# Parallel Processing

## Unit 3



Dr. Ahmed Sallam

# Fundamental GPU Algorithms

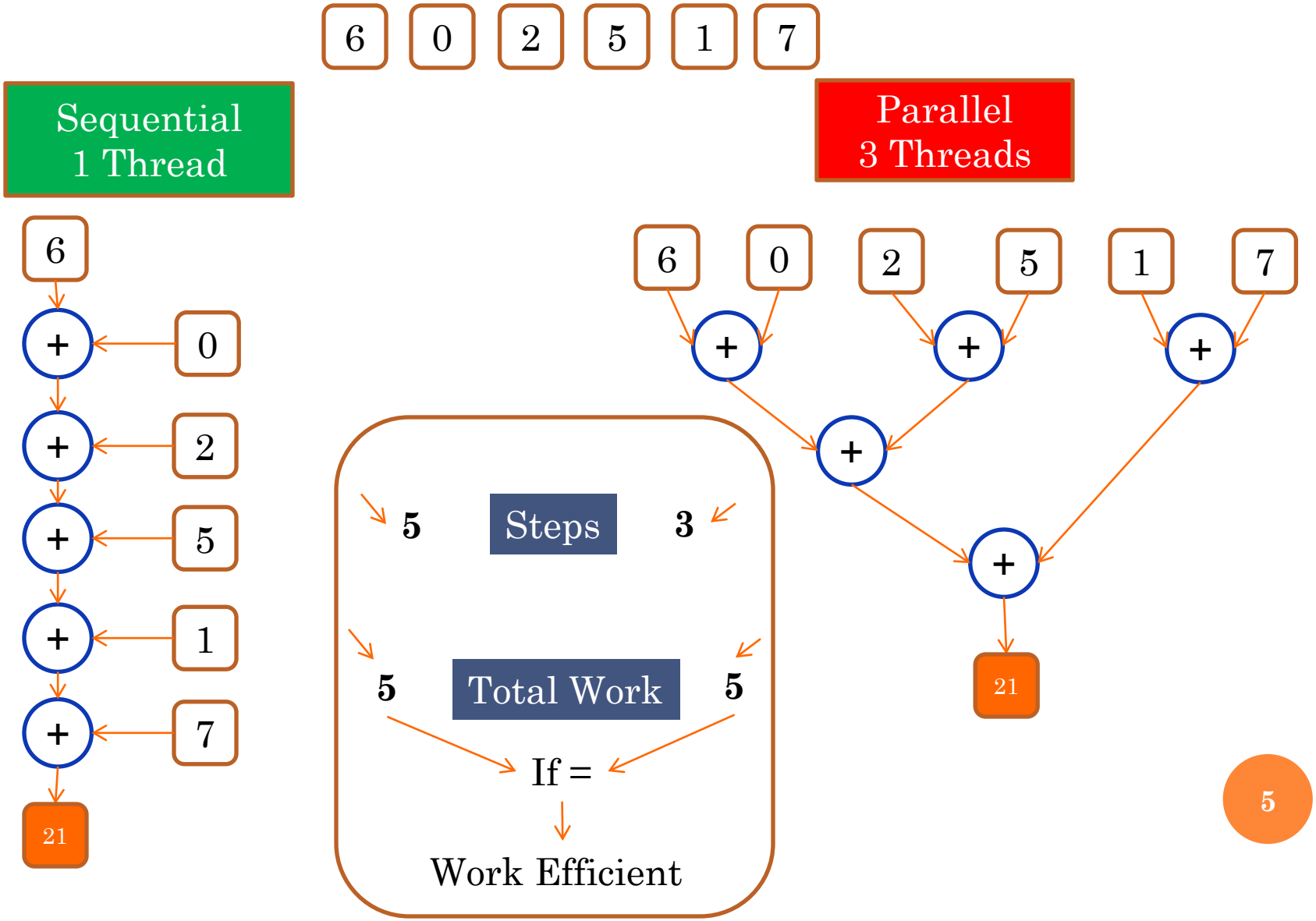
- More Patterns
  - Reduce
  - Scan

# Outlines

- Efficiency Measure
- Reduce primitive
  - Reduce model
  - Reduce Implementation and complexity analysis
- Scan primitive
  - Scan model
  - Scan Implementation and complexity analysis
- Histogram primitive
  - Histogram with atomic
  - Histogram with atomic and reduce

