CSE 6220 INTRODUCTION TO HIGH PERFORMANCE COMPUTING

PREFIX SUMS

Ümit V. Çatalyürek

School of Computational Science and Engineering

Georgia Institute of Technology

Prefix Sums Problem

Input n numbers: $x_0, x_1, x_2, \cdots, x_{n-1}$

Output: $S_0, S_1, S_2, \cdots, S_{n-1}$

$$S_i = \sum_{j=0}^{i} x_j$$

Best sequential algorithm

- $T(n, 1) = \Theta(n)$

```
S0 = x0
for i = 1 to n-1
    Si = Si-1 + xi
```

Prefix Sums in Parallel

Assume n = p = power of 2

Algorithm Alg-0:

$$\underbrace{x_0}_{S_0} \qquad \underbrace{x_0 \quad x_1}_{S_1} \qquad \underbrace{x_0 \quad x_1 \quad x_2}_{S_2} \qquad \cdots$$

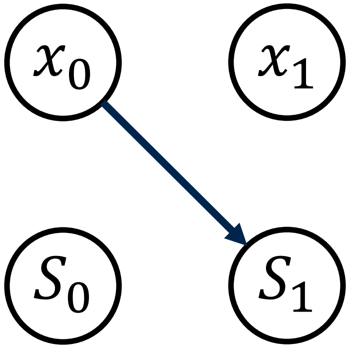
Parallel Prefix Sum Alg-0

- How many processors do we need?
- $p = 1 + 2 + 3 + \cdots + n = \frac{n(n+1)}{2}$
- What is the execution time?
- $T\left(n, \frac{n(n+1)}{2}\right) = \Theta(\log n)$
- $T\left(n, \Theta(n^2)\right) = \Theta(\log n)$
- $T(n, p) = \Theta\left(\frac{n^2 \log n}{p}\right)$

