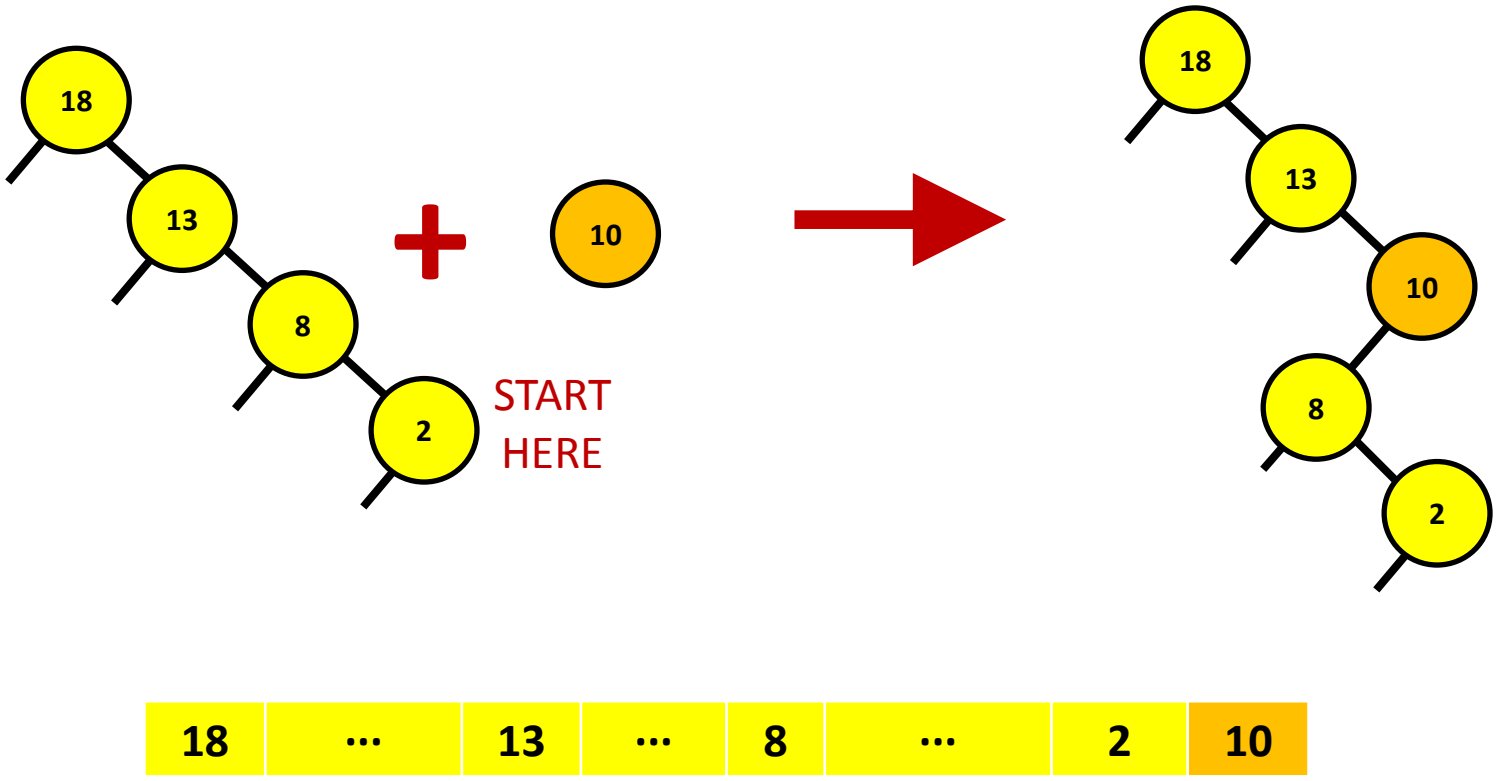


Discrete Range Maximum



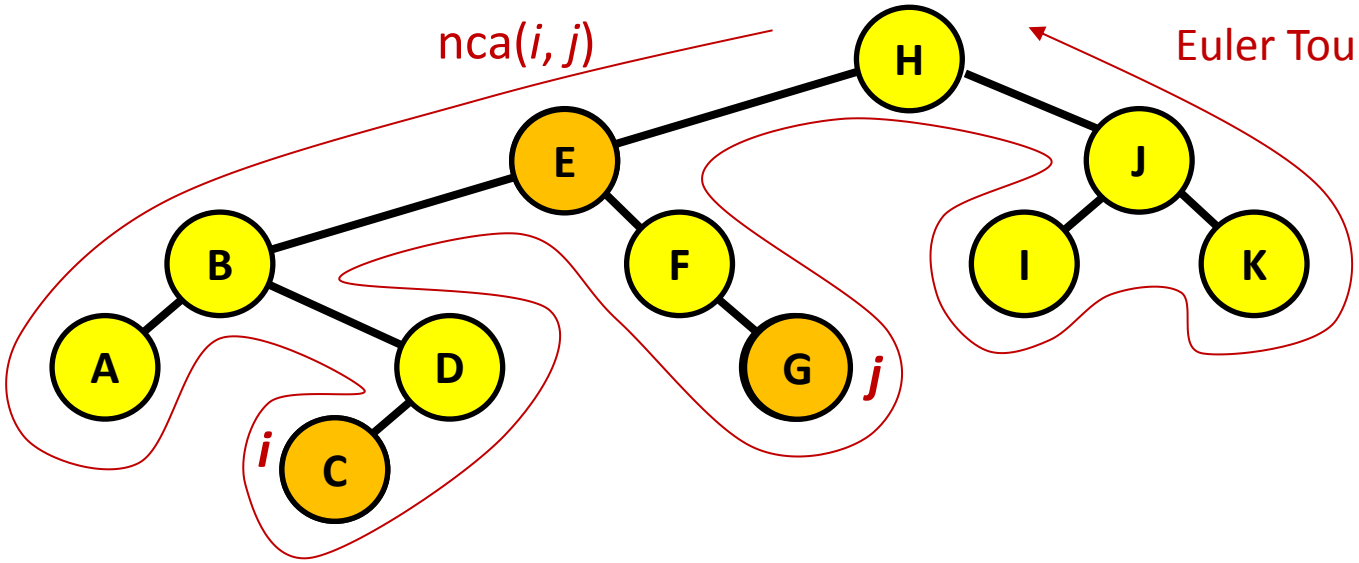