# Efficiency measure

# Sequential VS. Parallel efficiency?

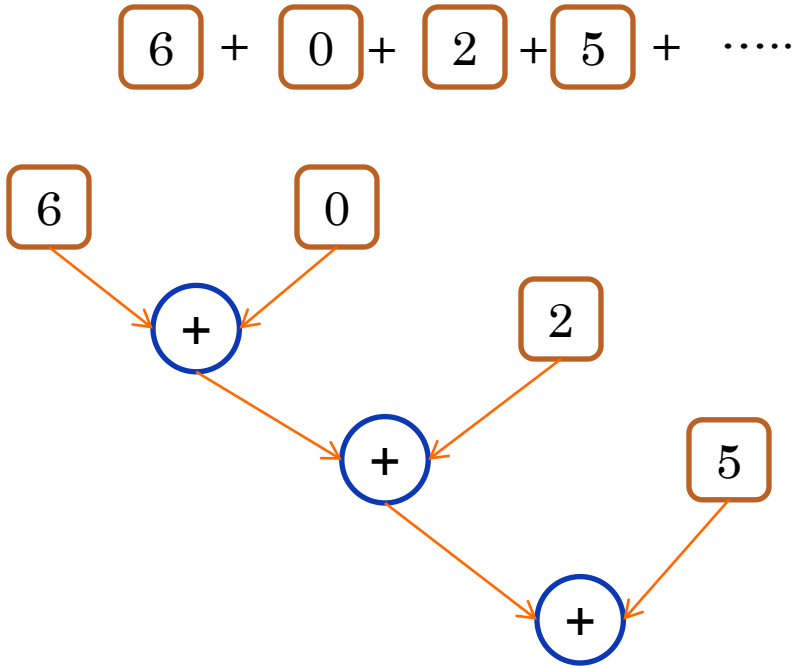


# Outlines

- Efficiency Measure
- Reduce Algorithm
  - Reduce model
  - Reduce Implementation and complexity analysis
- Scan Algorithm
  - Scan model
  - Scan Implementation and complexity analysis
- Histogram Algorithm
  - Histogram with atomic
  - Histogram with atomic and reduce

