Napredni algoritmi i strukture podataka

Write Ahead Log

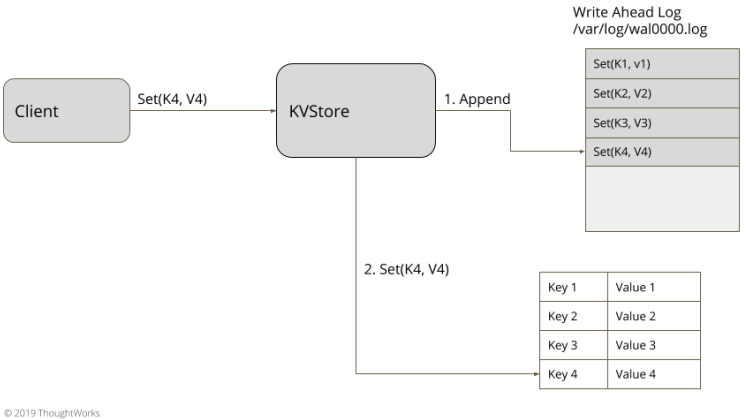
Univerzitet u Novom Sadu
Fakultet Tehničkih Nauka

Write Ahead Log

- WAL deluje kao rezervna kopija na disku za memorijsku strukturu tako što vodi evidenciju o svim operacijama koje su se desile
- U slučaju ponovnog pokretanja sistema, memoriska struktura se može u potpunosti oporaviti/rekonstruisati ponavljanjem operacija iz WAL-a
- WAL koristi isključivo sekvencijalne I/O operacije prilikom zapisa podataka na disk
- WAL kao struktura podatka, direktno se oslanja na strukturu zasnovanu na log-u

- Podatke u WAL možemo da dodamo
- *In place* izmena nije moguća
- Izmena ili brisanje nekog podatka, rezultuje novim zapisom u WAL
- WAL je *append-only* struktura
- Podaci se u WAL dodaju na kraj strukture

- Kada čitamo podatke, možemo da čitamo od početka, ili da skeniramo od nekog dela
- WAL prati vremenski tok rada sa podacima
- Noviji zapisi su na kraju WAL-a
- Stariji zapisi su na početku WAL-a
- Uvek možemo da se vratimo u vremenu i da vidimo tok operacija
- Podaci se čuvaju u memorijskoj strukturi, ali se prvo sačuvaju u WAL zbog trajnosti



(Martin Fowler Write-Ahead Log https://martinfowler.com/articles/patterns-of-distributed-systems/wal.html)

Format koji ćemo mi koristiti biće sličan RocksDB-u koji smo videli prošli put, ali malo uprošćen zbog jednostavnosti rada i naših potreba

```
+---------------+-----------------+---------------+----------------+-----------------+...+...+
|    CRC (4B)   | Timestamp (16B) | Tombstone(1B) | Key Size (8B) | Value Size (8B) | Key | Value |
+---------------+-----------------+---------------+----------------+-----------------+...+...+
CRC = 32bit hash computed over the payload using CRC
Key Size = Length of the Key data
Tombstone = If this record was deleted and has a value
Value Size = Length of the Value data
Key = Key data
Value = Value data
Timestamp = Timestamp of the operation in seconds
```

- Podaci se čuvaju u binarnom obliku
- Za čitanje, potrebno je ispravno prolaziti kroz binarni fajl
- Čitati podatke sa njihovih pozicija shodno tipu podatka koji je na toj poziciji zapisan
- Zato unapred moramo znati strukturu WAL-a, da bi ispravno čitali podatke!
- **Za vaš projekat, ovo je format koji će biti korišćen**

- CRC koristimo kao **error-detecting** mehanizam za otkrivanje promena u podacima
- CRC je *hash* funkcija koja detektuje promene nad podacima
- Kao tip kontrolnog zbira (checksum), CRC proizvodi skup podataka fiksne dužine na osnovu izvnorne datoteke ili većeg skupa podataka
- CRC se zasniva na binarnoj podeli i naziva se i *kontrolna suma polinoma koda*
- Ovo možemo koristiti kao mehanizam potvrde da li je bilo izmena/oštećenja kada se podaci pročitaju
- Ako je došlo do promene, taj podataka više nije validan

Zadaci

- Implementirati WAL algoritam kao **append-only** strukturu gde se podaci uvek dodaju na kraj datoteke
- Kao foramt zapisa koristiti format sa predavanja (i iz ove pripreme)
- Podatke upisivati u binarnom obliku prema segmentima kako su specificirani u formatu
- Omoguíti čitanje od početka datoteke pa do kraja datoteke
- Omogućiti CRC mehanizam (koristiti dati helper.go meanizam)
- Omogućiti *batch* zapis, tako što ćete napraviti nekakvu *buffer* strukturu u memoriji (koristeći strukturu po izboru), kada se napuni kapacitet (vi definišete) dodati zapise u WAL datoteku
- Omogućiti da se svaki zapis piše direktno u fajl, bez *buffer* strukture