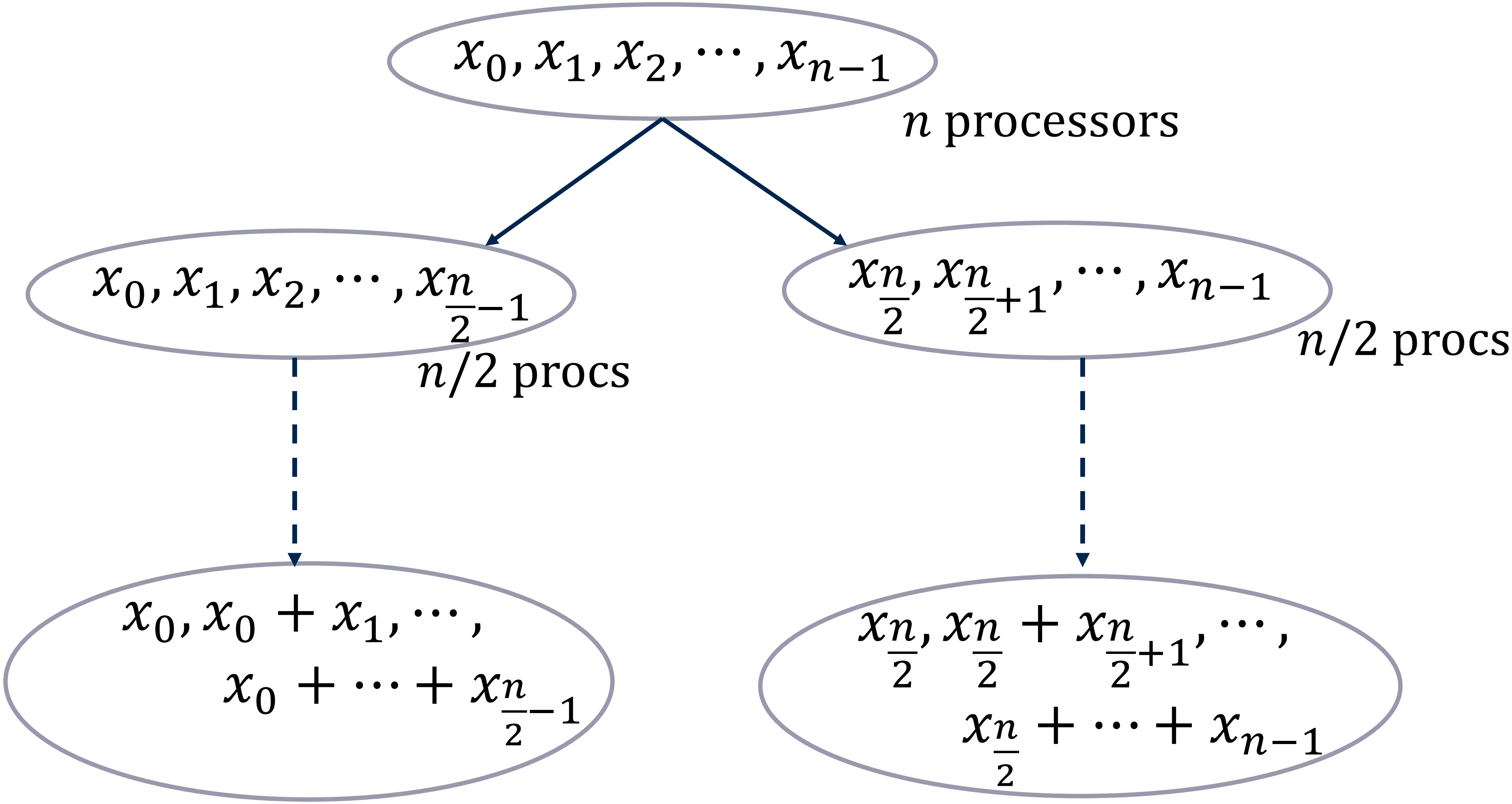
Parallel Prefix Sum Alg-1

- Use divide-and-conquer to develop new algorithm

- Base case



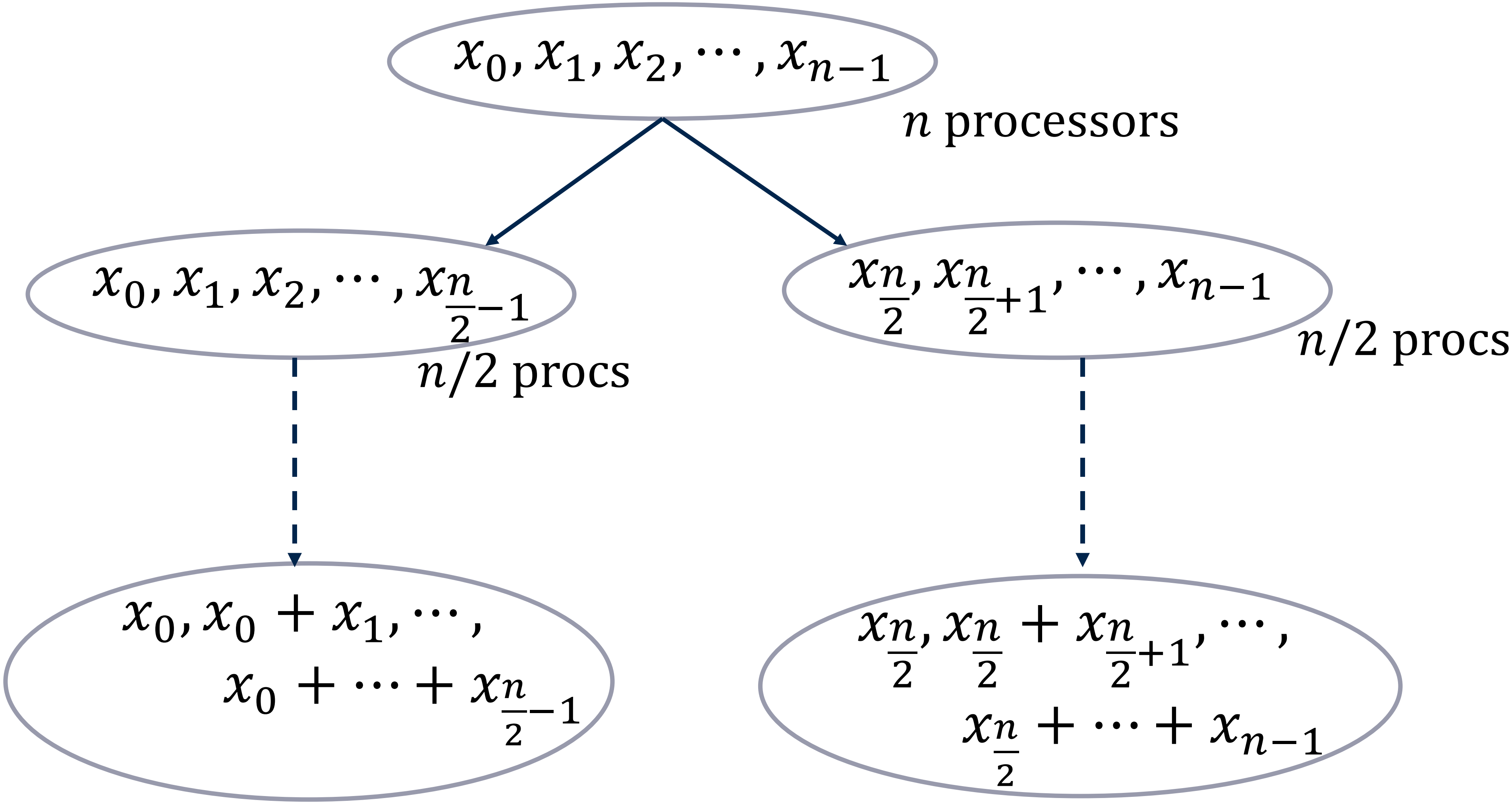
Parallel Prefix Sum Alg-1



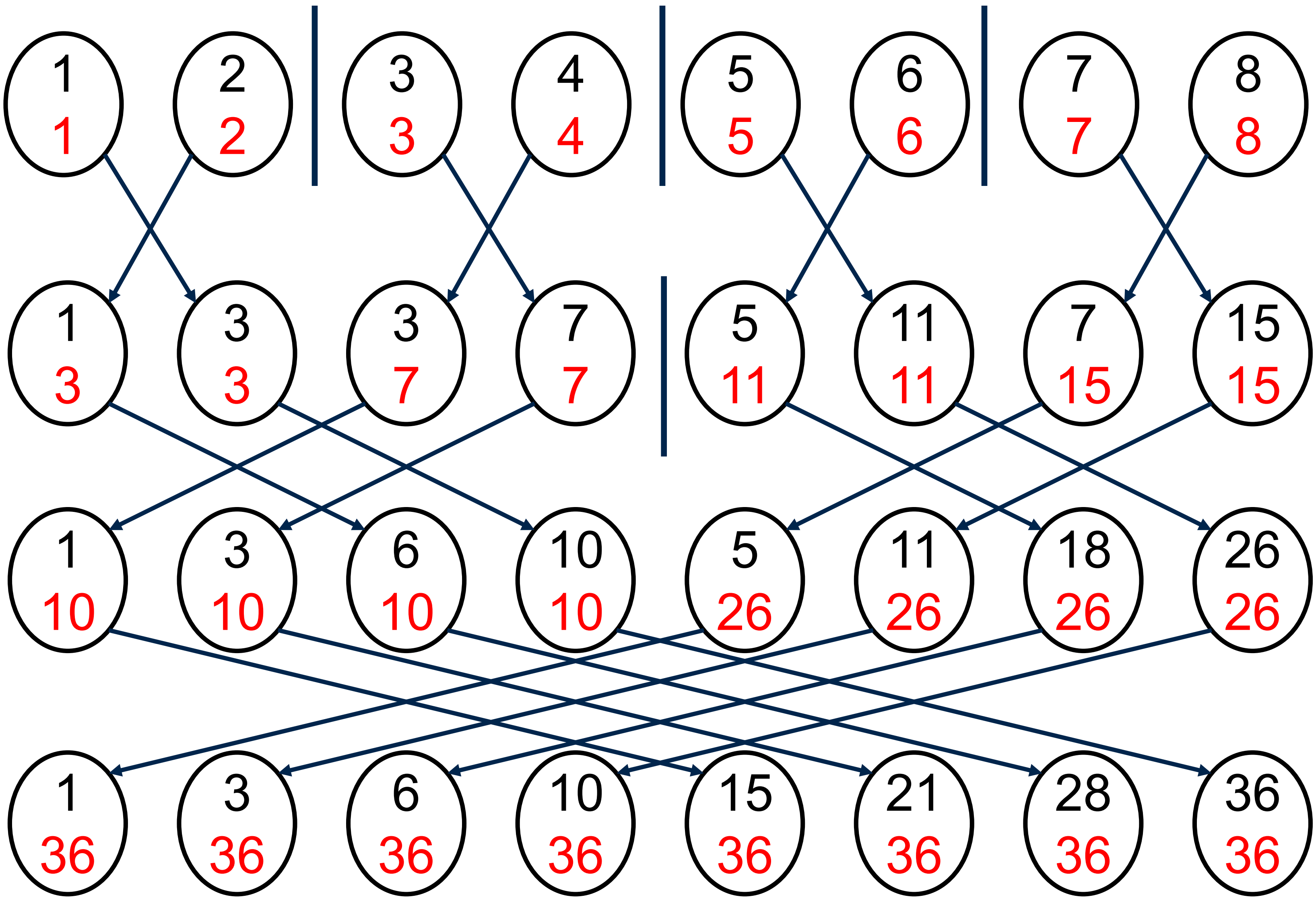
Parallel Prefix Sum Alg-1

- To merge $S_{\frac{n}{2}-1}$ needs to be communicated to all procs on the right.
- $T(n,n) = T\left(\frac{n}{2},\frac{n}{2}\right) + \Theta(\log n)$
- $\vdots$
- $T(n,n) = \Theta(\log^2 n)$
- Can we reduce this to $T(n,n) = \Theta(\log n)$?