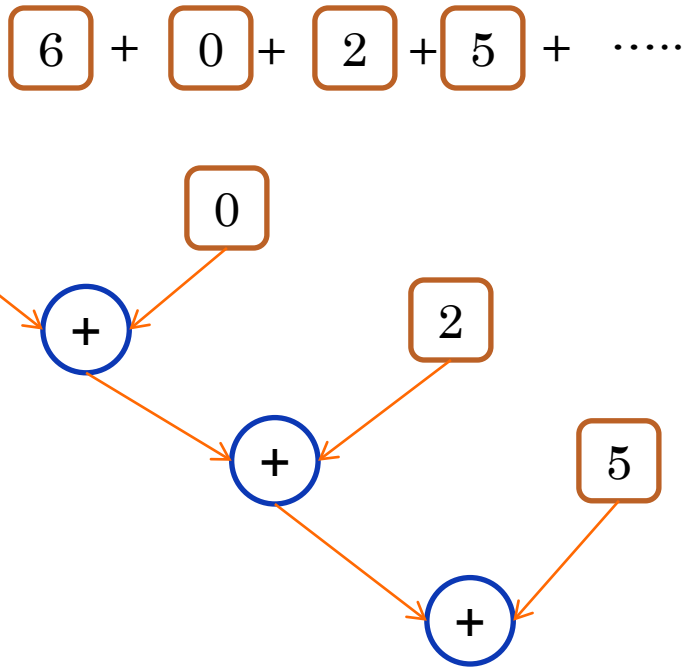
# Reduce Pattern

# Reduce



# Reduce Definition

- Set of elements
- Reduction operator
  - Binary
  - Associative operation

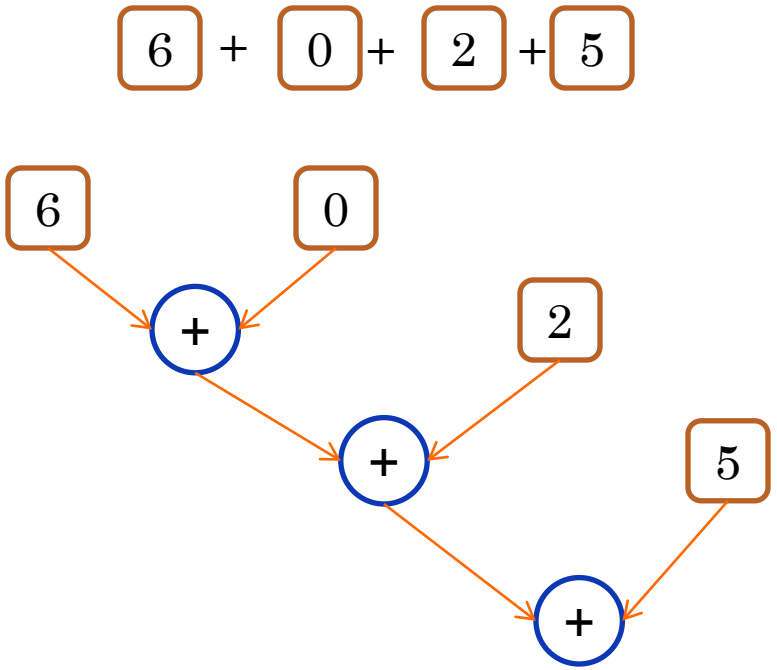


# Quiz

- ✓ Multiply ( a*b )
- ✓ Minimum
- Factorial ( a! )
- ✓ Logical Or ( a||b )
- ✓ Logical And ( a&&b )
- Division ( a/b )

# Reduce Serial (sequential) implementation

```
sum=0
For (i=1;i<n;i++){
        sum=sum+array[i];
}
return sum;
```




- Steps = 3
- Total work = 3

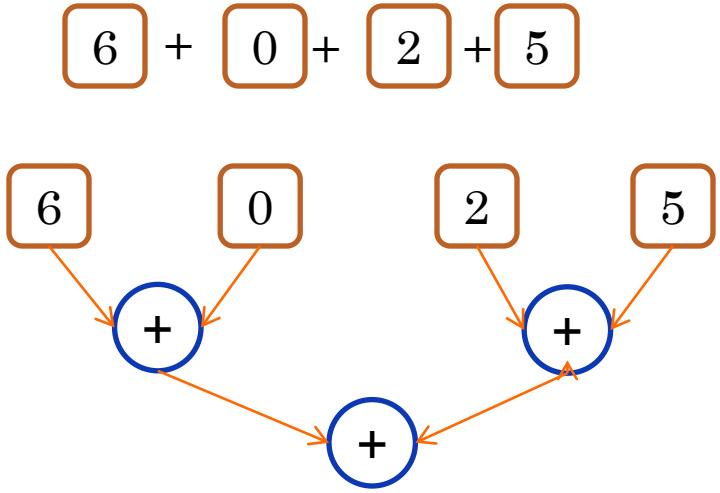
# Reduce Serial Complexity

Which is true to reduce n elements?

- ☐ It takes n operation
- ☑ It takes n-1 operation
- ☑ It's work complexity O(n)
- ☑ It's step complexity O(n)

