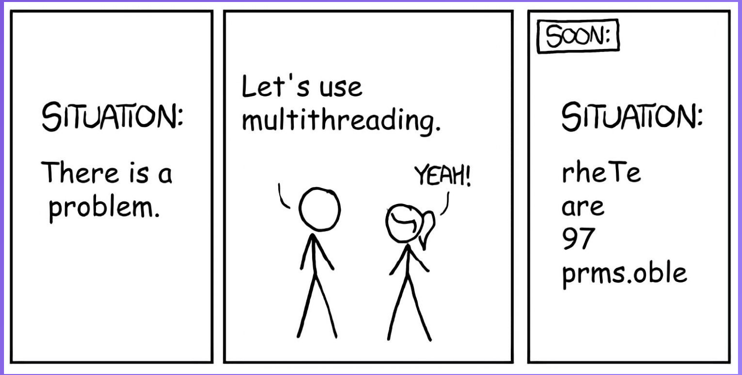
Parallel Programming

CSE 332 – Section 7

Slides by James Richie Sulaeman



ForkJoin

Suppose we wanted to take the sum of all the elements in the array `arr`

```
protected Integer compute() {
    // base case
    if (hi - lo <= CUTOFF) {
        return sequential(arr, lo, hi);         // sequentially handles arr[lo, ..., hi-1]
    }
    // recursive case
    int mid = lo + (hi - lo) / 2;
    SumTask left = new SumTask(arr, lo, mid);   // handles [lo, mid)
    SumTask right = new SumTask(arr, mid, hi);  // handles [mid, hi)

    left.fork();                           // fork a Thread (i.e. call compute() on a new Thread)
    int rightSum = right.compute();        // compute the right task using current Thread
    int leftSum = left.join();             // possibly wait and get result from left

    return rightSum + leftSum;             // combine outputs from left task and right task
}
```

ForkJoin

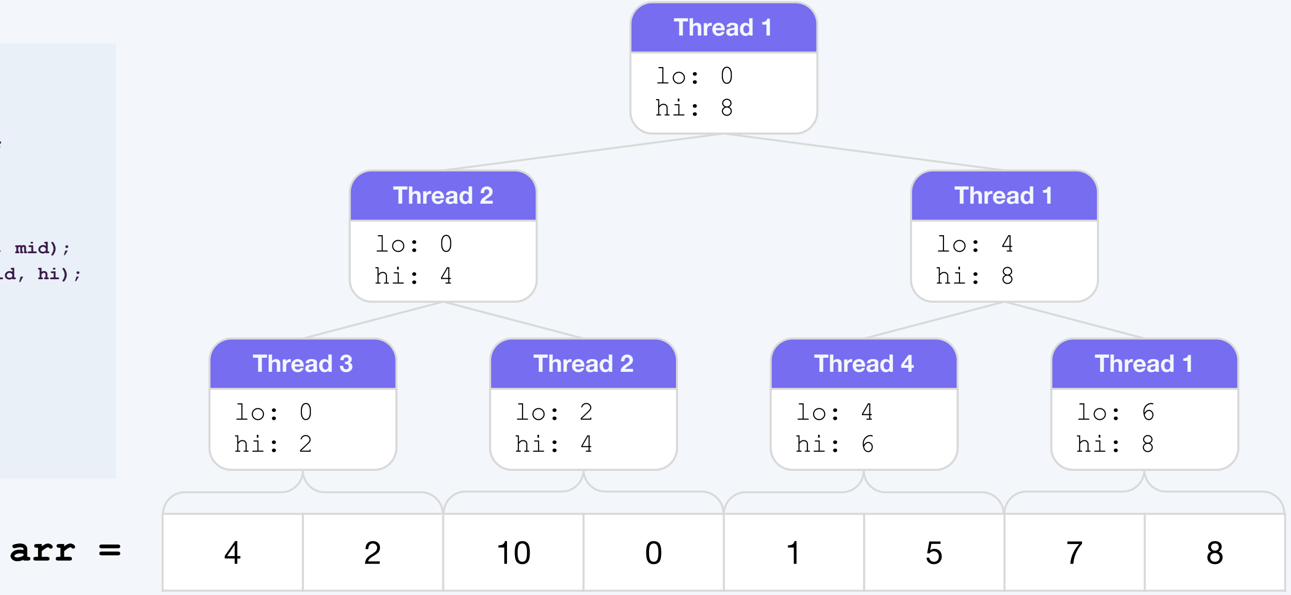
Suppose we wanted to take the sum of all the elements in the array `arr`

```
protected Integer compute() {
    // base case
    if (hi - lo <= CUTOFF) {
        return sequential(arr, lo, hi);
    }
    // recursive case
    int mid = lo + (hi - lo) / 2;
    SumTask left = new SumTask(arr, lo, mid);
    SumTask right = new SumTask(arr, mid, hi);

    left.fork();
    int rightSum = right.compute();
    int leftSum = left.join();

    return rightSum + leftSum;
}
```

`CUTOFF = 2`



Your Turn!

Download the code. The link is available on the schedule on the course web page.

Work in the following order:

1. LessThan7
2. PowMod
3. Parity
4. CountStrs
5. SecondSmallest

Thank You!