Parallel Prefix Sum Alg-1



Parallel Prefix Sum Alg-2



Parallel Prefix Sum Algorithm (Alg-2)

```
Algorithm (for Pi)
total_sum ← prefix_sum ← local_number
for j=0 to d-1 do
    rank' ← rank XOR 2^j
    send total_sum to rank'
    receive received_sum from rank'
    total_sum ← total_sum + received_sum
    if (rank > rank')
        prefix_sum ← prefix_sum + received_sum
endfor
```

Parallel Prefix Sum

- $T(n, n) = \Theta(\log n)$
- What if $n > p$?

P_0				P_1				P_2				P_3			
x_0	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}
S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}	S_{11}	S_{12}	S_{13}	S_{14}	S_{15}

- Use Brent's Lemma?
- $T(n, p) = \Theta\left(\frac{n}{p} \log n\right)$

Parallel Prefix Sum Alg-3

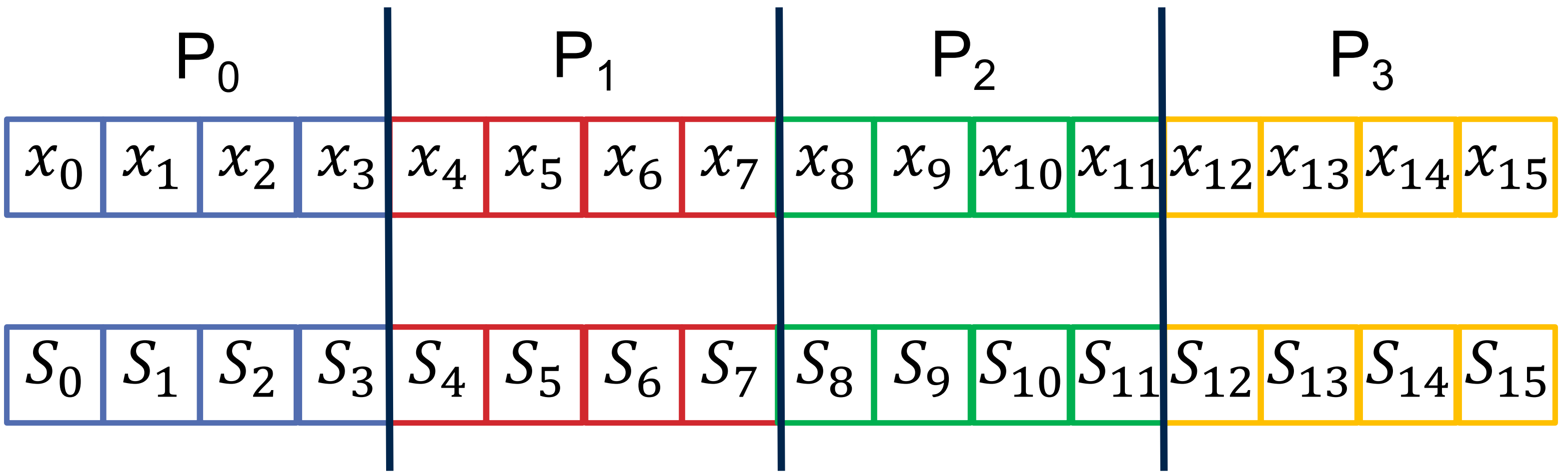
1. Compute prefix sum locally on each processor
2. Perform parallel prefix sum (Alg-2) using the last local prefix sum on each processor
3. Add the result of parallel prefix sum on a processor to each of its local prefix sum

