Cloud Computing

Storage Systems

Original: Eva Kalyvianaki
ek264@cam.ac.uk

Sadržaj

- **The Google File System** SOSP 2003
  - Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung
  - https://static.googleusercontent.com/media/research.google.com/en//archive/gfs-sosp2003.pdf

- **Bigtable: A Distributed Storage System for Structured Data** OSDI 2006
  - Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach Mike Burrows, Tushar Chandra, Andrew Fikes, Robert E. Gruber
  - https://storage.googleapis.com/pub-tools-public-publication-data/pdf/68a74a85e1662fe02ff3967497f31fda7f32225c.pdf

Zahtevi cloud aplikacija

- Većina aplikacija u oblaku su intenzivne po pitanju obrade podataka (data-intensive) i testiraju ograničenja postojeće infrastrukture. Zahtevi:
  - Brz razvoj aplikacija i kratko vreme isporuke na tržište
  - Mala kašnjenja
  - Skalabilnost
  - Visoka dostupnost
  - Dosledan prikaz podataka
- Postojeći modeli baze podataka ne mogu istovremeno da zadovolje ove zahteve; Npr. relacione baze podataka su jednostavne za razvoj aplikacije, ali se ne podešaju dobro

Google File System (GFS) - motivacija

- GFS ➔ razvijen krajem devedesetih; koristi hiljade sistema za skladištenje izgrađenih od komercijalno dostupnih komponenti za obezbeđivanje velikih skladišta podataka (PB) velikoj korisničkoj zajednici sa raznovrsnim potrebama
- Motivacija
  - Otkazivanje komponenti je očekivano
  - Apl./OS greške, ljudske greške, otkazivanja diskova, napajanje, ...
  - Fajlovi su ogromni (muti-GB do -TB fajlovi)
  - Najčešća operacija je da se u postojeću datoteku dodaje sadržaj (append); nasumične operacije pisanja u datoteku su izuzetno retke. Sekvencijalne operacije čitanja su standardni obrazav pristupa.
  - Model konzistentnosti treba da bude opušten kako bi se pojednostavila implementacija sistema, ali bez stavljanja dodatnog opterećenja na programere aplikacija

GFS Pretpostavke

- Sistem je izgrađen od jeftinih komercijalnih komponenti koje često otkazuju.
- Sistem skladišti umeren broj, ali velikih datoteka.
- Poslovi obrade (workload) se uglavnom sastoje od dve vrste čitanja: velikih *streming* učitavanja i malih nasumičnih čitanja.
- Tipično radno opterećenje – obrada (workload) takođe ima mnogo velikih sekvencijalnih upisa kojima se podaci dodaju datotekama.
- Sistem mora da primeni dobro definisanu semantiku za kontrolu itovremenog dodavanja u istu datoteku od strane više klijenata.
- Veliki propusni opseg je važniji od malog kašnjenja

GFS API

- On pruža poznati interfejs, sličan ali ne i isti kao POSIX.
- Podržava: kreiranje, brisanje, otvaranje, zatvaranje, čitanje i pisanje
- Plus: *snimak (snapshot)* i *dodavanje zapisa (record append)*
- Snimak
  - kreira kopiju datoteke ili stablo direktorijuma po niskoj ceni
- Dodavanje zapisa
  - omogućava da više klijenata uporedo u istu datoteku dodaju podatke, istovremeno garantujući atomičnost operacija

