COMP 3430

Operating Systems

June 10^{th}, 2019

Goals

By the end of today's lecture, you should be able to:

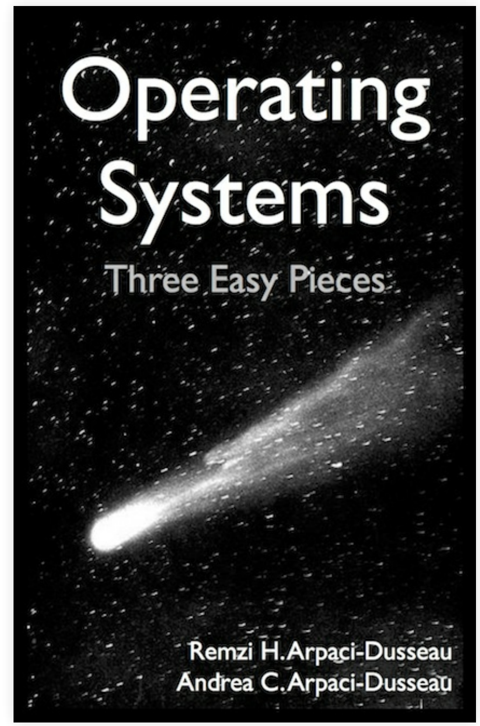
- Describe the producer/consumer model.
- Write a program that implements the producer/consumer model. (poorly)
- Describe *locks* and semaphores.
- Identify code or problems that should use locks or semaphores.

Pixabay License

OSTEP Q ’n A

Chapters for this week:

- Chapter 28
- Chapter 29
- Chapter 31
- Chapter 32
- Chapter 33 (optional)



Producers and consumers

- OS manages access to *resources*.
  - **Think about it** ⌛: What *are* the resources that the OS manages?
- Does the OS *produce* information in resources? Does the OS *consume* the information in resources?
  - Who (or what) *does* produce and consume resources?

Pixabay License

… *we* do?

- Resources may include: Persistent storage (hard drives), communication devices (network), input devices (keyboard, mouse)
  - Humans, hardware, or software may produce information.
  - Humans, hardware, or software may consume information.
- Producer/consumer model: **producers** create information, **consumers** use information.
  - Model may have one or more producers
  - Model may have one or more consumers

Let’s visualize the model

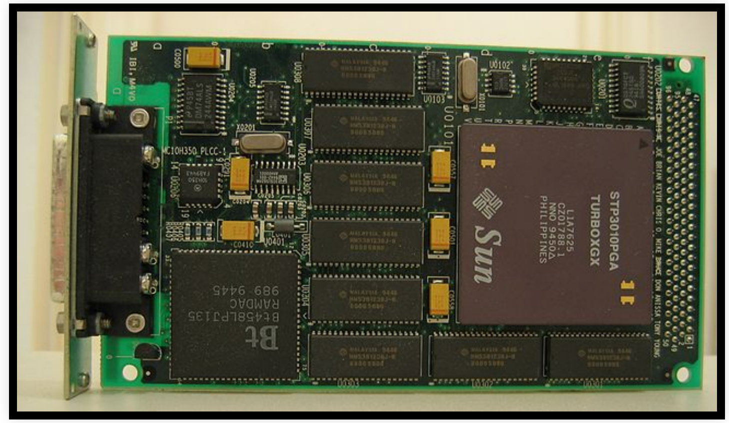
Let’s imagine what this looks like and draw some pictures.

- **Think about it** ⌛ What problems might we have **implementing** this model in software?

Pixabay License

Buffers

- Problem: Resources are *finite*.
  - Your drive has *exactly* n GB of “space” on it.
- We’ll refer to the resource and its “space” as a *buffer*.
  - The buffer is *bounded* – the buffer is not *infinite*
- Producer/consumer problem === bounded-buffer problem. (Note: also described in Chapter 30)



A *frame*buffer (© Caroline Ford CC BY-SA 3.0)

“Implementation”

Let’s take a look at some pseudocode that implements the producer/consumer model.

With some assumptions:

1. We have an infinitely long buffer.
2. We have exactly one producer and one consumer (to start).
3. Producer and consumer are executing concurrently (threads or processes).
4. The buffer and pointers are shared among producer and consumer.

