CSCI 4061
Locks, Lab 3 QAs

04/10/16
Threads

• **Review questions**
  1. Describe in your words what are Threads and when do you want to use them?
  2. Comparing Thread with Process
     • Why creating processes are more costly?
     • What do threads share whereas processes don’t (except for inheritance)? Candidates include: fd’s, signal actions, current working director, signal mask, Errno.
  3. Communication between Threads
     • Describe the differences between a) IPC **shared memory**, and b) **shared memory** across threads.
  4. Can you use **Mutex** for synchronization between processes?
Synchronization using Mutex

• **Critical section**: A section of code which works with shared resources can have unpredictable results depending on the order in which the threads execute.

• Mutual exclusion locks (**Mutexes**): A Mutex lets you lock a code section so that only one thread at a time executes the critical section.
The Pthreads API

- **Pthreads API**
  - **Thread management:** Routines that work directly on threads - creating, detaching, joining.
  - **Mutexes:** Routines for creating, destroying, locking and unlocking mutexes. These are supplemented by mutex attribute functions that set or modify attributes associated with mutexes.

<table>
<thead>
<tr>
<th>Routine Prefix</th>
<th>Functional Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>pthread_</td>
<td>Threads themselves and miscellaneous subroutines</td>
</tr>
<tr>
<td>pthread_attr_</td>
<td>Thread attributes objects</td>
</tr>
<tr>
<td>pthread_mutex_</td>
<td>Mutexes</td>
</tr>
<tr>
<td>pthread_mutexattr_</td>
<td>Mutex attributes objects</td>
</tr>
</tbody>
</table>
pthread_mutex_init, pthread_mutex_destroy

- #include <pthread.h>
  
  int pthread_mutex_init(pthread_mutex_t *mutex,
                        const pthread_mutexattr_t *attr);
  int pthread_mutex_destroy(pthread_mutex_t *mutex);

- Practice: complete dotprod_mutex_exercise.c
  - Master/Worker model
  - Make your Worker threads Joinable or Detaching?