Processes and Threads

Major Requirements of an Operating System

- Interleave the execution of several processes to maximize processor utilization while providing reasonable response time
- Allocate resources to processes
- Support interprocess communication and user creation of processes

Processes and Threads

- Processes:
  - Also called a task or job
  - Execution of an individual program
  - "Owner" of resources allocated for program execution
  - Encompasses one or more threads

- Threads:
  - Unit of execution
  - Can be traced
    - list the sequence of instructions that execute
  - Belongs to a process

Logical Execution Trace

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>5000</td>
<td>B000</td>
<td>12000</td>
</tr>
<tr>
<td>5001</td>
<td>B001</td>
<td>12001</td>
</tr>
<tr>
<td>5002</td>
<td>B002</td>
<td>12002</td>
</tr>
<tr>
<td>5003</td>
<td>B003</td>
<td>12003</td>
</tr>
<tr>
<td>5004</td>
<td>B004</td>
<td>12004</td>
</tr>
<tr>
<td>5005</td>
<td>B005</td>
<td>12005</td>
</tr>
<tr>
<td>5006</td>
<td>B006</td>
<td>12006</td>
</tr>
<tr>
<td>5007</td>
<td>B007</td>
<td>12007</td>
</tr>
<tr>
<td>5008</td>
<td>B008</td>
<td>12008</td>
</tr>
<tr>
<td>5009</td>
<td>B009</td>
<td>12009</td>
</tr>
<tr>
<td>5010</td>
<td>B010</td>
<td>12010</td>
</tr>
<tr>
<td>5011</td>
<td>B011</td>
<td>12011</td>
</tr>
</tbody>
</table>

(a) Trace of Process A  (b) Trace of Process B  (c) Trace of Process C

Execution snapshot of three single-threaded processes (No Virtual Memory)

Combined Traces

(Actual CPU Instructions)

What are the shaded sections?
Summary: The Process Model

- Multiprogramming of four programs
- Conceptual model of 4 independent, sequential processes (with a single thread each)
- Only one program active at any instant

Process and thread models of selected OSes

- Single process, single thread
  - MS-DOS
- Single process, multiple threads
  - OS/161 as distributed
- Multiple processes, single thread
  - Traditional Unix
- Multiple processes, multiple threads
  - Modern Unix (Linux, Solaris), Windows 2000

Note: Literature (incl. Textbooks) often do not cleanly distinguish between processes and threads (for historical reasons)

Process Creation

Principal events that cause process creation
1. System initialization
   - Foreground processes (interactive programs)
   - Background processes
     - Email server, web server, print server, etc.
     - Called a daemon (Unix) or service (Windows)
2. Execution of a process creation system call by a running process
   - New login shell for an incoming telnet connection
3. User request to create a new process
4. Initiation of a batch job

Note: Technically, all these cases use the same system mechanism to create new processes.

Process Termination

Conditions which terminate processes
1. Normal exit (voluntary)
2. Error exit (voluntary)
3. Fatal error (involuntary)
4. Killed by another process (involuntary)

Process/Thread States

- Possible process/thread states
  - running
  - blocked
  - ready
- Transitions between states shown
Some Transition Causing Events

Running → Ready
- Voluntary Yield()
- End of timeslice

Running → Blocked
- Waiting for input
  - File, network,
- Waiting for a timer (alarm signal)
- Waiting for a resource to become available

Dispatcher

- Sometimes also called the scheduler
  - The literature is also a little inconsistent on this point
- Has to choose a Ready process to run
  - How?
  - It is inefficient to search through all processes

The Ready Queue

(b) Queuing diagram

What about blocked processes?

- When an unblocking event occurs, we also wish to avoid scanning all processes to select one to make Ready

Using Two Queues

(a) Single blocked queue

(b) Multiple blocked queues