Lecture 29   synchronization

01 October 2018   09:34

- spin locks, non-blocking locks, mutex
  - non-blocking: spinning
  - mutex: 0/1 condition non-spinning lock.
- condition variables
  - variables: used to inder blocked processes
  - condition: decide block/runnable states of processes

```
-   if
       (condition)        // not favourable
          wait (variable)
          sleep

    if (work done)
       change condition
       ~~wakeup~~
       signal (variable)
       wateup
```

```
- mutex_lock ( mutex * m)
  {
     spinlock (m->s);

     while ( m->L)
           sleep (m->id, m->s);

     m->L = 1;

     spin unlock (m->s);

  }
```

```
mutex_unlock (mutex * m).
 {
   spin lock (m->s);
   m->L = 0;
   ~~sig~~ wakeup (m->id);
   spin unlock (m->s);
```

```
struct mutex {
     int L;       // lock
     int id;      // variable
     spinlock s;
}
```

```
   sleep (m-> id, m-> s) {
lookup
 xv6   add toqueue (m->id, currproc);
                            ⇓
                        state = BLOCKED
        spinunlock (m->s);
        scheduler ();     // got of the CPU
   ⇒   spin lock (m->s);
   }
```

```
wakeup (m->id)
{                              lock;     <- seriable the PCB list / state
                                            (ptable.lock)
  ~~St~~
  remove from queue (m->id);
             ↓                 unlock;
       set state
       to READY
}
```

producer - consumer

bufsize = current size of produced elements.

```
producer () {               |   Consumer ()   {
```

```
producer() {


    spinlock(S);

        bufsize++;

        wakeup(NOBUF);

    spin unlock(S);
}
```

```
consumer()  {
    spinlock(S);
    while if (bufsize == 0)
            sleep(NO BUF, S   );

        bufsize--;

    spinunlock(S);

}
```