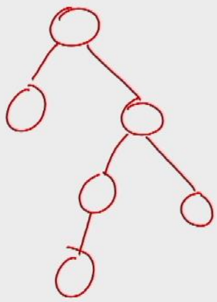
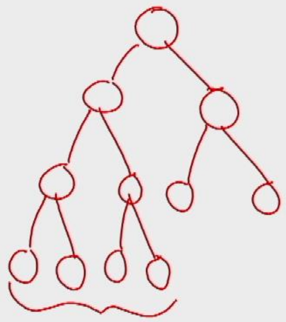
Balanced Binary Tree Technique

- Build a binary tree on the input
- Move up or down the tree one level a time

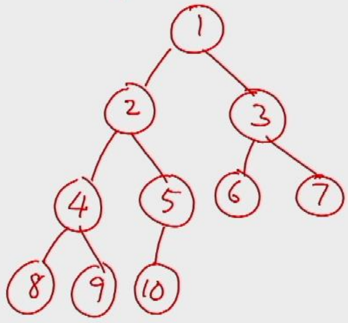
Binary Tree



Complete Binary Tree



Array representation



1	2	3	4	5	6	7	8	9	10

$1 \to 2, 3$

$2 \to 4, 5$

$3 \to 6, 7$

$i \to 2i, 2i+1$

---

step. $\underbrace{1\ 0\ 1\ 1\ 0\ 1\ 0\ 0}_{A[1..n]}$

n procs

$O(1)$ time

new array B

Binary tree of n leaves

$n-1$ internal nodes

leaves: n to $2n-1$

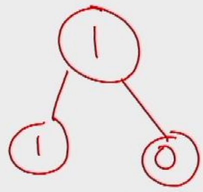
$A[1..n] \to B[n..2n-1]$

$A[i] \to B[n-1+i]$

Step 2 Binary tree in B

Substep 1 $n/2$ nodes that are parents of leaves

$n/2$ prcrs 1 time



Substep 2 $n/4$ prcrs 1 time

Substep 3 $n/8$ prcrs 1 time

Step 2

time: $\log_2 n$

$\text{Cost} = n/2 + n/4 + \cdots + 1 = O(n)$

$n/\log n$ processors	cost
$O(\log n)$ time	$O(n)$

OR of n bits on EREW PRAM

Input: Array $A[1 \dots n]$ of n bits. For simplicity, assume that n is a power of 2.
Output: $R =$ OR of the bits in A. Model: EREW PRAM.

```
{
    pardo for 1 ≤ I ≤ n
        B[n − 1 + I] = A[I];
    for s = 1 to log n do
        pardo for n/2^s ≤ I ≤ n/2^(s−1) − 1
            B[I] = B[2I] ∨ B[2I + 1];
    pardo for 1 == I
        return B[1];
}
```

Brent's Scheduling Principle

- Consider a parallel algorithm presented in T steps.
- Say the degree of parallelism of the i-th step is w_i
- So, the total number of instructions in the algorithm is $W = \sum_{i=1}^{n} w_i$
- If we use $P = \max_i(w_i)$ processors, the algorithm runs in exactly T steps
- But the cost of this execution may be $\omega(W)$
- For example, say $w_1 = n \log n$, while $w_i = n$, for $i > 1$, and $T = \log n$
- The above execution takes $T = \log n$ time with $w_1 = n \log n$ processors
- The cost is $O(n \log^2 n)$, whereas $W = O(n \log n)$