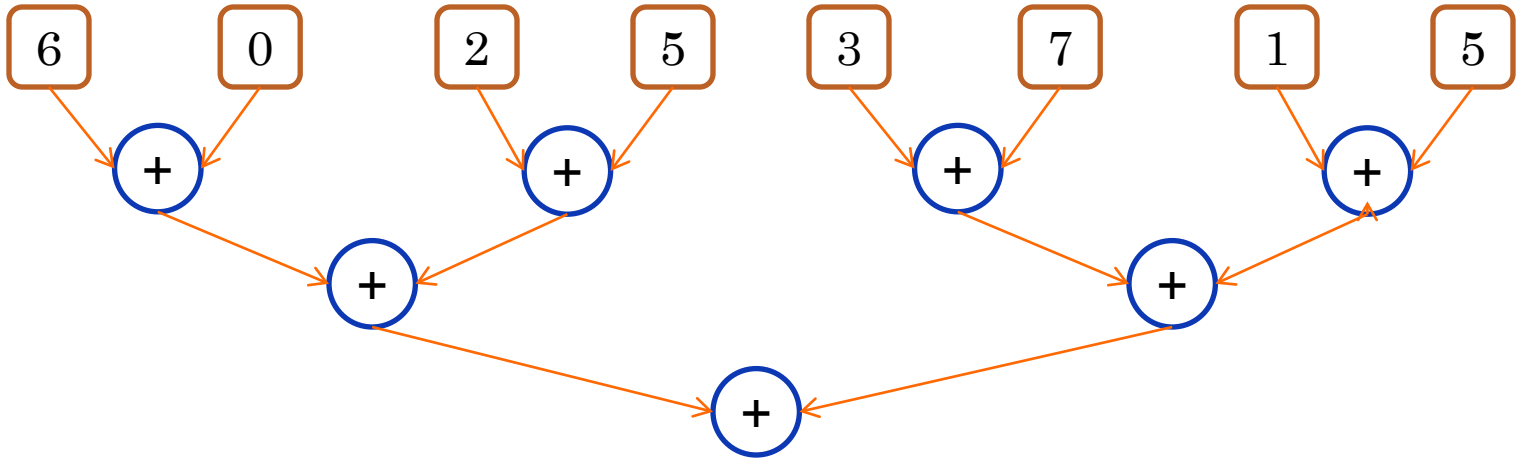
# Reduce Parallel implementation

- (a+(b+c))+d) = (a+b)+(c+d)




- Steps = 2
- Total work = 3

# Reduce Parallel complexity




| N | Steps | Total work |
|---|---|---|
| 2 | 1 | 1 |
| 4 | 2 | 3 |
| 8 | 3 | 7 |
| | = lg n | = n-1 |
| | O(log n) | O(n) |

Actually

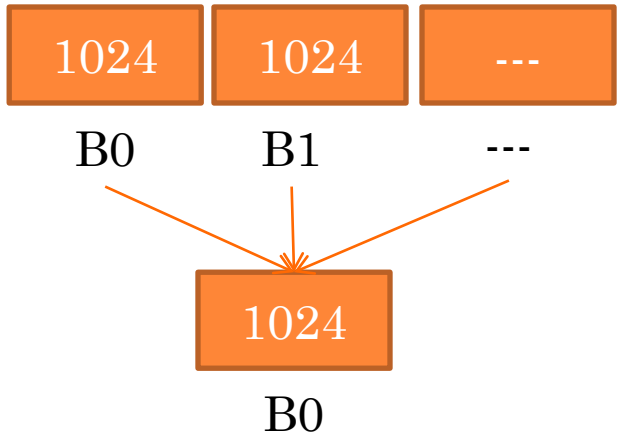
If we have n elements we need n/2 threads at first step. But this is not possible because we have only p processors

Thus O(log n) is not accurate and we need another calculation this called Brent's theorm

# Reduce in action

- Suppose we have 2^20 (~1m) elements
- Stage 1: 1024 block * 1024 thread
- Stage 2: 1 block
- PS: increase performance 3 times

  by using shared memory




Start

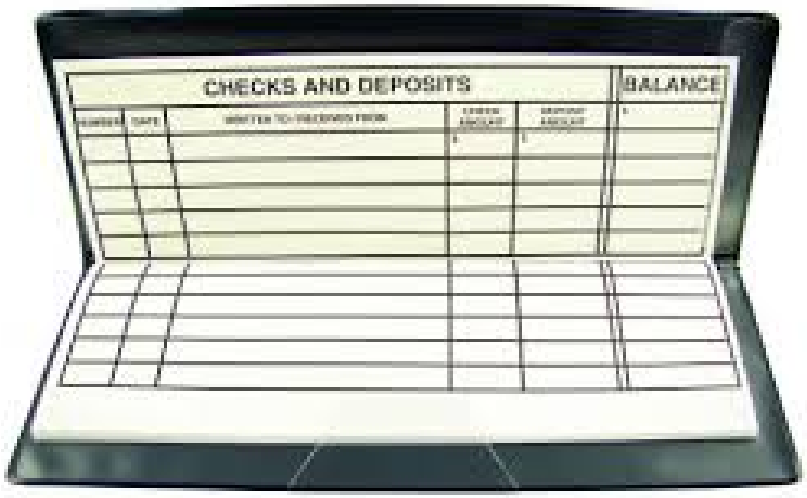
# Outlines

- Efficiency Measure
- Reduce Algorithm
  - Reduce model
  - Reduce Implementation and complexity analysis
- Scan Algorithm
  - Scan model
  - Scan Implementation and complexity analysis
- Histogram Algorithm
  - Histogram with atomic
  - Histogram with atomic and reduce

# Scan Pattern

# Scan

- Input: 1 2 3 4
- Op: add
- Output: 1 3 6 10




| Transaction | Balance |
|---|---|
| +10 | 10 |
| -5 | 5 |
| +4 | 9 |
| +3 | 12 |

# Scan Definition

- Set of elements
- Reduction operator (op)
  - Binary
  - Associative operation (We assume here it's also commutative e.g. x+y=y+x)
- Identity element [I op a= a]

| op | I | Because |
|---|---|---|
| + | 0 | a+0=a |
| * | 1 | a*1=a |
| min (unsigned char) | 0xFF | min(0xFF,a)=a |

