Synchronization: Finding the Max of an Array

CMPU 377
Parallel Programming

Introducing some notation

Finding the maximal element in an array is the problem. Let `m` be the variable to hold the max and `a` be the array of `n` values. Let the `forall` statement specify some number of concurrent iterations. Let $<$ and $>$ denote atomic actions. The goal of the program can be expressed in predicate logic as:

$$(\forall j : 1 \leq j \leq n : m \geq a[j]) \wedge (\exists j : 1 \leq j \leq n : m == a[j])$$

Let's look carefully at each successive version of this program, and see what we can discover regarding the issues of synchronization.

1 Sequential version

```
int m = 0;
for [i = 0 to n-1] {
  if (a[i] > m)
    m = a[i];
}
```

2 Version 2

Let's fully parallelize the loop with the forall statement, which executes all iterations in parallel:

```
int m = 0;
forall [i = 0 to n-1] {
  if (a[i] > m)
    m = a[i];
}
```

3 Version 3

Let's make the processes' actions atomic:

```
int m = 0;
forall [i = 0 to n-1] {
  < if (a[i] > m)
    m = a[i]; >
}
```

4 Version 4

Let's make only part of each processes' actions atomic:

```
int m = 0;
forall [i = 0 to n-1] {
  if (a[i] > m)
    < m = a[i]; >
}
```

5 Version 5

Let's combine parts of the last two versions:

```
int m = 0;
forall [i = 0 to n-1] {
  if (a[i] > m)
    < if (a[i] > m)
       m = a[i]; >
}
```