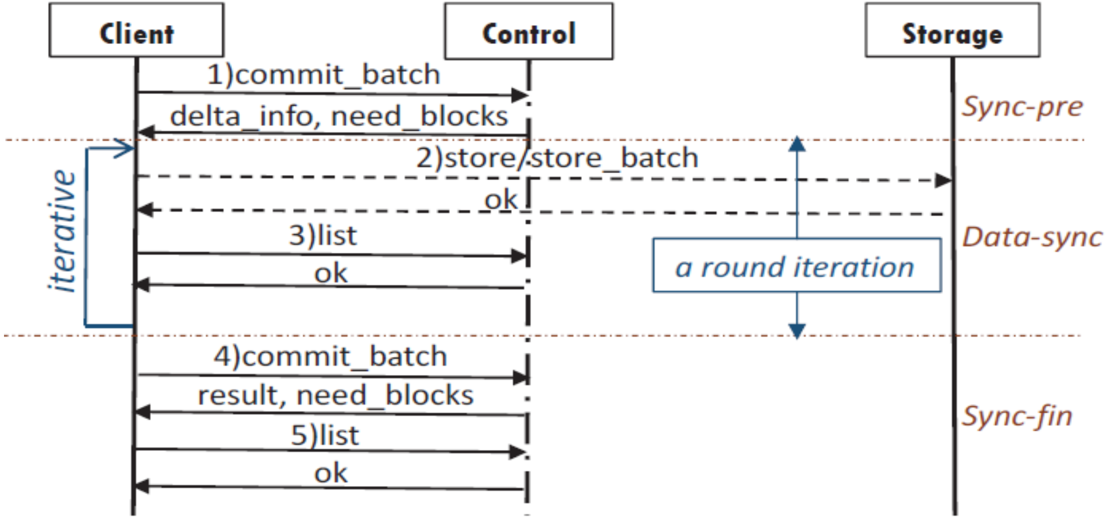
Improving Sync Efficiency for Mobile Cloud Storage Services

Zeqi Lai

Tsinghua University

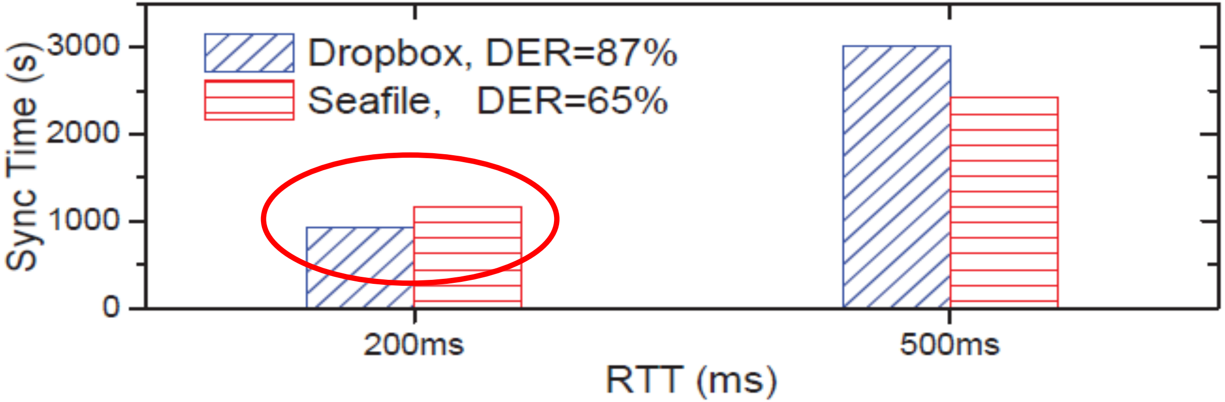
Measurement Methodology

- **Pinning down the sync protocol**
  - Methodology: network trace analysis & decryption
  - In-depth analysis: hijack SSL socket of Dropbox
  - Three sync/upload stages: *sync preparation*, *data sync*, *sync finish*



