Napredni algoritmi i strukture podataka

LSM Stabla, Kompakcije

Univerzitet u Novom Sadu
Fakultet Tehničkih Nauka

Problemi trenutne organizacije sistema

- Svaki put kada se Memtable popuni, odradimo Flush na disk praveći SSTable — vremenom će ih se nakupiti jako mnogo
- Zbog toga čitanje postaje vrlo skupo, treba da prođemo kroz veliki broj struktura u potrazi za elementom
- Pored toga, SSTable je nepromenljiva struktura, šta ako su podaci obrisani...
- Brisanje i ažuriranje zapravo predstavljaju nov zapis, samo dodajemo elemente — zauzimamo dodatni prostor zastarelim verzijama podatka
- Ako podatak više nije aktivan, trebalo bi da ga uklonimo sa diska

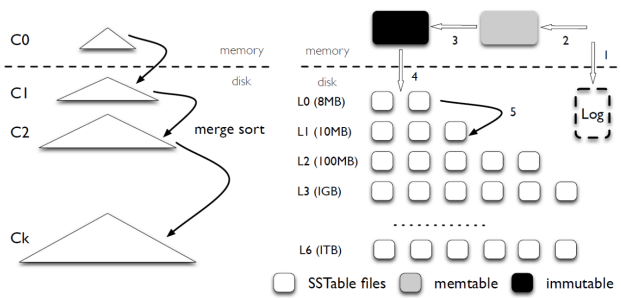
LSM stabla

- Želimo nekako da strukturiramo SSTable elemente kako bismo lakše upravljali njima
- Struktura koju koristimo zove se **Log-Structured Merge Tree** ili **LSM stablo**
- SSTable je struktura tipa log-a, SSTable grade LSM stabla, LSM stabla su strukture tipa log-a
- LSM stablo je struktura podataka dizajnirana da obezbedi jeftino indeksiranje za datoteke koje imaju visoku stopu dodavanja (i brisanja) tokom dužeg perioda
- Jedina stvar koja je potrebna da bi LSM imala prednost izbora, spram drugih struktura, je visoka stopa pisanja naspram stope čitanja

LSM Stabla — struktura

- Pošto govorimo o stablu, ono sigurno ima nekakve nivoe
- LSM stablo se sastoji od dva ili više nivoa $C_i = (C_0, ..., .C_k)$
- Na primer, dvokomponentno LSM stablo ima:
  1. Manju komponentu koja je u potpunosti rezidentna u memoriji, poznata kao C_0 komponenta
  2. Veću komponentu koja je rezidentna na disku, poznatu kao C_1 komponenta
- Podaci se prvo ubacuju u C_0, koji služi kao bafer za zapise, a odatle migriraju/zapisuju u C_1
- Maksimalan broj nivoa je konfigurabilan, možemo specificirati koliko god nam je potrebno (deo konfiguracije sistema)

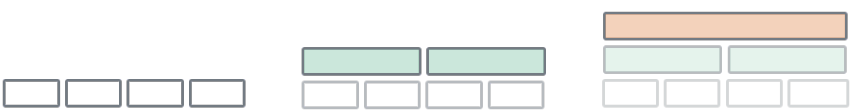
- U našem slučaju nivo C_0 je Memtable, ona se nalazi u memoriji
- Drugi nivo C_1 (i sve ostale) čine SSTables koje nastaju kada se Memtable (nivo C_0) popuni i njen sadržaj zapiše na disk



(LSM Tree Levels)

Kompakcije

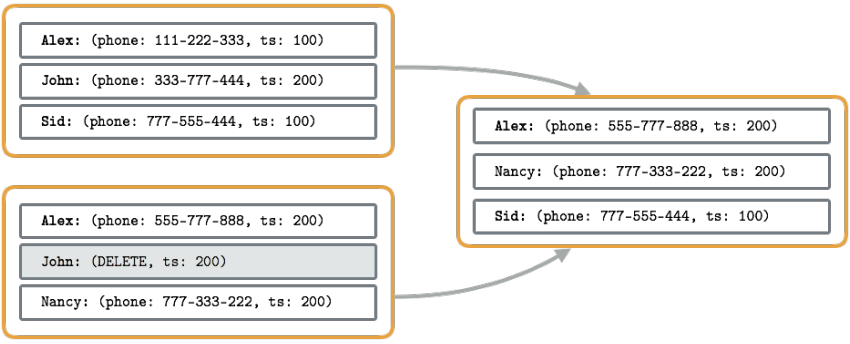
- LSM stabla su (uglavnom) sačinjena od SSTable-a
- Svaki nivo C_k LSM stabla sadrži n_k SSTable-a
- Kompakcijama spajamo više SSTable-a i brišemo nepotrebne podatke
- Ovaj proces (najčešće) kreira veću SStable, a može da kreira i novi C_k nivo u LSM stablu