Arhitektura GFS klastera

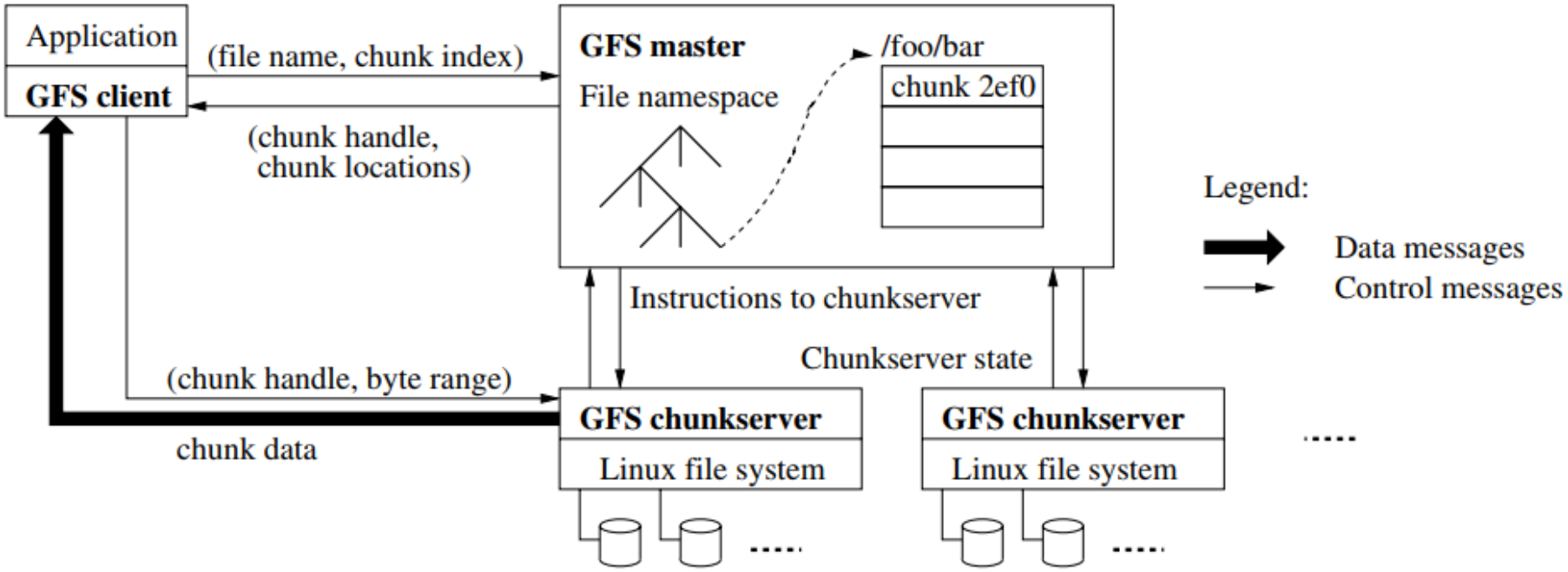


Figure 1: GFS Architecture

Architektura GFS Cluster

- Jedan glavni (master), više servera za delove (chunkservers) i klijente, sve na Linux mašinama.
- Blokovi (chunks) fiksne veličine, pristup blokovima preko 64-bitnih jedinstvenih i nepromenljivih *chunk handler-a* (identifikatora).
- Blokovi se čuvaju na lokalnim diskovima na chunkserverima, tri replike.
- Master održava sve metapodatke sistema datoteka: kontrolu pristupa, mapiranje iz datoteka u blokove, lokacije blokova itd.
- GFS kôd klijenta primenjuje FS API i komunicira sa masterom i chunkserverima za čitanje/pisanje podataka za aplikacije.
- Nema keširanja na strani klijenta ili *chunkservera*.

- Jedan master??? Da li je to dobra ideja?
  - Jednostavan dizajn, master donosi preciznije odluke o raspoređivanju blokova na blok-servere i replikaciju koristeći globalno znanje.