# Scan Definition cont.

- Exclusive
  - in : 1 0 4 2 3
  - Op= + I= 0
  - Out: 0 1 1 5 7

- Inclusive:
  - in : 1 0 4 2 3
  - Op= + I= 0
  - Out: 1 1 5 7 11

# Scan Serial implementation and complexity

```
acc=identity
For (i=1;i<n;i++){
        acc=acc+array[i];
        out[i]=acc;
}
return out;
```

Inclusive

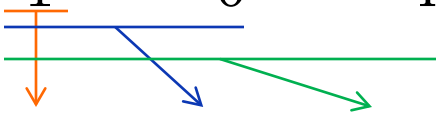
```
acc=identity
For (i=1;i<n;i++){
        out[i]=acc;
        acc=acc+array[i];
}
return out;
```

Exclusive

- Steps = n
- Total work = n

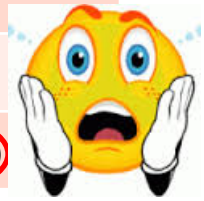
# Scan Parallel implementation and complexity

- Inclusive (+ Scan):
  - in : 1 0 4 2 3
  - Out: 1 1 5 7 10



- So if we consider the problem a set of reduce problems with different n then:

| n | Step | Work |
|---|---|---|
| 1 | lg 1 | 0 |
| 2 | lg 2 | 1 |
| … | … | … |
| n | lg n | n-1 |
| | = O ( log n) | = O ( $n^2$ ) |

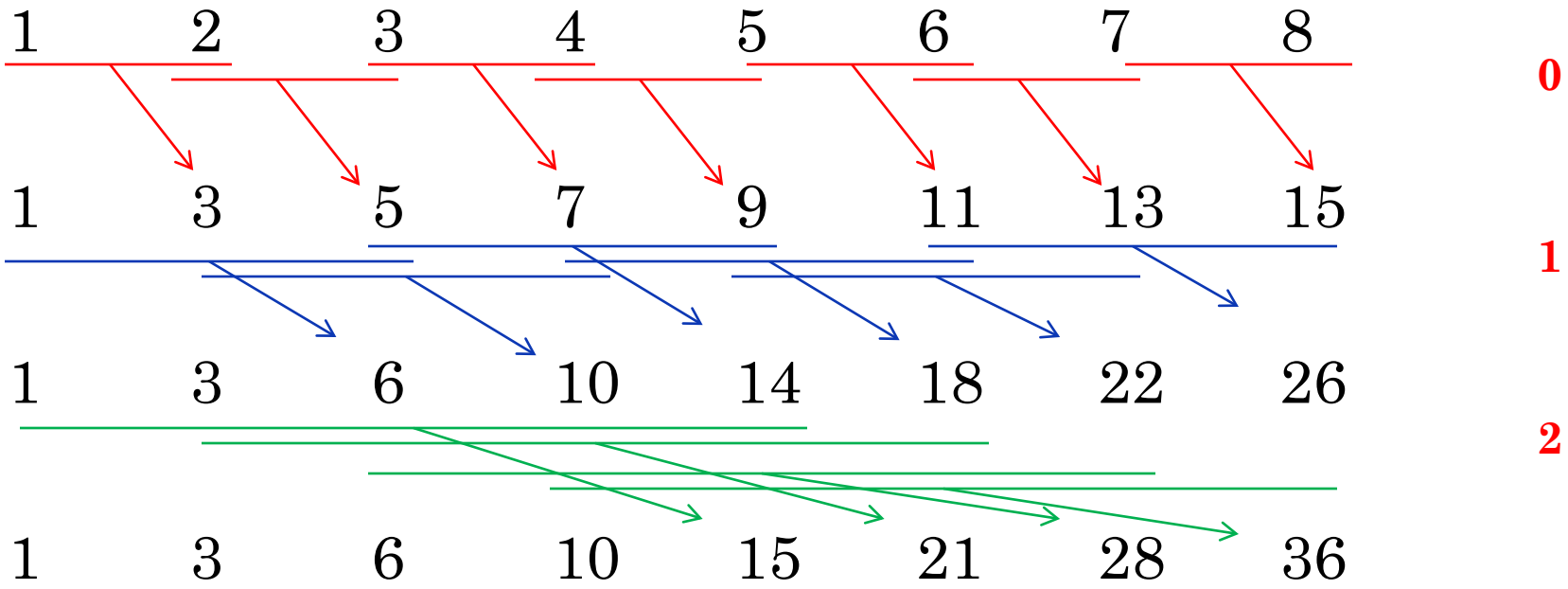


# Scan Parallel implementation and complexity (cont.1)

| Method | Step | Work |
| --- | --- | --- |
| Hillis & Steele | ✓ | |
| Blelloch | | ✓ |

