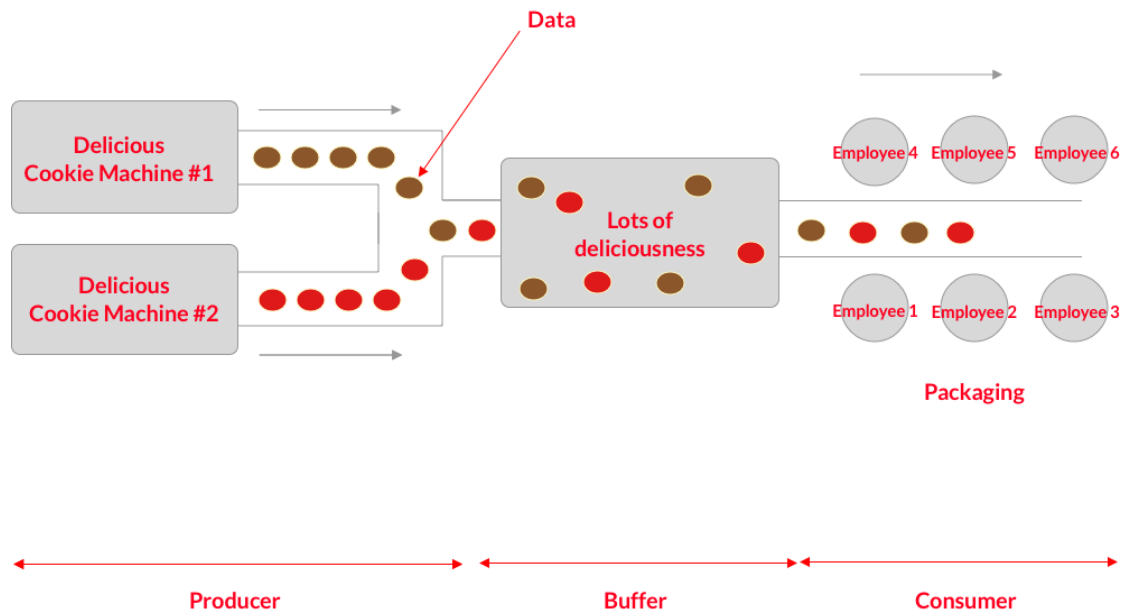# CSC369 Week 3 Notes

Hyungmo Gu
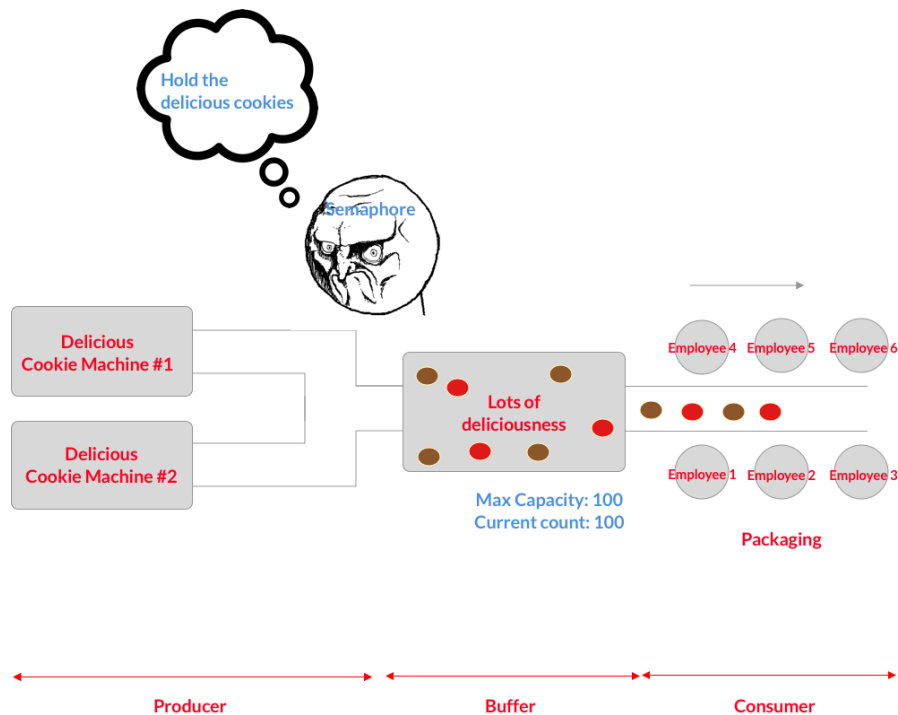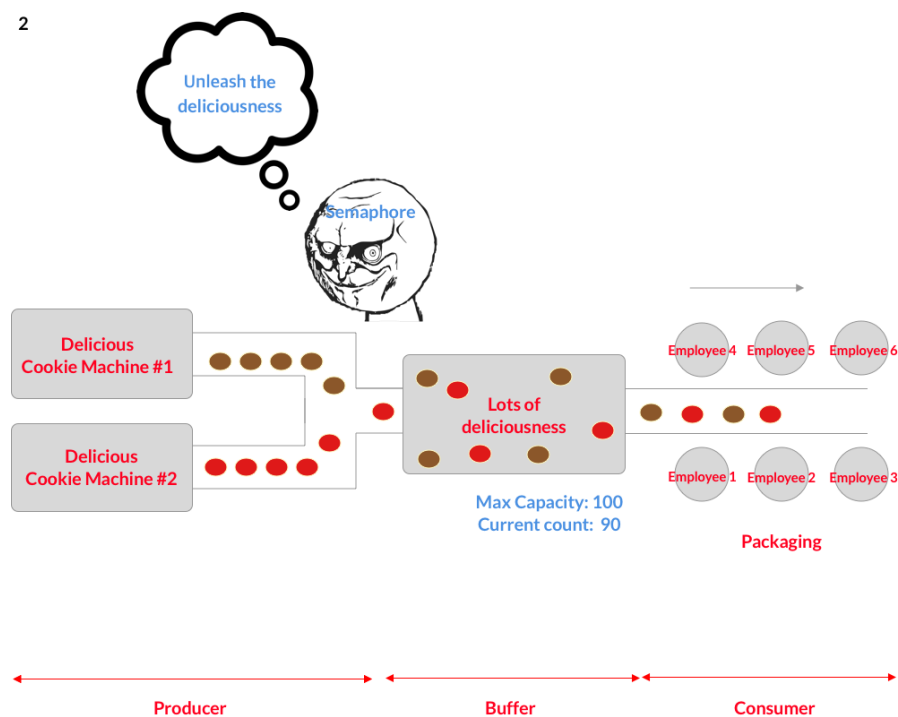
May 23, 2020

1 Synchronization

- Producer and Consumer Problem

  - Is also known as **bound-and-buffer** problem
  - Achieves synchronization
  - Has two types of processes
    1. **Producer**
       * Produces data
       * Puts data into buffer
    2. **Consumer**
       * Consumes data
       * Removes data from buffer, one piece at a time
  - It's like kimchi factory, or delicious cookie factory :)

- Semaphore

  - Developed by Dijkstra in 1962.

  - is a signal

    * Uses a non-negative integer variable that is shared between threads [*Note: Need to come back later*]

    * Has two "**atomic**" operations

      1. **Wait** (Also called P, or decrement)
      2. **Signal** (Also called V, or increment)

  - Is easy to understand

  - Is difficult to use

    * One tiny mistake and everything comes to a halt

- Types of Semaphores

  1. Counting Semaphore

     - $count = N \Rightarrow$ Max number of resources
     - $count \uparrow$ when resource added
     - $count \downarrow$ when resource used
     - $count = 0 \Rightarrow$ No resources available $\Rightarrow$ **Wait** until $count > 0$