GFS – Design Decisions

- Izdeliti datoteke na velike blokove
- Impelmenirati atomičnu operaciju dodavanja na fajl – što omogućava konkurentno dodavanje na isti fajl
- Klaster izgraditi koristeći mrežu velike propusne moći, bitnije nego da bude sa malim kašnjenjem. Odvojiti tok podataka od tokova komandi. Iskoristiti topologiju mreže kako bi se podaci raspoređivali na najbliže čvorove.
- Eliminisati keširanje na klijenstkoj strain jer ono usložnjava održavanje konzistentnosti između različitih kopija.
- Konzistentnost se postiže tako što se kritične operacije usmeravaju kroz master.
- Minimizuje se učešće mastera kada se samo pristupa podacima (avoid hot-spot contention)
- Podrška za jednostavan system "snimanja" trenutnog stanja (checkpointing) I brze mehanizme oporavka sistema
- Podrška za efikasan *garbage collection*

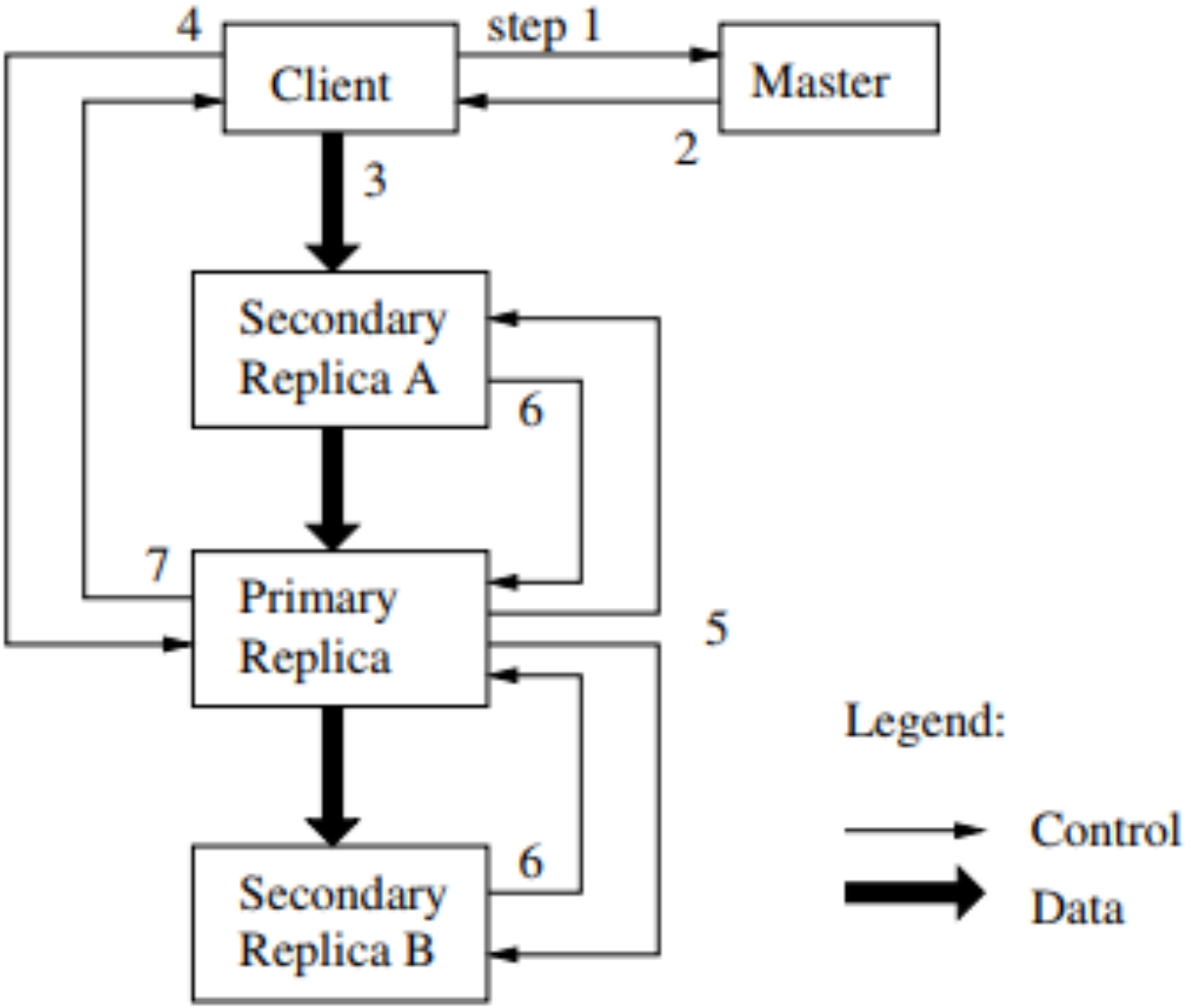
GFS Blokovi

- GFS datoteke su kolekcija blokova fiksne veličine - chunks
- Veličina bloka je 64 MB
- Velika veličina bloka povećava verovatnoću da će se više operacija usmeriti ka istom čvoru I time se smanjuje broj dodatnih zahteva za utvrđivanje lokacije bloka u sistemu
- Veliki blokovi smanjuju tabee sa metapodacima na master čvoru
- Svaki chunk se dalje deli na podblokove veličine 64KB
- Problem sa malim fajlovim sa malim brojem blokova (chunks) → hot spots → povećati factor replikacije

Model konzistentnosti

- Mutacije se izvršavaju kao upis ili češće dodavanje zapisa
- Svaka mutacija se izvršava na svakoj replici datog bloka
- Korišćenje "najma" za održavanje korektnog redosleda mutacija:
  - Master dodeljuje *lease* jednoj od replica (primarnoj – *primary*)
  - Ova replica određuje serijski redosled mutacija na bloku