# Scan Parallel implementation and complexity (cont.2)

- Hillis & Steele (Inclusive sum scan):

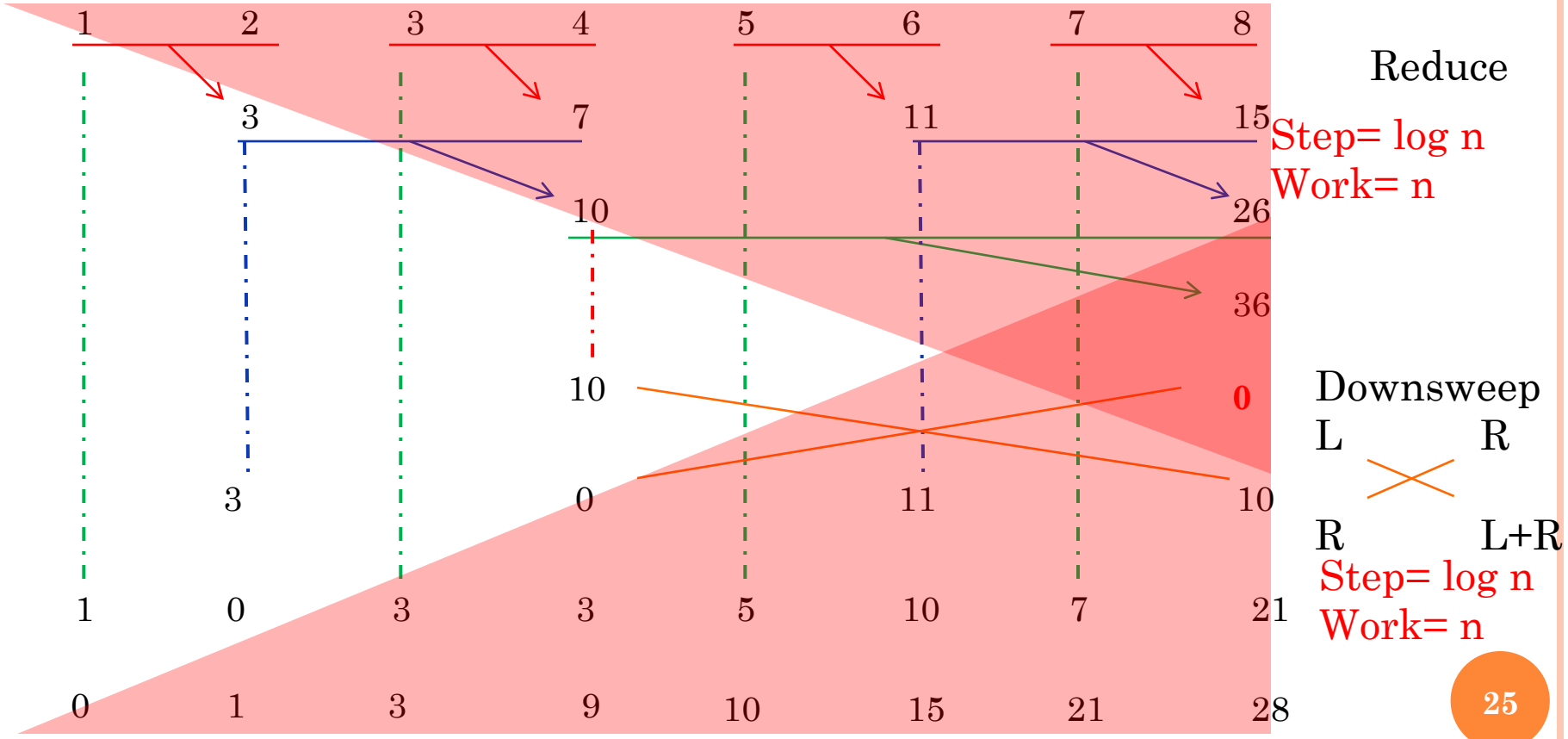



Step= O( log n )

work= matrix = O( n log n )