2

Unleash the
deliciousness

Semaphore

**Delicious
Cookie Machine #1**

**Delicious
Cookie Machine #2**

**Lots of
deliciousness**

**Max Capacity: 100
Current count: 90**

Employee 4   Employee 5   Employee 6

Employee 1   Employee 2   Employee 3

Packaging

Producer   **Buffer**   **Consumer**

2. Binary Semaphore

    − *count = 1* ⇒ Unlocked / Available

       ∗ A thread can go in

    − *count = 0* ⇒ Locked / Unavailable ⇒ **Wait** until *count >0*

       ∗ Other threads must wait

    − It's like the security at airport, or the portable bathroom from week 1 notes
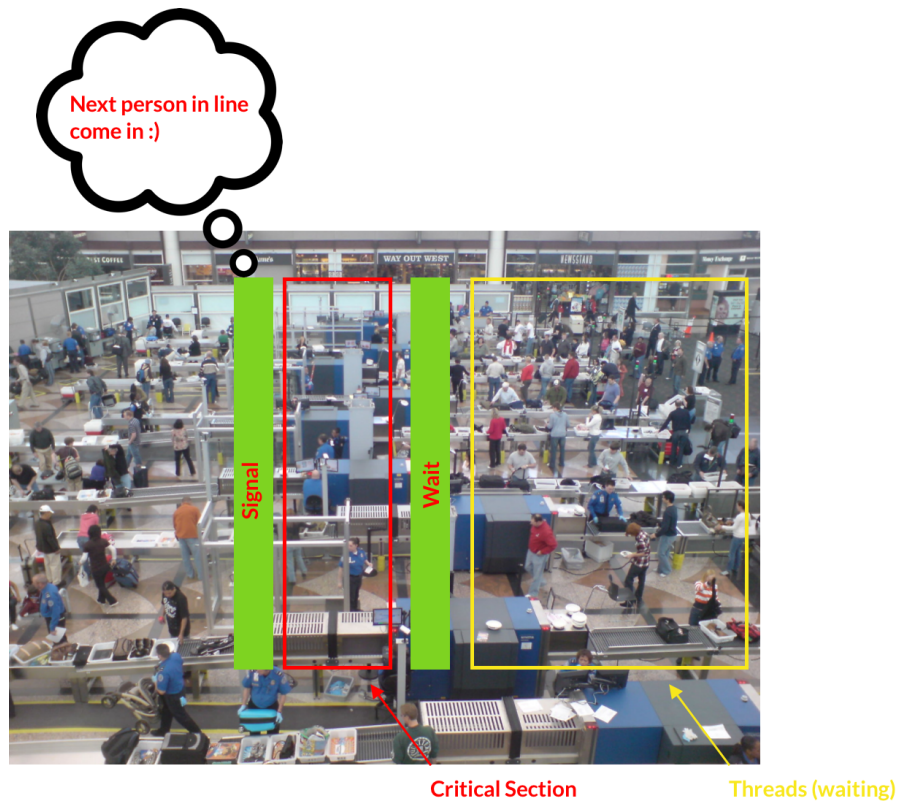
- Using Binary Semaphores

- Atomicity of wait() and signal()

- Read-write problem

- Reader's operation

- Writer's operation

- Reader's and Writer's Operation

- Notes on Readers/Writers

- Monitors

    - Is solution to semaphore
        * Is easier to implement
        * Creates less bugs
    - Still not a good solution
        * Usable only in few programming languages
            · C not supported
            · Java supported
        * Is designed for single or multiple CPUS with access to a common memory

* Not designed for the age of internet
  · Fails with distributed system with each having private memory connected via internet
  · Can't exchange information between machines

2   Intro to Scheduling

- State Queues

- Proccessor Scheduling

  –

- What Happens on Dispatch / Context Switch

- Process Life Cycle

- What is Proceess Scheduling

- When to Schedule

- Types of Scheduling

  –