  - Sve ostale rteplike poštuju ovaj utvrđeni redosled mutacija
  - Globalni redosled mutacija je određen na osnovu:
  1. Na osnovu broja "najma" koju izabrana replica dobija od strane mastera i
  2. unutar jednog *leasa* na osnovu serijskog broja koji dodeljuje primara replica Leases are initially 60 secs
- Ukoliko master utvrdi gubitak primare replike – pravi novi *lease*

Kontrola upisa i tok podataka



Atimično dodavanje zapisa - Atomic Record Appends

- Klijent specificira samo podatke
- GFS dodaje podatke na fajl na udaljenosti (offset) koju on određuje I vraća taj podatak klijentu
- Primarna replica proverava da li bi dodavanje tih podataka dovelo do toga da blok pređe maksimalnu veličinu

Opearacije Master čvora

Prostor imena I zaključavanja

- Svaka master operacija prvo obezbeđuje skup zaključavanja
- Dozvoljava konkurentne mutacije u istom direktorijumu
- Zaključavanja se dobijaju u konzistentnom poretku kako bi se izbeglo međusobno zaključavanje

Upravljanje replikama

- Replike blokova se nalaze na različitm rekovima
- Saobraćaj ka pojedinim blokovim troši ukupan propusni opseg mreže ka pojedinim rekovima.
- Novi blokovi se smeštaju na serverima koji imaju nizak nivo iskorištenosti diskova i na više rekova
- Radi se rerepliciranje ukoliko broj kopija padne ispod zadtaog broja
- Master povremeno rebalansira raspoređivanje replika kako bi bolje iskoristio resurse

GFS Zaključak

- Otkazi komponenti očekivani
- Sistem optimizovan za velike datoteke na koje se često samo dodaju novi podaci
- Otpornost na otkaze se postiže konstantim monitoringom, replikacijom podataka I automatskim oporavkom sistema
- Velika propusnost jer se odvaja tok podataka od kontrolnih tokova