Cartesian Tree Construction

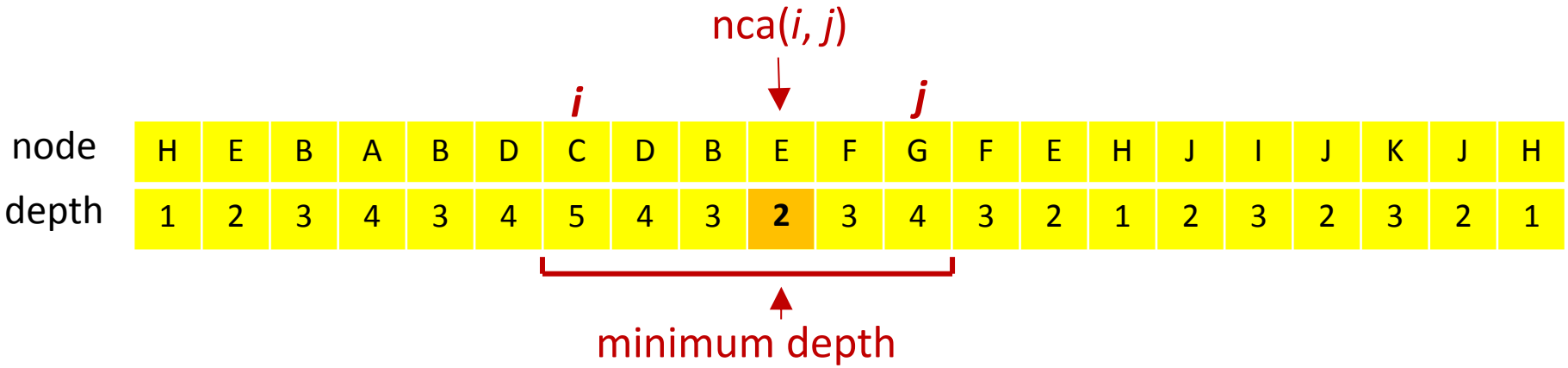
- Incremental construction left-to-right