“Bird” (Pixabay License)

Pseudocode

Producer:

```
void produce()
    v =
  makeNewData();
    buffer[in] =
  v;
    in++;
```

Consumer:

```
int consume()
    while (in <= out)
  {};
    x =
  buffer[out++];
    return x;
```

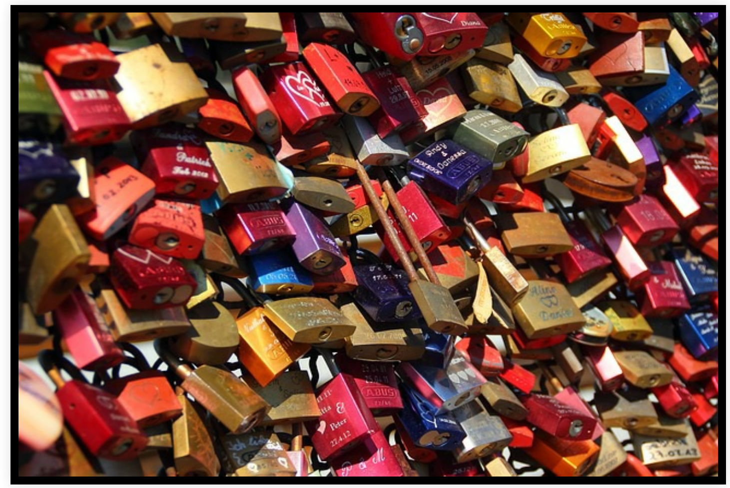
Step through this code *concurrently* ⇊.

- **Think about it** ⌛, what kind of problems might come up:
  - … when producer and consumer run at the same time, with pre-emption, on one CPU?
  - … when there are **multiple** producers or **multiple** consumers?

. . . what do?

- Let’s all agree: we need to enforce **mut**ual **ex**clusion with multiple consumers and producers.
- We need *some kind* of locking mechanism

.



Pixabay License

What do we lock?

Here's our pseudocode again:

Producer:

```
void produce()
    v =
  makeNewData(); ← lo
    buffer[in] =
  v;
    in++;
              ← un
```

Consumer:

```
int consume()
    while (in <= out)
  {}; ← L
    x =
  buffer[out++];
    return x; ← u
```

Let's *assume* we have functions `lock(&v)` and `unlock(&v)`, each take an *instance* of a lock.

Think about it ⌛:

1. Does this code already *have* a lock? yes, it just lock single pro/con.
2. What *else* would we lock in this code? Among who?
3. What problems are we going to encounter if we lock *anything*?

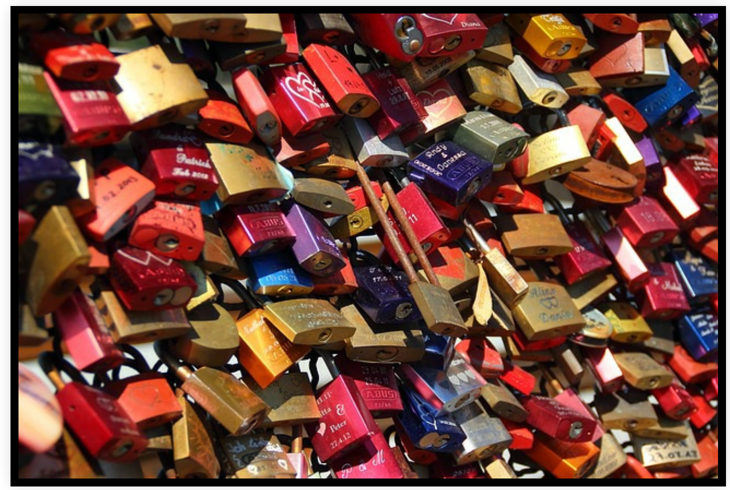
Semaphores

- We need a more *general* abstraction than a lock.
- Main limitation of locking: a lock **binary**.
  - We have possibly producers/consumers.
- A **counting semaphore** is a *generalization* of a lock.

Public Domain

Discussion: Locks and scheduling

When we add locks to concurrent code, the scheduler seems to become our enemy. **Why**?



Pixabay License

Goals

You should be able to:

- Describe the producer/consumer model.
- Write a program that implements the producer/consumer model. (poorly)
- Describe *locks* and semaphores.
- Identify code or problems that should use locks or semaphores.

Pixabay License

greencrook:

> greencrook:
>
> > My uni students asked me if they had homework for the holidays and I felt so bad for them and their tired, dead eyes that I told them to just mail me pics of their favorite pokemons.
>
> Three students sent me digimons I can't fucking trust them with anything I give up