Parallel Program Design

Jun Li, Department of Computer Science, CUNY Queens College

Foster's methodology

1. **Partitioning**: divide the computation to be performed and the data operated on by the computation into small tasks.

   The focus here should be on identifying tasks that can be executed in parallel.

Foster's methodology

2. **Communication**: determine what communication needs to be carried out among the tasks identified in the previous step.

Foster's methodology

3. **Agglomeration or aggregation: combine tasks and communications identified in the first step into larger tasks.**

   For example, if task A must be executed before task B can be executed, it may make sense to aggregate them into a single composite task.

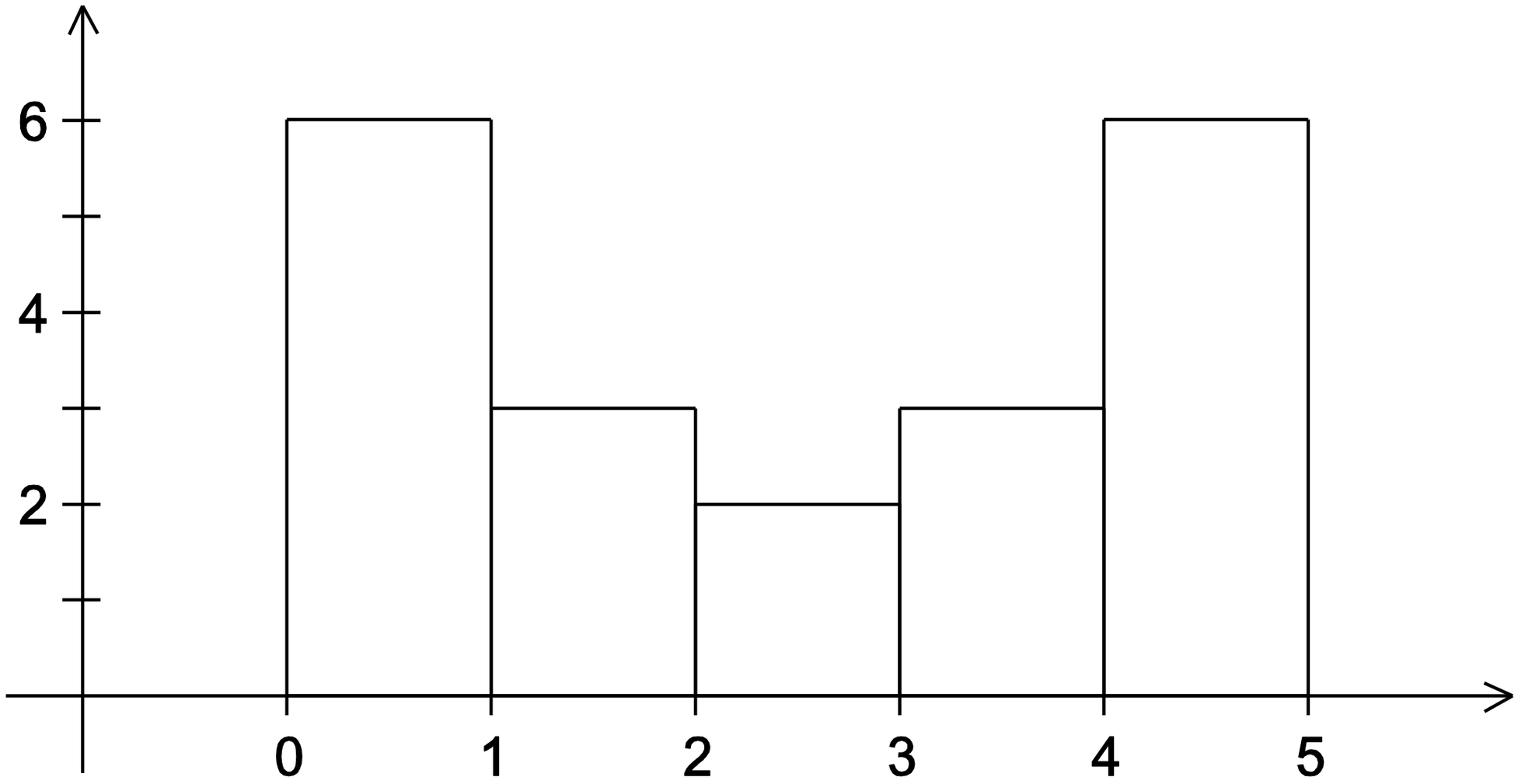
Foster's methodology

4. **Mapping**: assign the composite tasks identified in the previous step to processes/threads.

   This should be done so that communication is minimized, and each process/thread gets roughly the same amount of work.