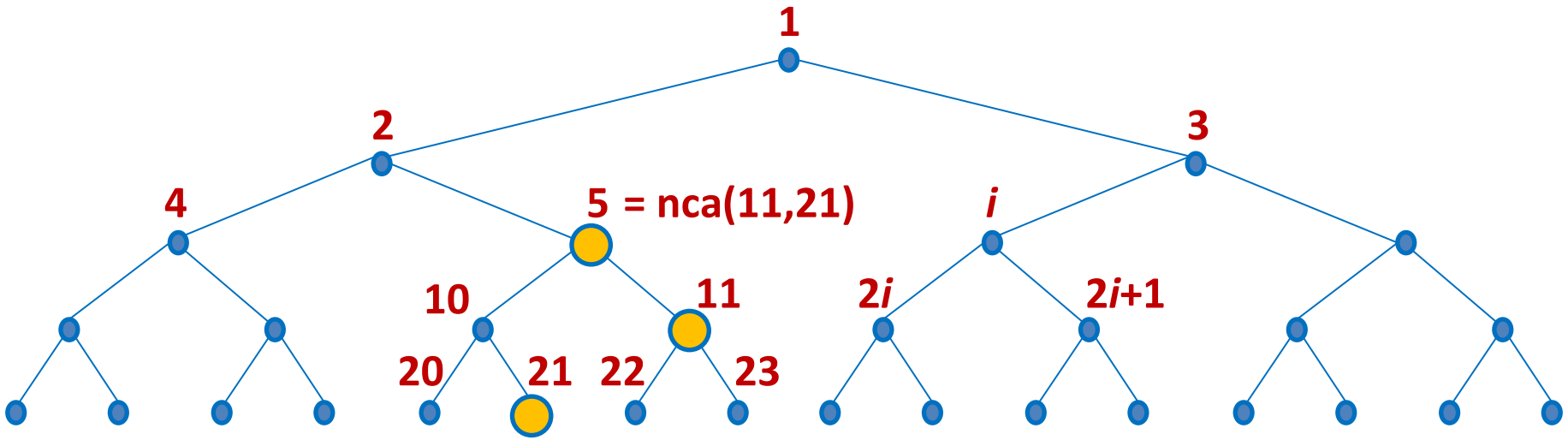
- O(n) time (Φ = #nodes on rightmost path)