Computation time = $\Theta\left(\frac{n}{p} + \log p\right)$

Communication time = $\Theta((\tau + \mu) \log p)$

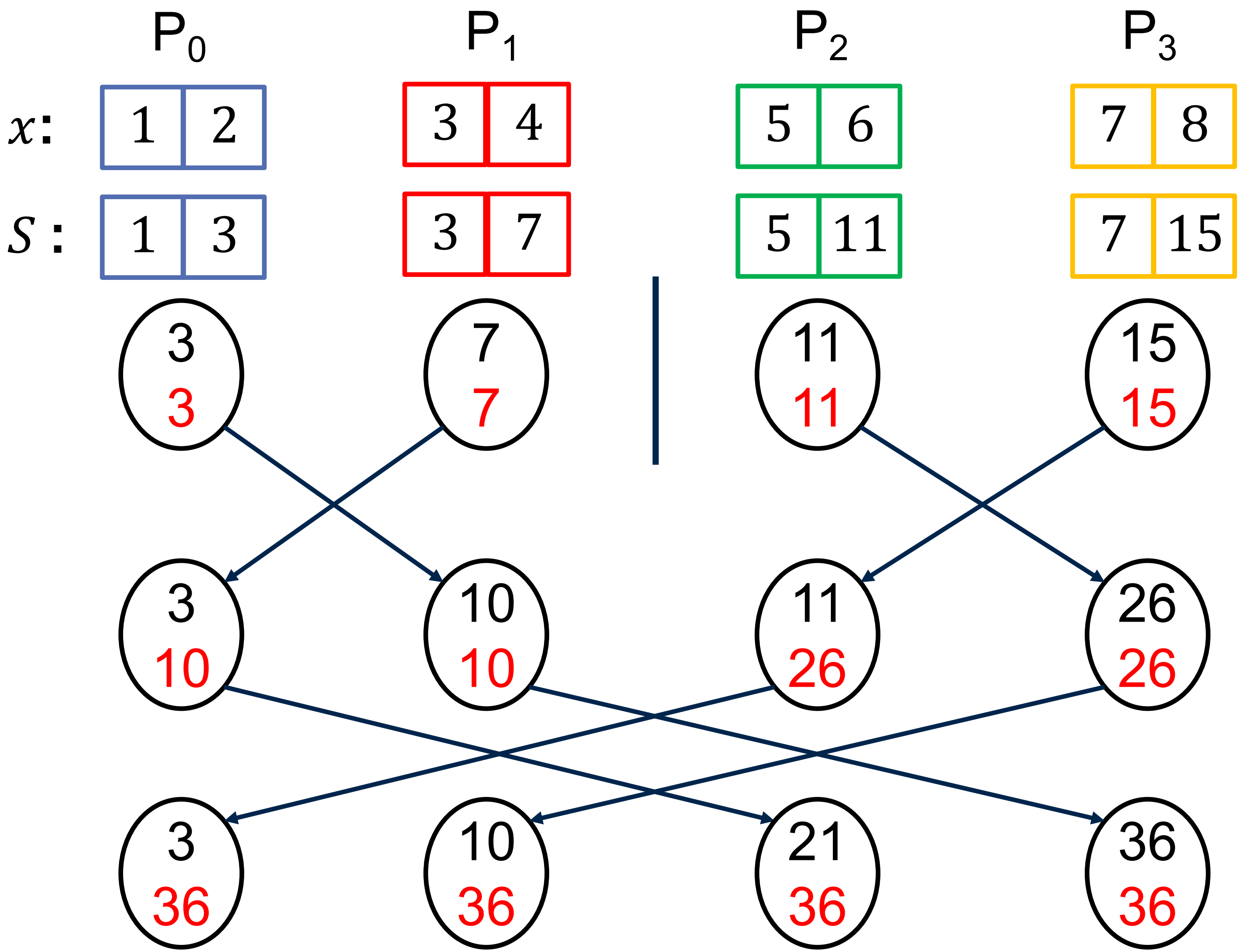
Is this algorithm correct?

