Assignment 2 Notes

Hyungmo Gu

November 4, 2020

1 Read and Writer

- Reader don't modify the data so we can have multiple readers, but only one writer
- Are examples of a common computing problem in concurrency
- Is a part of semaphore problem

2 Product/Consumer

```
producer() {
  while(1) {

    add_to_buf()

  }
}
```

```
consumer() {
  while(1) {

    remove_from_buf()

  }
}
```

- Is essentially how `pipes()` are implemented
- Has bounded buffer as a shared variable
  - Bounded buffer is also used when piping the output of one program into another

    Example
    `grep foo file.txt | wc -l`
    * `grep`
      · searches the input files for lines containing a match to a given pattern lis
      · when it finds a match in a line, it copies the line to standard output (by default)
    * `wc -l`
      · stands for word count
      · is used to find the number of lines (in this case)
    * `grep` is the producer
    * `wc` is the consumer
- Single buffer producer/consumer solution
  - Is to use two different conditial variables
    * Is nice, trouble free and simple

3 Condtional Variable

```
lock_acquire(lock);
while(condition not true) {
    cv_wait(cond, lock);
}
...  // do stuff

cv_signal(cond); //or cv_broadcast(cond)
lock_release(lock);
```

Condition

Conditional variable

- is a queue of waiting threads
- has two operations associated with it:
  1. `cv_wait(struct cv *cv, struct lock *lock)`
     - Is executed when a thread wishes to put itself to sleep
     - Releases lock, waits, re-acquires lock before return
       * Is to prevent race condtions from occuring when a thread is trying to put itself to sleep
  2. `cv_signal(struct cv *cv, struct lock *lock)`
     - Wakes <u>one</u> enqueued thread
  3. `cv_broadcast(struct cv *cv, struct lock *lock)` [from notes]
     - Wakes <u>all</u> enqueued threads
- If no one is waiting, signal or broadcast has no effect
- has rules
  - always use with `while` loops
    * on waking up, tread checks for condition in while loop
    * if condition is true, then thread goes back to sleep
- is always used together with locks

4 Semaphore

-