Reduction: NCA $\Rightarrow$ ± 1 Discrete Range Maximum





NCA on Perfect Binary Trees



$11 = 1011_2$

$21 = 10101_2$

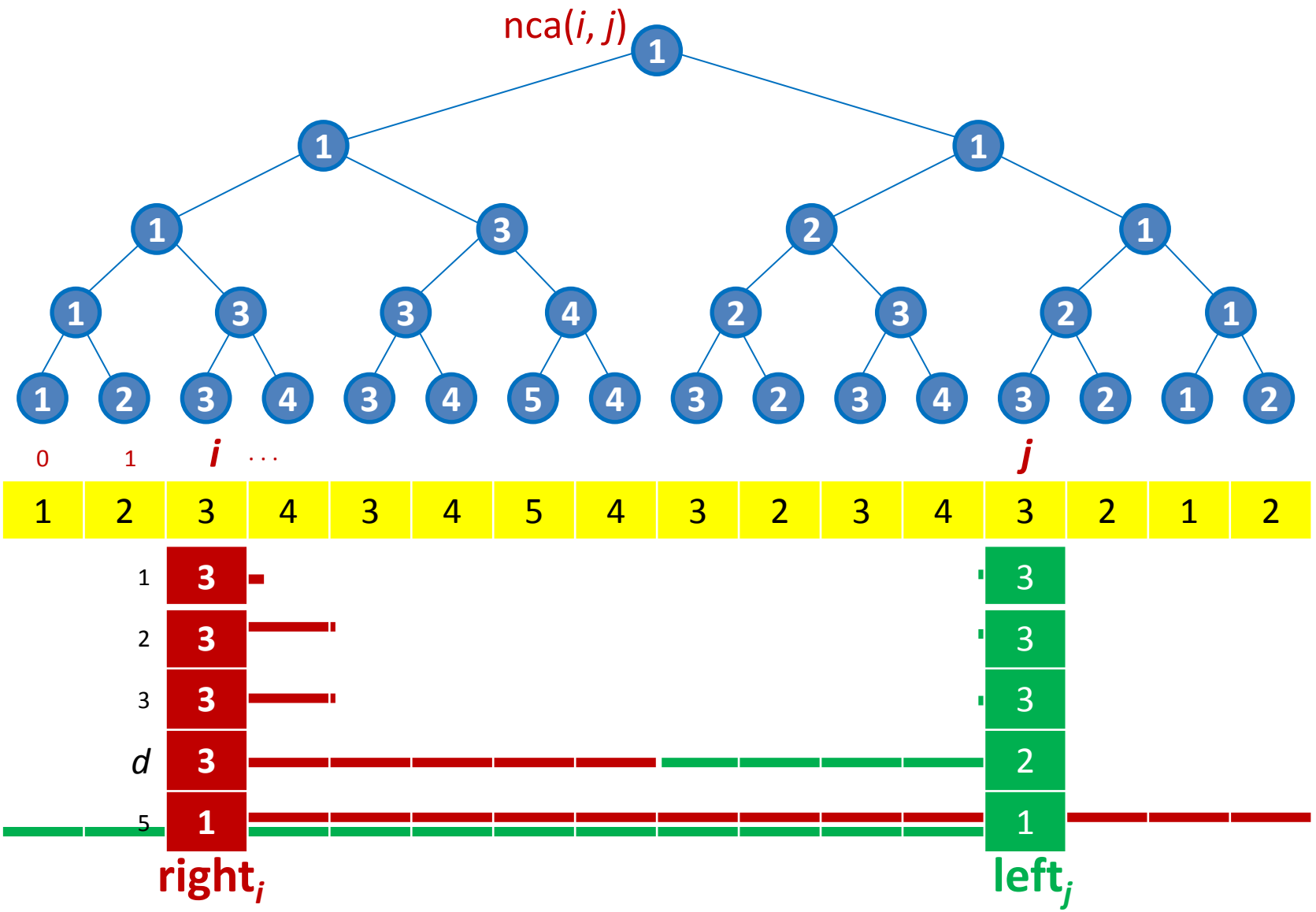
longest common prefix

$nca(21,21) = 5 = 101_2 = lcp(\ \mathbf{1011}_2\ ,\ \mathbf{101}01_2\ )$

```
proc lcp(x, y)
    if y < x then swap (x, y)
    return x >> (msb(x XOR (y >> (msb(y)-msb(x)))
```

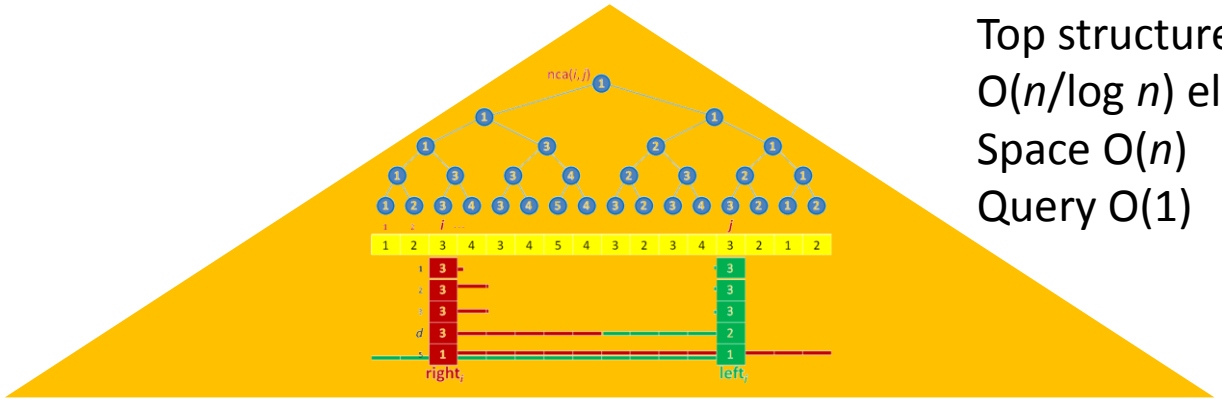
position of most significant bit ≠ 0

Discrete Range Mimimum – Space O(*n*·log *n*) words



drm(*i*, *j*) = min(right$_i$(*d*), left$_j$(*d*)) *d* = msb(*i* XOR *j*)