# Scan Parallel implementation and complexity (cont.3)

- Blelloch(Exclusive sum scan):



# Quiz

- Which Scan algorithm to use?

| | Serial | Hillis & Steele | Blelloch |
|---|---|---|---|
| 512 elements<br>512 processor | | ✓ | |
| 1m elements<br>512 processor | | | ✓ |
| 128k elements<br>1 processor | ✓ | | |

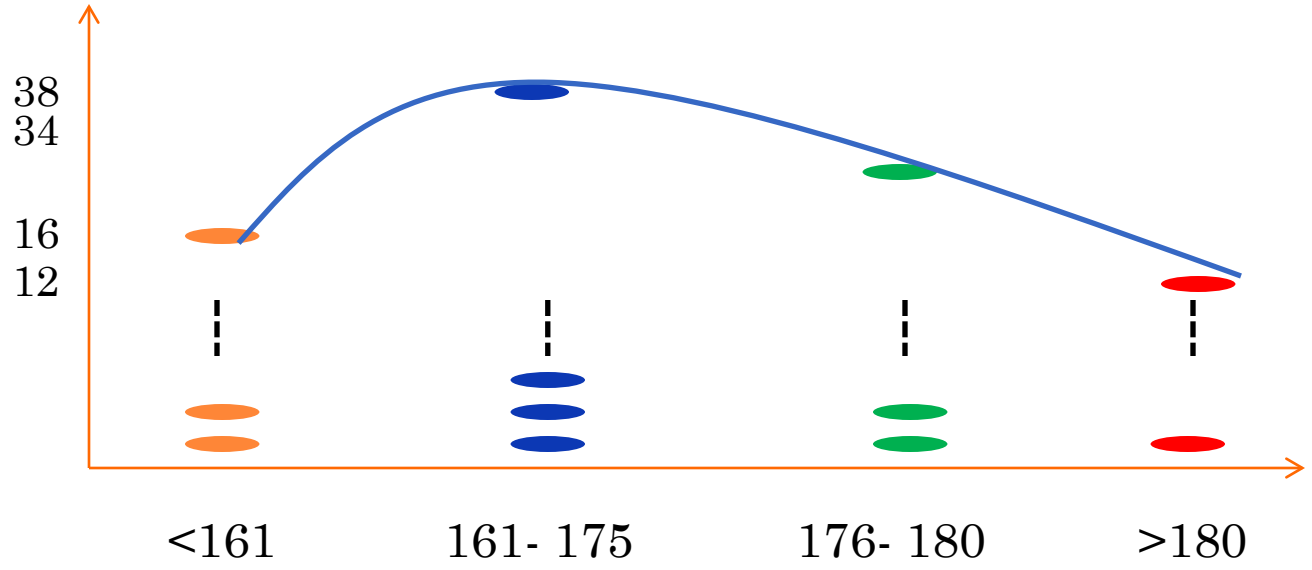
# Outlines

- Efficiency Measure
- Reduce Algorithm
  - Reduce model
  - Reduce Implementation and complexity analysis
- Scan Algorithm
  - Scan model
  - Scan Implementation and complexity analysis
- Histogram Algorithm
  - Histogram with atomic
  - Histogram with atomic and reduce

# Histogram

# Histogram

- Measure the students height in your class:




- How is shorter than 175?
  - Cumulative distribution (Scan)

# Histogram serial implementation

```
In: measurements[], n-elements
Out: result[]

For (i=0; i<bin-count; i++)
        result[i]=0;

For (i=0; i< n-elements ; i++)
        result[computeBin(measurements[i])]++
```

# Histogram parallel Naïve

```
In: measurements[], n-elements
Out: result[]

For (i=0; i<bin-count; i++)
        result[i]=0;

For (i=0; i< n-elements ; i++)
        result[computeBin(measurements[i])]++
```