Example - histogram

1.3, 2.9, 0.4, 0.3, 1.3, 4.4, 1.7, 0.4, 3.2, 0.3, 4.9, 2.4, 3.1, 4.4, 3.9, 0.4, 4.2, 4.5, 4.9, 0.9



Serial program - input

1. The number of measurements: data_count
2. An array of data_count floats: data
3. The minimum value for the bin containing the smallest values: min_meas
4. The maximum value for the bin containing the largest values: max_meas
5. The number of bins: bin_count

Serial program - output

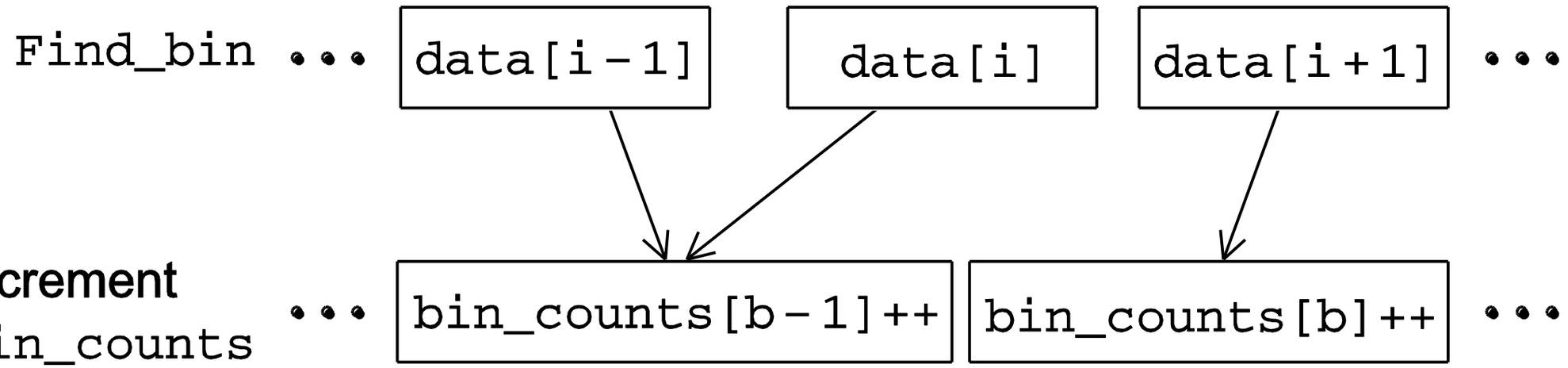
1. bin_maxes : an array of bin_count floats

2. bin_counts : an array of bin_count ints

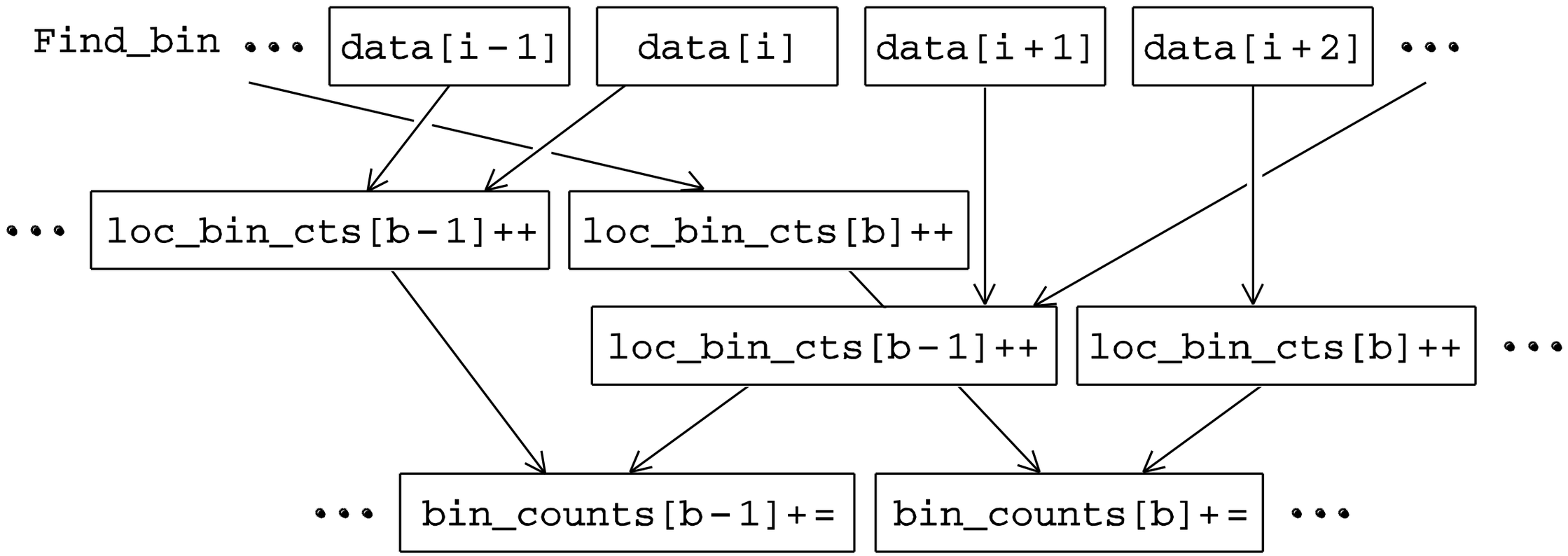
```
1 for (b = 0; b < bincount; b++)
2     bin_maxes[b] = min_meas + bin_width*(b+1);
```

```
1 for (i = 0; i < datacount; i++) {
2     bin = Find_bin(data[i], bin_maxes, bin_count, min_meas);
3     bin_counts[bin]++;
4 }
```