Blocked solution – Space O(*n*) words



Top structure
O(*n*/log *n*) elements
Space O(*n*)
Query O(1)

1 2 1 2

1 2 3 4 3 4 5 4 1 4 3 4 3 2 3 2

block of O(log *n*) elements

i

j

0 1 0 1 1 W_j (One for each *j*)

5 4 3 2 1

Block query: j+1-msb(W_j AND ((1 << (j-i+1))-1))

General query: 1 top query + 2 bottom queries

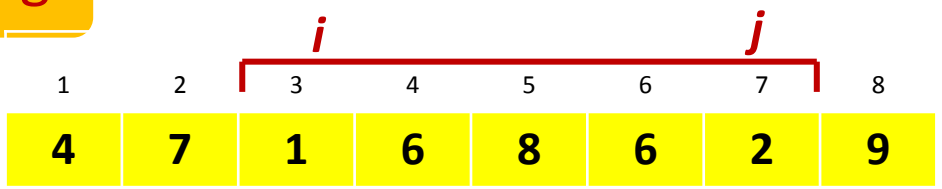
O(*n*) Preprocessing Time O(1) Query Time

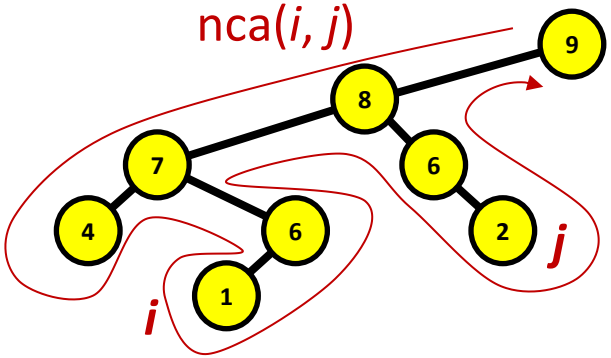
Summary...

