Review Session for Midterm

Ennan Zhai

ennan.zhai@yale.edu
Midterm Structure

• Chapter 1-7 (Introduction - Deadlocks)
• Four parts (each part 25 points)
  - 3 short questions (3x3=9 points)
  - 5 multiple choice (2x5=10 points)
  - 1 longer question (6 points)

• All the questions are based on lecture slides
Homework
Chapter 1 & 2

- OS
- Kernel
- Real-time OS
- Microkernel
- System boot
- System calls
- System security and protection
- Interrupts
- Multiprocessor
Microkernel
System Boot

- When power initialized on system, execution starts at a fixed memory location
- Firmware ROM used to hold initial boot code
- Bootstrap loader is responsible for loading and starting the kernel
- See page 60 in cha2 for more details
Multiprocessor

<table>
<thead>
<tr>
<th>T₁</th>
<th>T₂</th>
<th>T₃</th>
<th>T₄</th>
<th>T₁</th>
<th>T₂</th>
<th>T₃</th>
<th>T₄</th>
<th>T₁</th>
<th>...</th>
</tr>
</thead>
</table>

| T₂ | T₄ | T₂ | T₄ | T₂ | T₄ | T₂ | T₄ | T₂ | ... |

| T₁ | T₃ | T₁ | T₃ | T₁ | ... |

| T₁ | T₃ | T₁ | T₃ | T₁ | ... |

| T₂ | T₄ | T₂ | T₄ | T₂ | ... |

| T₂ | T₄ | T₂ | T₄ | T₂ | ... |

| T₁ | T₃ | T₁ | T₃ | T₁ | ... |

| T₂ | T₄ | T₂ | T₄ | T₂ | ... |

```
T₁ T₂ T₃ T₄ T₁ T₂ T₃ T₄ T₁ ...

T₂ T₄ T₂ T₄ T₂ T₄ T₂ T₄ T₂ ...

T₁ T₃ T₁ T₃ T₁ ...

T₂ T₄ T₂ T₄ T₂ ...
```

**time**
Chapter 3 & 4

- What is process? thread? difference?
- Process states and components
- Zombie and Orphan
- Concurrency and Parallelism
- fork(); exec(); exit(); wait();
- Process direct and indirect communication
- Resources shared by threads
- Why we need thread?
- Job-scheduling and CPU-scheduling
Process States

- new
- admitted
- ready
- interrupt
- running
- exit
- terminated
- waiting
- I/O or event completion
- scheduler dispatch
- I/O or event wait
Process Components

- Stack:
- Heap:
- Data section:
- Program counter and register:
- Code:
Process Components

- Stack: Temporary data
- Heap: Dynamically allocated memory
- Data section: Global variables
- Program counter and register: Current activity
- Code: Program
System Call Example

- `fork()` system call creates new process
- `exec()` system call used after a `fork()` to replace the process’ memory space with a new program
System Call Example

```c
#include<stdio.h>
#include<unistd.h>

int main() {
    int i;
    for(i=0; i<3; i++)
        fork();
    return 0;
}
```
Direct & Indirect Communication

• What is direct communication?
• What is indirect communication?
• Why we need both?
Why we need threads?
Chapter 5

- HW and SW synchronization mechanisms
- Requirement for a solution to critical-section
- Mutex lock
- Semaphore (binary and counting)
- Starvation
- Preemptive and non-preemptive
- Synchronization problems
- Synchronization solutions
- Monitor
Types

• Hardware solutions
  - Test and set
  - Compare and swap

• Software solutions
  - Mutex lock
  - Semaphore

• Why we need both?
Algorithms

- Peterson’s algorithm (the basic version)
- Reader-writer algorithm
- Producer-consumer (bounded buffer)
- Dining philosopher
Chapter 6

- What is the scheduling criteria
- Recall process states and their transition
- Recall non-preemptive and preemptive
- Scheduling algorithms (Grantt chart)
- How to compute average waiting time
- Difference between multilevel Queue and multilevel feedback queue
Shortest Job Remaining First

<table>
<thead>
<tr>
<th>Process</th>
<th>Arrival time</th>
<th>Burst time</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P_1$</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>$P_2$</td>
<td>0</td>
<td>9</td>
</tr>
<tr>
<td>$P_3$</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>$P_4$</td>
<td>10</td>
<td>8</td>
</tr>
<tr>
<td>$P_5$</td>
<td>12</td>
<td>7</td>
</tr>
</tbody>
</table>
### Shortest Job Remaining First

<table>
<thead>
<tr>
<th>Process</th>
<th>Arrival time</th>
<th>Burst time</th>
</tr>
</thead>
<tbody>
<tr>
<td>P₁</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>P₂</td>
<td>0</td>
<td>9</td>
</tr>
<tr>
<td>P₃</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>P₄</td>
<td>10</td>
<td>8</td>
</tr>
<tr>
<td>P₅</td>
<td>12</td>
<td>7</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>P₁</th>
<th>P₃</th>
<th>P₁</th>
<th>P₄</th>
<th>P₅</th>
<th>P₂</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>5</td>
<td>7</td>
<td>10</td>
<td>18</td>
<td>25</td>
</tr>
<tr>
<td>34</td>
<td>25</td>
<td>18</td>
<td>10</td>
<td>7</td>
<td>5</td>
</tr>
</tbody>
</table>
Chapter 7

• Difference between starvation and deadlock
• Deadlock features
• Resource allocation graph
• Deadlock prevention and avoidance
• Banker’s algorithm
• Deadlock detection
• Deadlock detection algorithm