Identifying the sync inefficiency problem

- **Redundancy deduplication**
  - Seafile eliminates more redundancy than Dropbox on a same data set
  - Seafile uses smaller content-define chunk size than Dropbox
  - But Seafile may need more sync time even with reduced traffic!
  - Finding more redundancy needs more CPU time

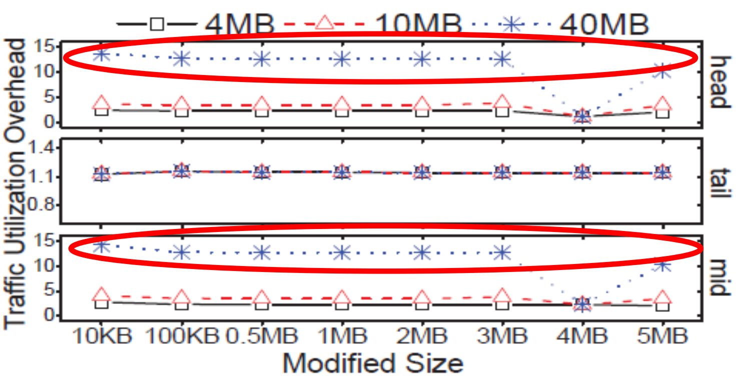


DER:
deduplicated file size/the original file size

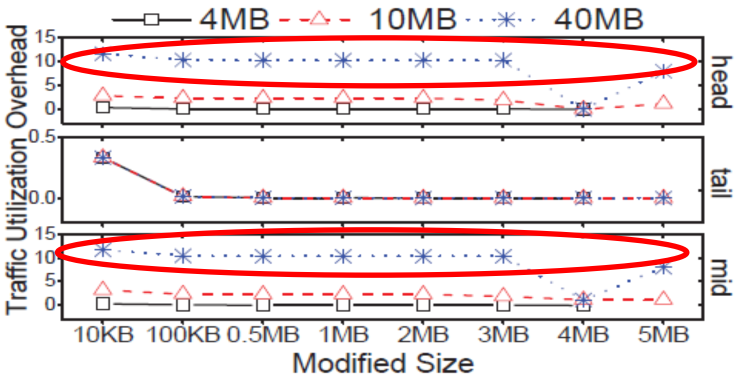
Identifying the sync inefficiency problem

- **Dropbox fails on incremental sync with delta encoding**
  - 3 operations (*flip bits, insert, delete*) over continuous bytes of a synced test file
  - Insert 2MB at head of a 40MB file, but 20MB transmitted

TUO: generated traffic size /expected traffic size



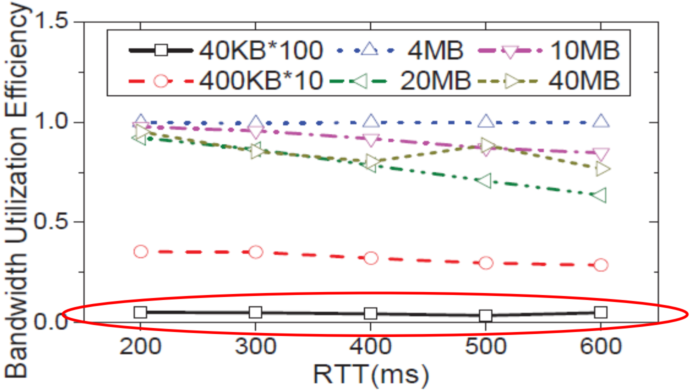
(b) insert



(c) delete

Identifying the sync inefficiency problem

- **Bandwidth inefficiency**
  - Synchronize files differing in size
  - Sync is not efficient for large # of small files in high RTT conditions
  - BUE: measured throughput / theoretical TCP bandwidth



(b) GoogleDrive

- **Root cause analysis**
  - Client waits for ack from server before transmit next chunk
  - Sequential ack for each small chunk and even too many new connections bear the TCP slow start, especially with high RTT
  - Bundling is quite important

Thank you!

Questions?