General Discrete Range Searching

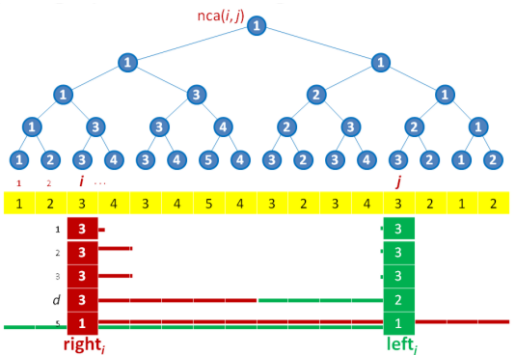
Cartesian Tree

NCA





Discrete Range Max on Depth Array

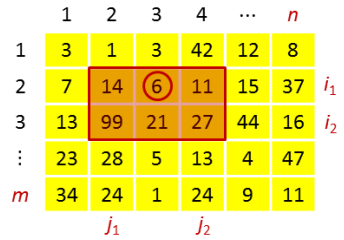


”O(n·log n)” solution on O(n/log n) blocks

O(log n) size blocks

O(n) Preprocessing Time O(1) Query Time

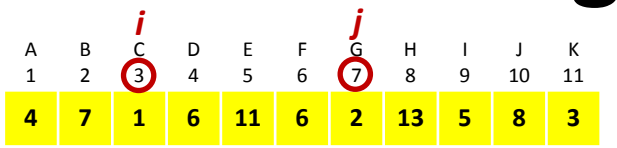
1d & 2D DRM Results

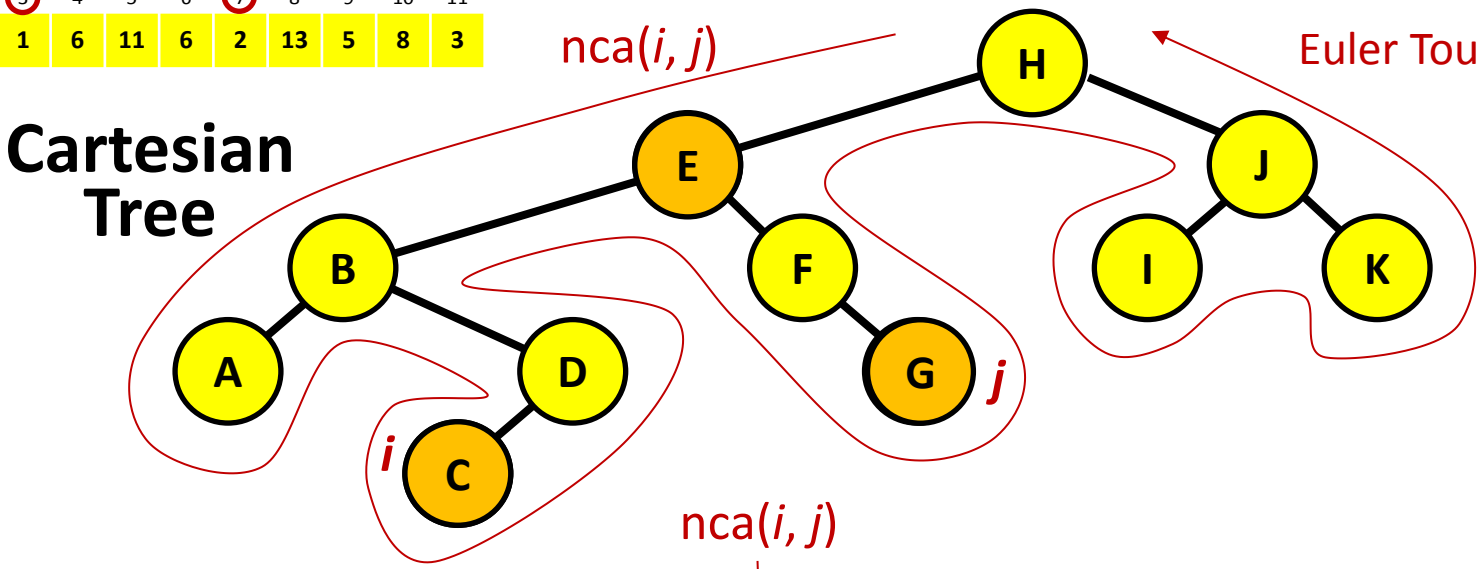


	Indexing Model (input accessible)	**Encoding Model** (input not accessable)
$m = 1$ 1D	$2n+o(n)$ bits, $O(1)$ time [FH07] n/c bits $\Rightarrow$ $\Omega(c)$ time [BDR10] n/c bits, $O(c)$ time [BDR10]	$\geq 2n - O(\log n)$ bits $2n+o(n)$ bits, $O(1)$ time [F10]
$1<m<n$	$O(mn \cdot \log n)$ bits, $O(1)$ time [AY10] $O(mn)$ bits, $O(1)$ time [BDR10] mn/c bits $\Rightarrow$ $\Omega(c)$ time [BDR10]	$\Omega(mn \cdot \log m)$ bits [BDR10] $O(mn \cdot \log n)$ bits, $O(1)$ time [BDR10] $O(mn \cdot \log m)$ bits, $O(mn)$ time [BBD13]
$m = n$ squared	$O(c \cdot \log^2 c)$ time [BDR10] $O(c \cdot \log c \cdot (\log\log c)^2)$ time [BDLRR12]	$\Omega(mn \cdot \log n)$ bits [DLW09] $O(mn \cdot \log n)$ bits, $O(1)$ time [AY10]

better upper or lower bound ?