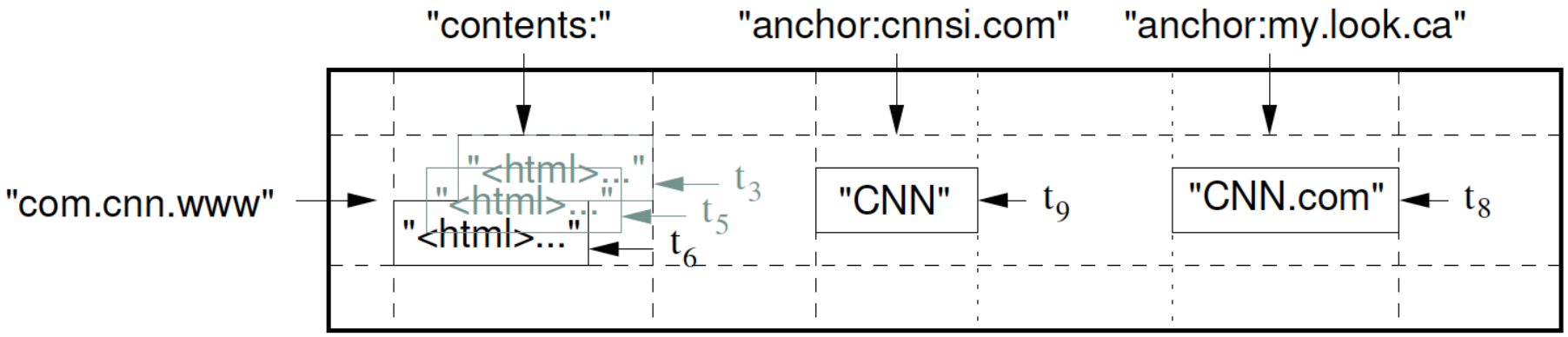
Bigtable

- Bigtable: **distribuirano** skladište za **strukturirane podatke** koje je dizajnirano da se skalira na velike skupove podataka (petabajti podataka na hiljadama mašina)
- Koriste je mnogi Google proizvodi:
  - Google Earth, Google Analytics, web indexing, …
- Može da opsluži razne tipove obrada:
  - Batch obrade koje treba da imaju veliku propusnu moć
  - Aplikacije koje u osetljive na kašnjenja
- Klijent može da utiče na lokalnost podataka I na to da li se podaci poslužuju iz memorije ili sa diska

Model podataka

- "A Bigtable is a sparse, distributed, persistent multi-dimensional sorted map."

  (row:string, column:string, time:int64) → string



Tableti

- Podaci se održavaju u leksikografskom poretku po ključu.
- Opseg redova u tabeli se može dinamički particionisati.
- Svaki pojedinačni opseg je jedan **tablet**. Ovo je ujedno I jedinica distribucije podataka.

- Bliski redovi se opslužuju na istom serveru
- Dobra lokalnost podataka se postiže dobrim izborom ključa