(LSM Tree compaction)

Proces spajanja

- Zbog strukture SSTable-a, operacija spajanja je efikasna
- Koristimo algoritam sličan algoritmu za spajanje koji koristi **merge sort**
- Zapisi se čitaju iz nekoliko izvora uzastopno i mogu se odmah dodati u izlaznu datoteku
- Iteratori moraju pokazivati na korespodentne elemente iz svih tabela
- Na svakom koraku, najmanji element se preuzima iz odgovarajuućeg iteratora i upisuje u novu tabelu
- Tokom kompakcije, sekvencijalno čitanje i sekvencijalno pisanje pomažu u održavanju dobrih garancija performansi
- Kada napravimo novi SSTable, moramo napraviti i sve prateće elemente za njega


Alex: (phone: 111-222-333, ts: 100)
John: (phone: 333-777-444, ts: 200)
Sid: (phone: 777-555-444, ts: 100)
Alex: (phone: 555-777-888, ts: 200)
John: (DELETE, ts: 200)
Nancy: (phone: 777-333-222, ts: 200)
Alex: (phone: 555-777-888, ts: 200)
Nancy: (phone: 777-333-222, ts: 200)
Sid: (phone: 777-555-444, ts: 100)

Čitanje podataka

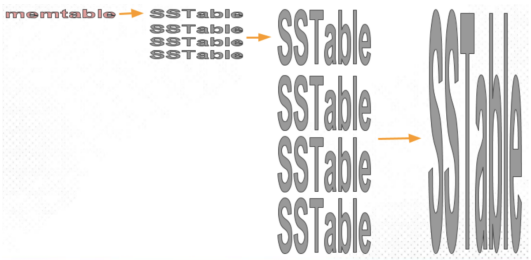
- Od dva zapisa sa istim ključem, vraća se onaj sa kasnijom vremenskom oznakom
- Ako najnoviji zapis pod nekim ključem označava obrisan element, klijentu ćemo vratiti odgovor da takav element nije prisutan u sistemu
- Kako se na nižim nivoima LSM stabla nalaze svežiji podaci, pretragu obavljamo od nižih ka višim nivoima

Algoritmi i amplifikacije

- Postoji nekoliko algoritama za kompakciju podataka
  - Size-tiered kompakcija
  - Leveled kompakcija
  - Hibridna kompakcija — kombinacija algoritama
- Ovi algoritmi imaju različite amplifikacione osobine koje se dešavaju tokom ovog procesa:
  - Amplifikacija čitanja označava broj operacija koje se dešavaju na disku pri zahtevu za čitanje
  - Amplifikacija pisanja n je definisana kroz bajtove podataka koji su stvarno upisani na disk kada je potrebno upisati jedan bajt podataka
  - Amplifikacija prostora se uglavnom odnosi na količinu nesakupljenih isteklih podataka, koji su ili stare verzije podataka ili izbrisani unosi

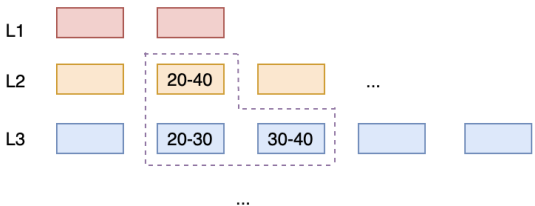
Size-tiered kompakcija

- Kada na nekom nivou C_i nakupimo n SStable-a, spajamo ih kako bismo dobili novi nivo C_{i+1}
- Spajanje na nivou C_i može da izazove spajanje na višim nivoima lančano
- Kada uradimo spajanje, obrišemo nepotrebne SSTable-e
- Prednost: nizak nivo amplifikacije pisanja, pogodno za sisteme koja zahtevaju intenzivno pisanje
- Nedostatak: amplifikacija čitanja i prostora je relativno visoka



Leveled kompakcija

- LSM-stablo se sastoji od više nivoa — naredni nivo je T puta veći od prethodnog nivoa
- Svaki nivo je **run** koji se sastoji od više SSTable-a — naredni **run** 10x prethodni
- Kada veličina svakog nivoa dostigne gornju granicu, ovaj nivo cé se spojiti sa **run**-om sledećeg nivoa
- Pošto je sve sortirano, prilikom kompakcije gledamo gde se ključ nalazi i spajamo sa tom SSTable-om
- Problem sa kojim se suočava ovaj algoritam je amplifikacija pisanja



Zadaci

- Implementirati algoritam za kompakciju po želji (size-tiered ili leveled)
- Proširiti konfiguracioni fajl, tako da sadrži maksimalan broj nivoa LSM stabla (minimalna vrednost je 4)
- Svi potrebni parametri za algoritam su ostavljeni studentima na izbor (npr. da li se nivoi ograničavaju po veličini ili broju tabela) i treba da budu konfigurabilni
- Ispravno implementirati lančane kompakcije