DRM encoding - O(*n*) bits





nca(*i*, *j*)

node	H	E	B	A	B	D	C	D	B	E	F	G	F	E	H	J	I	J	K	J	H
depth	1	2	3	4	3	4	5	4	3	**2**	3	4	3	2	1	2	3	2	3	2	1

minimum depth

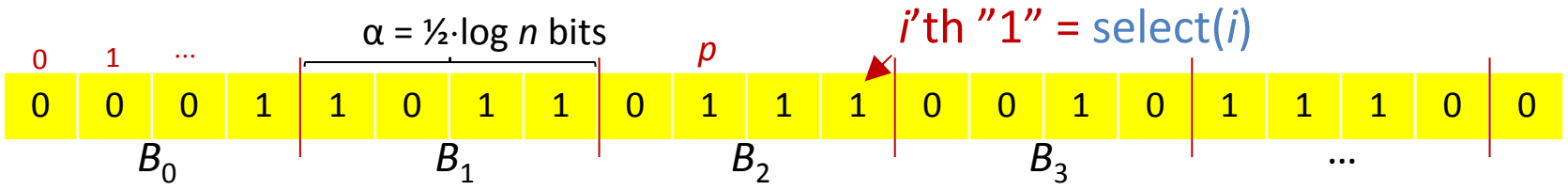
select(*i*) = *i*'th "1" — p_i — p — p_j — *j*'th "1" = select(*j*)

4*n* bits	0	0	0	1	1	0	1	1	0	1	1	1	0	0	1	0	1	1	1	0	0
	+	+	+	+	-	+	+	-	-	-	+	+	-	-	-	+	+	-	+	-	-

rank(p) = drm(i,j) = $\#_1$ left of p

depth = $\#_+ - \#_-$ = min-prefix-sum(p_i,p_j)

Succinct data structures for DRM, O(*n*) bits



$\text{rank}(p) = R_1[\lfloor p/\alpha \rfloor] + T_{\text{rank}}[B[\lfloor p/\alpha \rfloor], p \bmod \alpha]$ ($\alpha+\log\alpha$ bits)

- $R_1[i]$ = #$_1$ in first i blocks ($n/\alpha \cdot \log n$ bits)
- T_{rank} = rank inside block, table lookup ($2^{\alpha+\log\alpha} \cdot \log\alpha$ bits)

$\text{select}(i) = \alpha b + T_{\text{select}}[B[b], i\text{-rank}(\alpha b-1)]$

$b = R_{\text{nonempty}}[\text{rank}_{\text{leader}}(i)]$

- i is in block B[b]
- leader[i] = is the ith "1" the first "1" in its block? (n bits)
- $\text{rank}_{\text{leader}}(i)$ = rank structure for leader array (O(n) bits)
- R_{nonempty} = index of nonempty blocks ($n/\alpha \cdot \log n$ bits)
- T_{select} = select inside block, table lookup ($2^{\alpha+\log\alpha} \cdot \log\alpha$ bits)

$\text{min-prefix-sum}(p_i, p_j) = \alpha(b_k-1)+d_k$

$b_1 = \lfloor p_i/\alpha \rfloor$

$b_3 = \lfloor p_j/\alpha \rfloor$

$b_2 = \text{drm}_{\text{PS}}(b_1+1, b_3-1)$

$d_1 = T_{\text{mps}}[B[b_1], p_i \bmod \alpha, \alpha-1]$

$d_2 = T_{\text{mps}}[B[b_2], 0, \alpha-1]$

$d_3 = T_{\text{mps}}[B[b_3], 0, p_j \bmod \alpha]$

$k = \text{argmin}_{t=1..3}\ PS[b_t-1]+T_{\text{ps}}[B[b_t], d_t]$

- PS[b] = #$_+$ - #$_-$ for blocks $B_0..B_b$ ($n/\alpha \cdot \log n$ bits)
- T_{ps} = #$_+$ - #$_-$ for block prefix, table lookup ($2^{\alpha+\log\alpha} \cdot (1+\log\alpha)$ bits)
- T_{mps} = index of minimum prefix sum #$_+$ - #$_-$ inside range in a block, table lookup ($2^{\alpha+2\log\alpha} \cdot \log\alpha$ bits)
- MPS[b] = PS[b-1]+ T_{mps}[B[b], 0, α-1] ($n/\alpha \cdot \log n$ bits)
- drm_{MPS} = drmin structure for MPS, O(n/α) words (O(n) bits)