Gradivni elementi

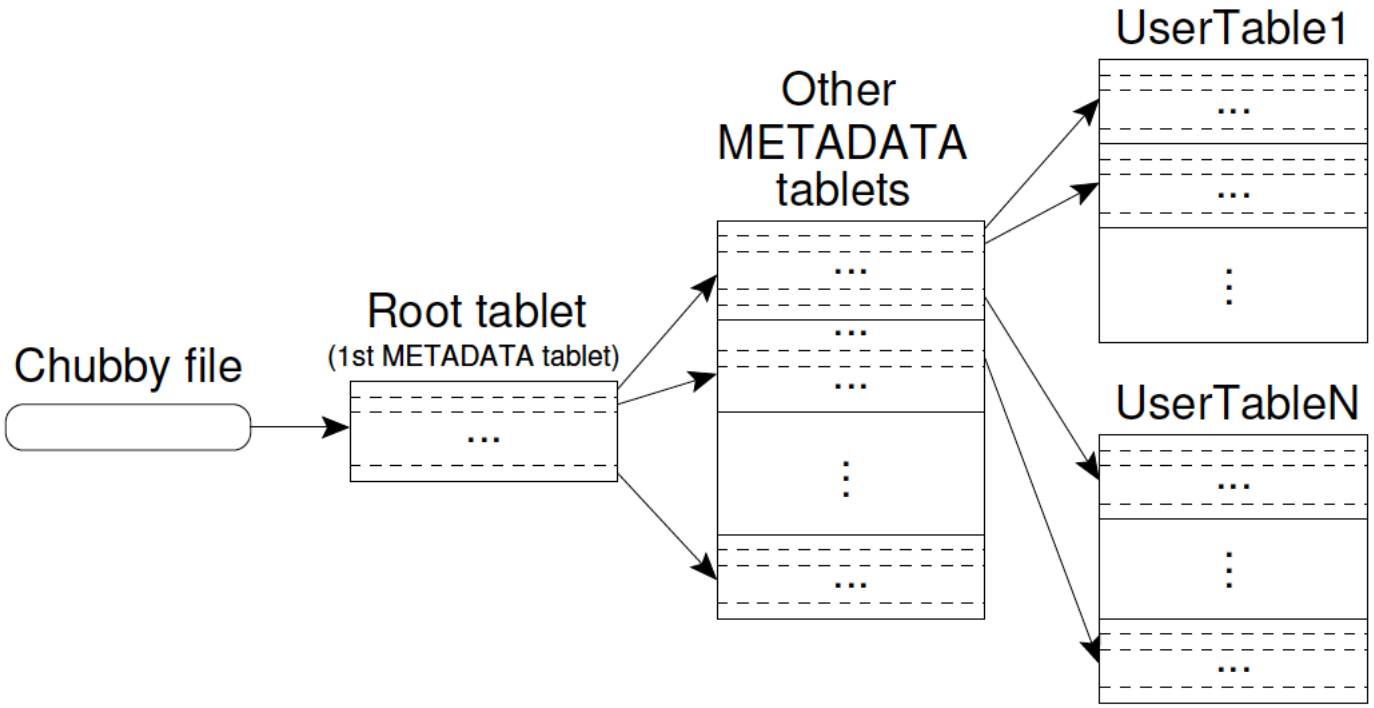
- GFS za skladišenje logova i podataka
- Bigtable klasteri se izvršavaju na mašinama iz deljenog bazena resursa (co-location).
- Za operacije raspoređivanja zadataka se oslanja na sistem za upravljanje klasterom
- Google SSTable se koristi za čuvanje Bigtable podataka
  - SSTable: perzistetna, sortirana mapa
  - sekvence 64KB blokova
  - Indeks bloka za njegovolociranje, lookup operacije sa samo jednim disk seek-om, blok se pronalazi na osnovu indeksnih podataka koji su u memoriji
- Bigtable koristi Chubby perzistentni servis za zaključavanje:
  - Obezbeđuje da ima najviše jedan aktivni master u svakom momentu
  - Čuva informaciju o lokaciji sa koje treba učitati BigTable
  - Čuva šemu podataka
  - Chubby koristi Paxos za obezbeđivanje konzistencije

Implementacija

- Tri glavne komponente
  1. Biblioteka koja se ugradjuje u svaki klijent
  2. Jedan master čvor
  3. Više servera za tablete
- Master server: vrši raspoređivanje pojedinačnih tableta na tablet-server, dodaje i nadzire tablet servere, balansira opterećenje…
- Svaki tablet server: upravlja skupom tabela, odrađuje operacije čitanja/pisanja nad tabletima, deli prevelike tabele
- Klijent komunicira direktno sa pojedinim tablet serverima kada odrađuje operacije čitanja/pisanja na tabletima. Ne zavise od mastera za utvrđivanje lokacije tableta → lightly loaded master

- Bigtable klaster čuva više tabela → tabela se sastoji od skupa tableta → svaka tabela sadrži podatke koji su u vezi sa određenim opsegom redova
- Na početku tabela se deli na više tableta 100-200MB

Lokacija tabela



Addresses 2^{34} tablets

Raspoređivanje tabela

- Svaki tablet je u svakom momentu dodeljen jednom serveru
- Master čuva listu "živih tablet" servera, trenutnih alokacija tableta po tablet-serverima i nedodeljene tablete

- Kada se master startuje
  - Postavlja master lock na Chubby file
  - Traži žive tablet servere
  - Prikuplja listu tableta sa svakog tablet-servera, kako bi utvrdio koji tableti su raspoređeni (I gde)
  - Utvrdi ukupan skup tableta → nedodeljene tablete dodaje u listu za raspoređivanje

Pristup tabelama