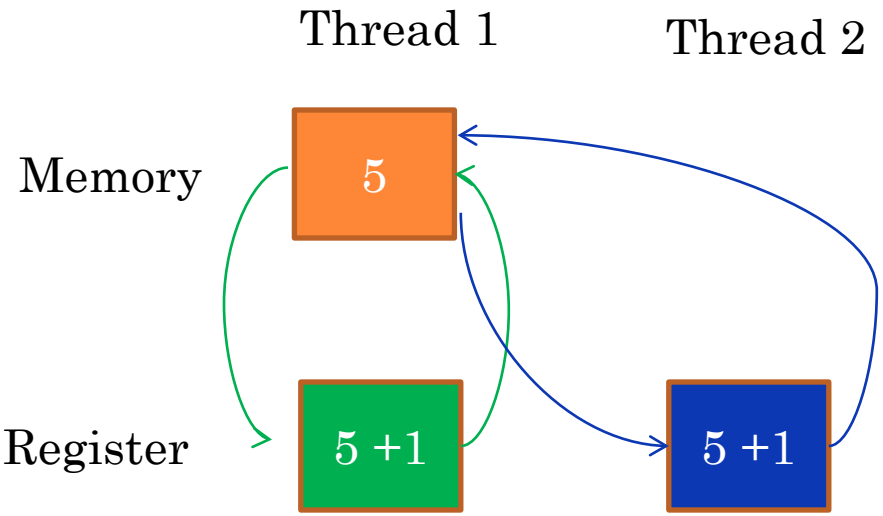
- We have synchronization problem

Start

# Histogram parallel Naïve (cont.)

- Example
  - 128 element  8 threads  3 bin


- Read
- Increment
- Write




Race condition

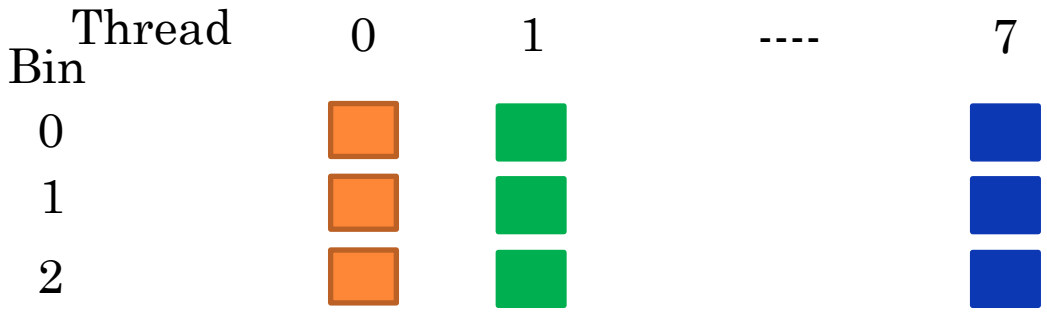
# HISTOGRAM PARALLEL SIMPLE

- Use atomic operation
- However this serialize the problem.

# Histogram parallel Reduce base

- Example
  - 128 element  8 threads  3 bin

- Use local bins, which means every thread had 3 local bin.

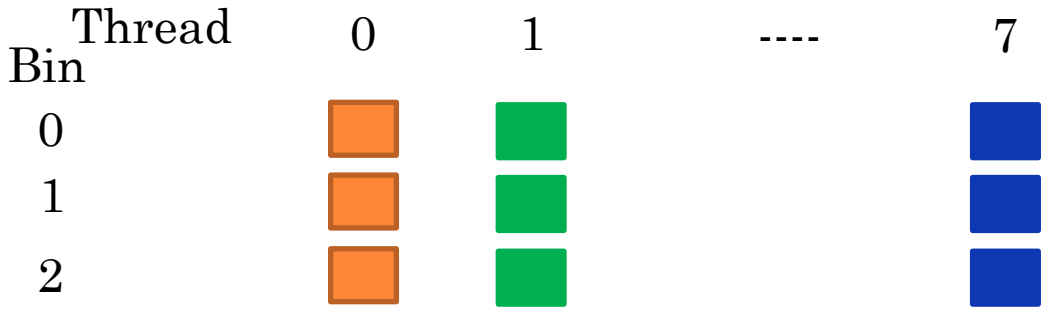



- Each thread accumulate 16 item
- Then apply a reduce back in global memory

# Histogram parallel Reduce base

- Example
  - 128 element  8 threads  3 bin

- Use local bins, which means every thread had 3 local bin.




- Each thread accumulate 16 item
- Then apply a reduce back in global memory 3 times for each bin (3 times is bad)

# Histogram parallel Sort & Reduce

- Example
  - 128 element  8 threads  3 bin


- First we sort


- Then we reduce

# Outlines

- Efficiency Measure
- Reduce Algorithm
  - Reduce model
  - Reduce Implementation and complexity analysis
- Scan Algorithm
  - Scan model
  - Scan Implementation and complexity analysis
- Histogram Algorithm
  - Histogram with atomic
  - Histogram with atomic and reduce

# Tone mapping

# Tone mapping

- Using  reduce, scan and histogram