Parallel Prefix Sum Alg-3



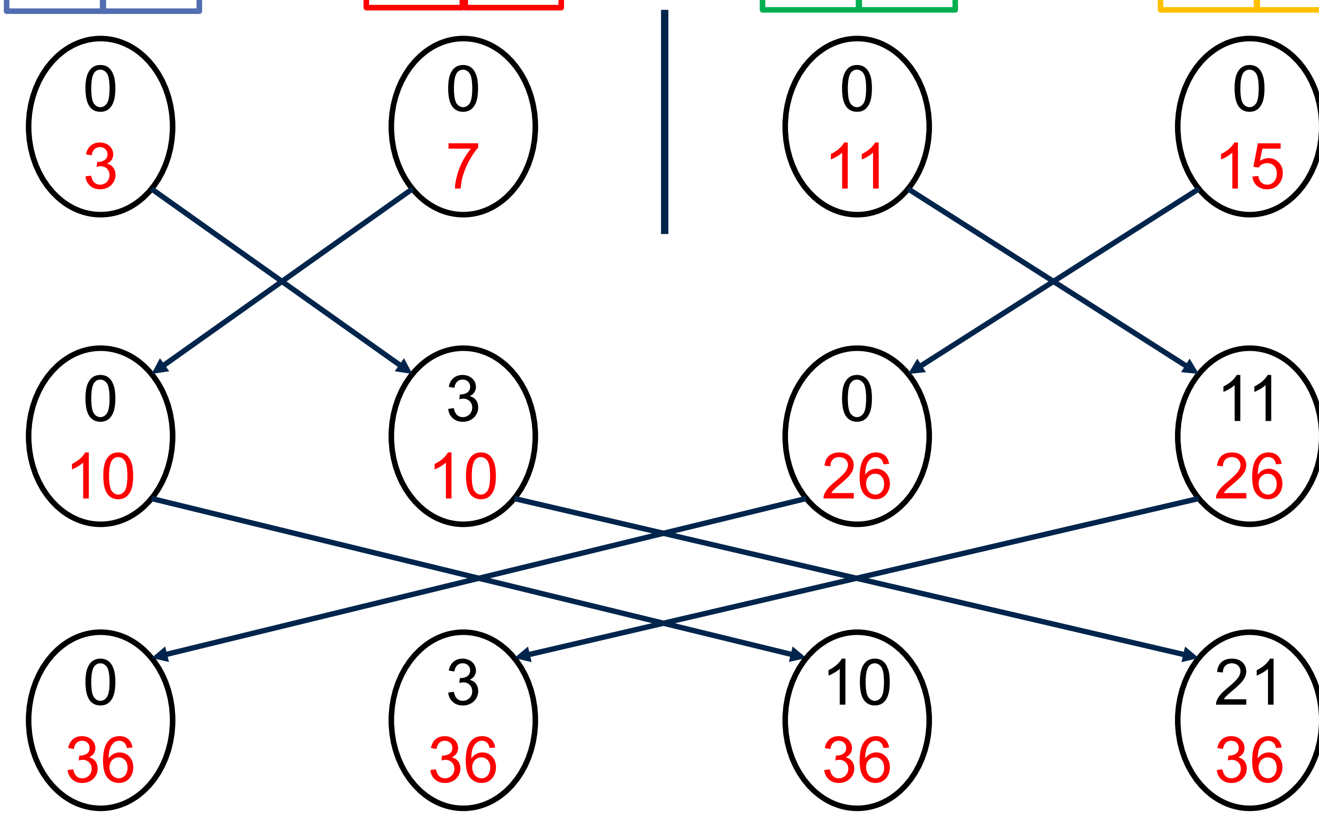
- What are we adding (on step 3) on each processor?
  - P_0 adds S_3
  - P_1 adds S_7
  - P_2 adds S_{11}
  - P_3 adds S_{15}
- Multiple alternative solutions, but here is a good/generic one:
  - **Modify Alg-2 to start with `prefix_sum` $\leftarrow$ 0**

Parallel Prefix Sum Alg-3



Parallel Prefix Sum Alg-3

	P_0		P_1		P_2		P_3	
x:	1	2	3	4	5	6	7	8
S :	1	3	3	7	5	11	7	15



Parallel Prefix Sum Alg-3

	P_0	P_1	P_2	P_3
x:	1 | 2	3 | 4	5 | 6	7 | 8
S :	1 | 3	3 | 7	5 | 11	7 | 15
	0 / 36	3 / 36	10 / 36	21 / 36
S :	1 | 3	6 | 10	15 | 21	28 | 36

