# Distributed Systems Recitation

Lamont Nelson

New York University

*lamont.nelson@nyu.edu*

September 11, 2015

# Overview

# Concurrency in Go

- Channels are a FIFO queue abstraction that allow inter-thread communication
- Two types: buffered and unbuffered
- Buffered channels block the sender if the buffer is full and blocks the receiver if the buffer is empty.
- Unbuffered channels block the receiver/sender if there is no data to be sent/received.
- Multiple channels can be polled for data using the switch construct.
- Channels can be used to replace tradtional concurreny control abstractions such as Mutexes and Semaphores.
- Based on the work by C. A. R. Hoare. Communicating Sequential Processes. 1978.

# Concurrency in Go (cont)

- The language also offers other concurrency control mechanisms in the "sync" and "sync/atomic" packages.
- Examples: Mutex, Barriers (WaitGroup), Condition Variables, Atomic Load/Stores

# Producer/Consumer Problem

- Also called the bounded buffer problem
- Setup:
  - Fixed size buffer
  - $N$ Producer threads - put items onto the buffer
  - $M$ consumer threads - retrieve items from the buffer
- Problem: How can we allow the producers to concurrently distribute items to the consumers without overflowing the buffer?
- Multiple ways to accomplish this in Go

# Lab 1 Questions

- How can we determine the current list of workers?
- What are the paramaters required to make an RPC call?
- What should happen if a worker fails? How do we know a worker has failed?
- Other Questions?

# The End