Plan
* Comments on DPPRs
* Recitation Qs
* Setting for GFS
* How GFS works

Logistics
* Midterm graded next week
* Technical feedback on DPRR next week
* Volunteers for recitation next week
Comments on Technical Writing

1. Passive voice
   "message is sent" ← P.V.
   "server sends message"

2. Specifics
   "GFS provides large amounts of storage to many machines."

3. Claims must evidence
   beautifully built machine
   single dish machine

4. Compound adjectives
   single-disk machine
Recitation Qs

1. What assumption does GFS rely on?
   - Cheap parts, fail often
   - b/w > latency
   - Large streaming reads, small random reads
   - "Large" files, many GBs

2. How does GFS use these assumptions?
   - Main + chunks — replication
   - Large chunk size — streaming read
   - Concurrent appends — write straight to chunk server

3. Why does GFS make these assumptions?
   - Google FS, Google workload
Let's design our own storage system.

**Attempt 1**
- Single server
  - + Simple
  - - Faults
  - - Capacity

**Attempt 2**: Replica
- + Fault tolerance
- - Inconsistency
Attempt 3: 

- Main Sail
- Bottlenecks

GFS is attempt 3

+ No client side problem
- Main Sail
- Bottlenecks

write $A=3$

$A=5$

$A=3$

Replicas
Key GFS properties

* Single main server
  - Replicated w/ logs
* Throughput
  - Caching
    - Data for reads/writes bypasses main server
* Weak consistency guarantee
  - Two replicas in GFS may not always have identical copies of a chunk.
Plan

- Comments on DPPRs
- Recitation Qs
- Setting for GFS
- How GFS works

Logistics

- Midterm graded next wk
- Technical feedback on DPRR next week
- Volunteers for recitation next week
Comments on Technical Writing

1. Passive voice
   - "message is sent"  vs. "server sends msg"

2. Specifics

3. Claims must have evidence.

4. Compound adjectives
   - single disk drive
   - single-disk drive
Recitation Qs

1. What assumptions does GFS rely on?
   - Mostly append
   - Large files
   - Failures common, cheap parts
   - B/W > latency

2. How does GFS exploit?
   - Chunk size big ⇒ less metadata
   - Cheaper hardware

3. Why does GFS make these assumptions?
   - $, co-design
To understand GFS, let's design our own.

**Attempt 1:** Single server

- Simple
- Capacity
- Fault tolerant
- Throughput
Attempt 2: More servers, replication

+ Fault tolerance
- Inconsistency
Attempt 3: Use a main server

+ Fault tolerance
+ Consistency?
- Bottleneck at main
- Single part of Salun

GFS as Attempt 3 with
Sixes:
Key GFS properties

- Single main
- Main server only serves metadata
  - Data (≥MB chunks) bypasses main
  - QoS tight

- Weak consistency guarantee
  - Two replicas may store non-identical copies of a chunk!
GFS: Record append (append block & data ≥ 1 km)

Chunk server

Main

append < ...>

primary <append>

OK

$.txt?