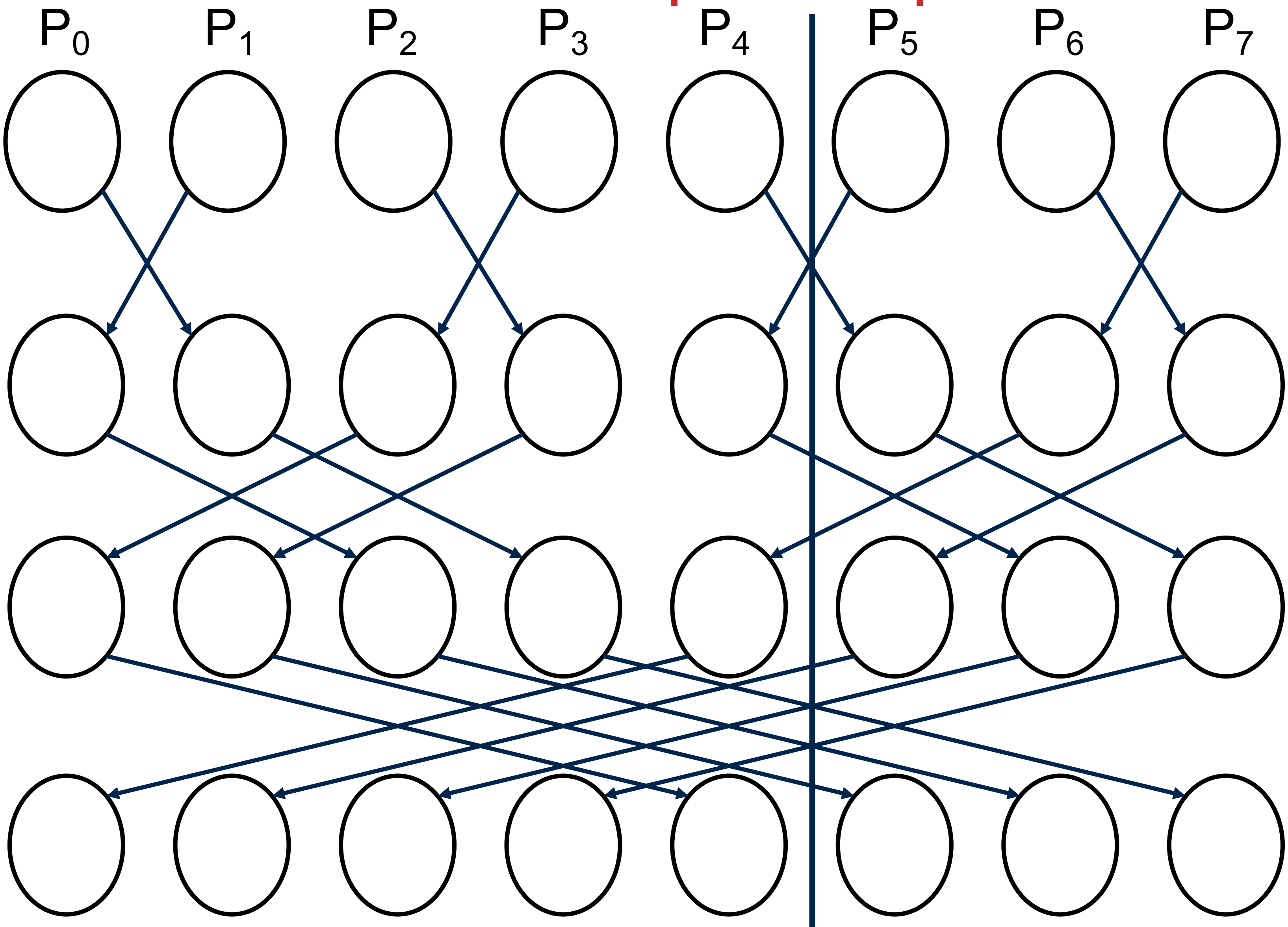
- What if n is not divisible by p?
  - Assign max $\left\lceil \frac{n}{p} \right\rceil$ to each processor: some processors will have 1 more element than the others.

Parallel Prefix Sum Alg-3

What if p is not a power of 2.

Find p' = a power of 2 such that $\frac{p'}{2} < p < p'$

Parallel Prefix when p is not power of 2



Parallel Prefix Sum Alg-3

What if p is not power of 2.

Find p' = a power of 2 such that $\frac{p'}{2} < p < p'$

Run your code like you have p' processors.

Ignore communications to/from non-existing processors, i.e., rank >= p.