PREFIX SUMS

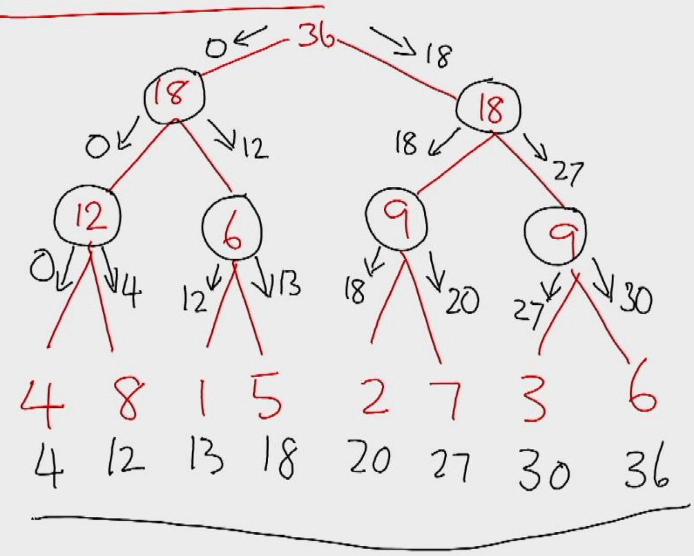
A = 4 8 1 5 2 7 3 6

B = 4 12 13 18 20 27 30 36

$B[i] = \sum_{j=1}^{i} A[i]$

Seq. complexity $= O(n)$

EREW PRAM



At each node i $\qquad n = 2^k$

$L[i]$, $\underline{\underline{M[i]}}$, $R[i]$

Copy A to the M[ ] of the leaves

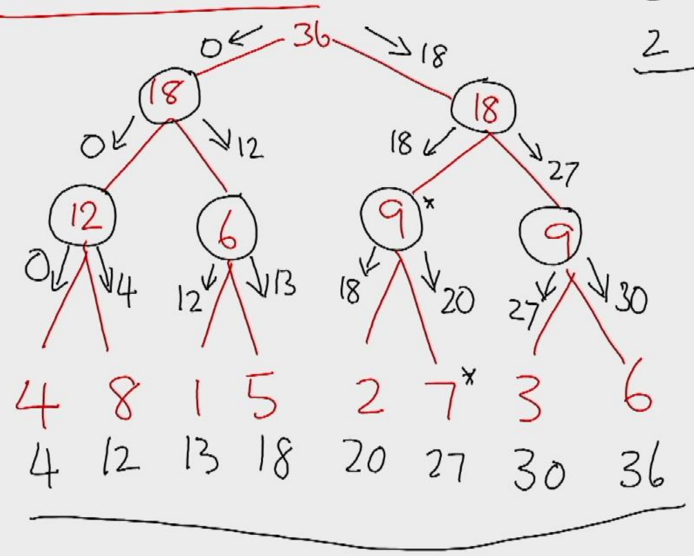
B : binary tree $\quad |B| = 2n-1$

Bottom to top phase

$$M[i] = M[2i] + M[2i+1]$$

$$L[i] = M[2i]$$

EREW PRAM



L	M	R
2	9	18

M	R
7	20

27

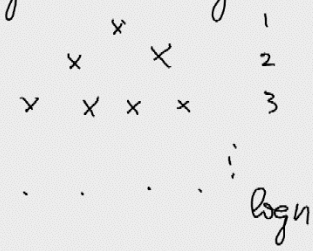
Top to Bottom phase

for each i,

$R[2i+1] = R[i] + L[i]$

$R[2i] = R[i]$

$j = 1$ to $\log n$



Leaves

$= R[i] + M[i]$

Prefix Sums 1

Input: Array $A[1 \ldots n]$ of integers. For simplicity, assume that n is a power of 2.

Output: An array $B[1 \ldots n]$ such that $B[i] = \Sigma_{j=1}^{i} A[j]$. Model: EREW PRAM.

```
{
    pardo for 1 ≤ I ≤ 2n − 1
        L[I] = M[I] = R[I] = 0;
    pardo for 1 ≤ i ≤ n
        M[n − 1 + I] = A[I];
    for s = 1 to log n do
        pardo for n/2^s ≤ I ≤ n/2^(s−1) − 1
            M[I] = (L[I] = M[2I]) + M[2I + 1];
```

n to $2n-1$

$1 \rightarrow n$

$2 \rightarrow n+1$

$\vdots$

$i \rightarrow n-1+i$

Prefix Sums 1 (Cont'd)

for $s = \log n$ to 1 do
    pardo for $n/2^s \le I \le n/2^{s-1} - 1$
    {
        $R[2I] = R[I]$;
        $R[2I+1] = L[I] + R[I]$;
    }
pardo for $1 \le I \le n$
    $B[I] = M[n-1+I] + R[n-1+I]$;
return B;
}

Prefix Sums 1

$1 \to n$
$2 \to n/2$
$\log n \to 1$
$\Big\}$ $O(n)$

$O(\log n)$ time
$O(n)$ cost

Input: Array $A[1 \dots n]$ of integers. For simplicity, assume that n is a power of 2.
Output: An array $B[1 \dots n]$ such that $B[i] = \Sigma_{j=1}^{i} A[j]$. Model: EREW PRAM.

{
pardo for $1 \le I \le 2n-1$
    $L[I] = M[I] = R[I] = 0$; $O(1)$ $O(n)$
pardo for $1 \le i \le n$
    $M[n-1+I] = A[I]$; $O(1)$ $O(n)$
for $s = 1$ to $\log n$ do
    pardo for $n/2^s \le I \le n/2^{s-1} - 1$
        $M[I] = (L[I] = M[2I]) + M[2I+1]$;

n to $2n-1$
$1 \to n$
$2 \to n+1$
$\vdots$
$i \to n-1+i$

Prefix Sums 1 (Cont'd)

for $s = \log n$ to 1 do
    pardo for $n/2^s \leq I \leq n/2^{s-1} - 1$
    {
        $R[2I] = R[I]$;
        $R[2I+1] = L[I] + R[I]$;
    }
pardo for $1 \leq I \leq n$
    $B[I] = M[n-1+I] + R[n-1+I]$;
return B;
}

1	1	
2	2	$O(n)$
3	4	
⋮		
$\log n$	n	
n	1	

$\frac{n}{\log n}$ processors

$O(\log n)$ time

Cost : $O(n)$

optimal

Compaction

	1	2	3	4	5	6	7	8	9
A	4			9			8		1

	1	2	3	4
B	4	9	8	1

Prefix Sums

C	1	0	0	1	0	0	1	0	1
D	1	1	1	2	2	2	3	3	4

with n processors

B	4	9	8	1

$O(\log n)$ time

$O(n)$ cost

Prefix Max

3 4 8 1 2 0 8 9

3 4 8 8 8 8 8 9

sum $\rightarrow$ max

$O(\log n)$ time

$n/\log n$ prcrs