CS766: Analysis of concurrent programs 2023

Lecture 6: Linearizability

Instructor: Ashutosh Gupta

IITB, India

Compile date: 2023-02-16

Topic 6.1

UnboundedQueue

Example: UnboundedQueue

Here is an another concurrent implementation of queue

```
int q[LARGE]; // array initialized to zero
int back;

void* enq(int x) { // x > 0
  atomic{l = back++;}
  q[l] = x;
}

int deq() {
  while( true ) {
    l = back;
    for( i = 0 ; i < l; i ++ ) {
      atomic{ x=q[i]; q[i]=0; }                    //    atomic
      if ( x != 0) return x;
    }
  }
}
```

enq has two global actions.

deq has single effective action.

Example: an interesting execution

```
thread1: call enq(42)
thread1:   0 = back++;

thread2: call enq(24)
thread2:   1 = back++;
thread2:   q[1] = 24;
thread2: return;  //enq(24);
```

Status of array q after the above execution.

0	24	0	0	···	···

Example: two possible expansion of the execution

Execution 1

```
thread1:call enq(42)
thread1: 0 = back++;

thread2:call enq(24)
thread2: 1 = back++;
thread2: q[1] = 24;
thread2:return; //enq(24)

thread3:call deq()
thread3: 2 = back;
thread3: 0 = q[0];q[0]=0;
thread3: 24= q[1];q[1]=0;
thread3:return 24; //deq()

thread1: q[0] = 42;
```

Linearization point of enq(24)

Linearization point of enq(42)

Execution 2

```
thread1:call enq(42)
thread1:  0 = back++;

thread2:call enq(24)
thread2:  1 = back++;
thread2:  q[1] = 24;
thread2:return; //enq(24)

thread1:  q[0] = 42;

thread3:call deq()
thread3:  2 = back;
thread3:  42=q[0];q[0]=0;
thread3:return 42; //deq()
//
```

Linearization point of enq(42)

Linearization point of enq(24)

How to characterize the linearization point?

End of Lecture 6