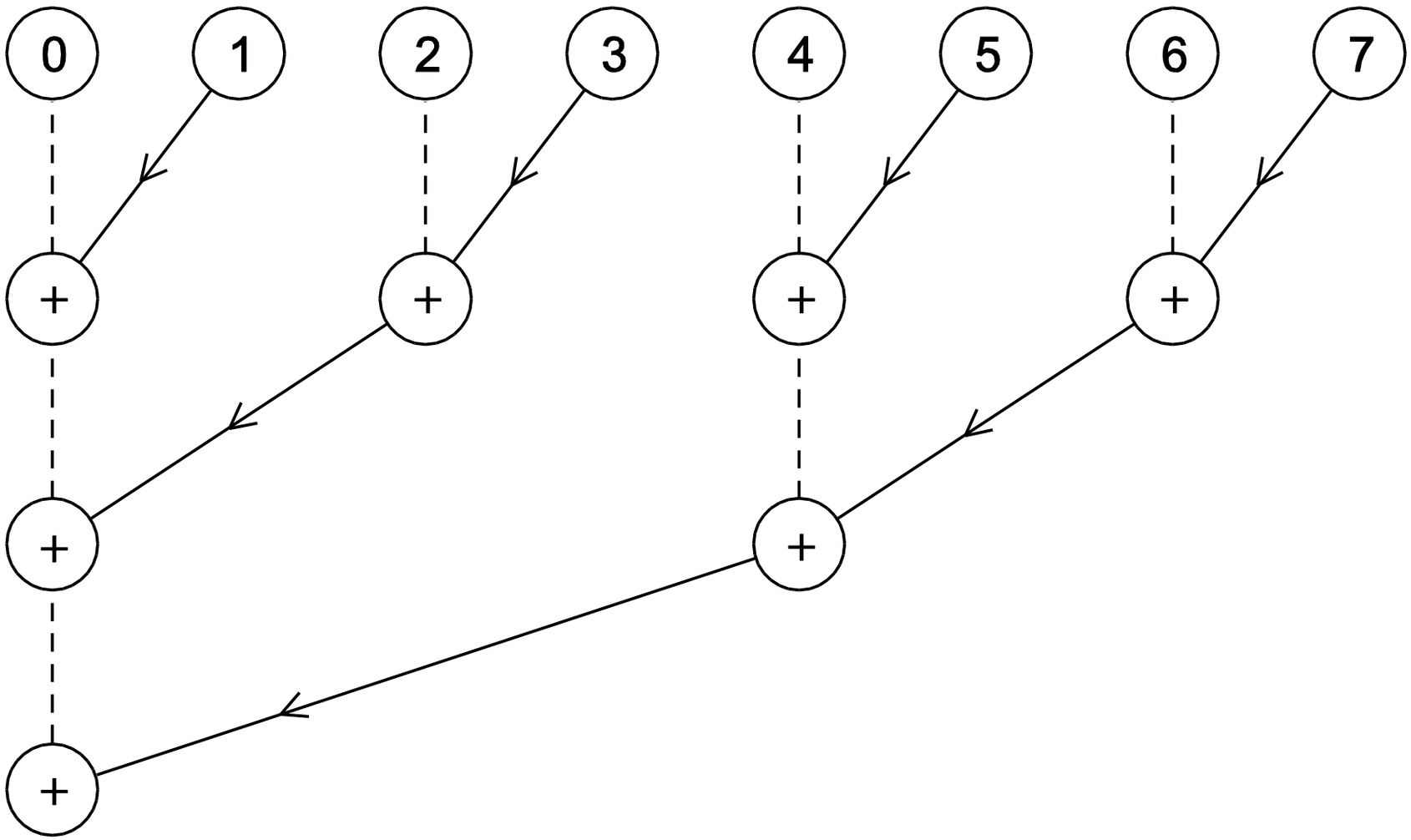
First two stages of Foster's Methodology



Alternative definition of tasks and communication



Adding the local arrays



Concluding Remarks (1)

Serial systems

The standard model of computer hardware has been the von Neumann architecture.

Parallel hardware

Flynn's taxonomy.

Parallel software

We focus on software for homogeneous MIMD systems, consisting of a single program that obtains parallelism by branching.

SPMD programs.

Concluding Remarks (2)

Input and Output

We'll write programs in which one process or thread can access stdin, and all processes can access stdout and stderr.

However, because of nondeterminism, except for debug output we'll usually have a single process or thread accessing stdout.

Concluding Remarks (3)

Performance

- Speedup
- Efficiency
- Amdahl's law
- Scalability

Parallel Program Design

- Foster's methodology