Arhitekture Lambda i Kappa

Principi i slučajevi korišćenja

1

Sadržaj

- Arhitektura Lambda
- Arhitektura Lambda - Sloj paketne obrade
- Arhitektura Lambda - Uslužni sloj
- Arhitektura Lambda - Sloj obrade podataka u realnom vremenu
- Arhitektura Lambda - Napredni koncepti
- Arhitektura Kappa

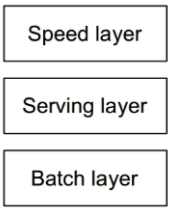
2

Arhitektura Lambda

3

Arhitektura Lambda

- Pisanje proizvoljnih upita nad velikim skupovima podataka
  - otežano je nepostojanjem standarda ili alata koji će obuhvatiti sve potrebe korisnika
  - rešenje se ogleda u korišćenju više alata i tehnika
    - koje će biti moguće iskoristiti optimalno u datom domenu problema
    - koji će biti dovoljno fleksibilni da odgovore na trenutne ali i buduće potrebe korisnika
  - **arhitektura Lambda**
    - arhitektura sistema velikih skupova podataka
    - koja organizuje obradu velikih skupova podataka kroz tri sloja
      - sloj paketne obrade podataka (engl. *batch layer*)
      - uslužni sloj (engl. *serving layer*)
      - sloj obrade podataka u realnom vremenu (engl. *speed layer*)
    - koristi javno dostupne alate
    - polazi od pretpostavke **`query = function(all data)`**
    - obuhvata paketnu obradu i obradu u realnom vremenu



izvor: *Big Data: Principles and best practices of scalable real-time data systems, Nathan Marz, James Warren*

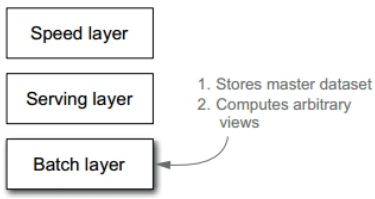
4

Arhitektura Lambda

- Arhitektura Lambda
  - zasnovana na ideji **izračunavanja unapred** pogleda potrebnih za upite
    - pogledi predstavljaju podatke obrađene tako da se mogu lako koristiti u upitima
      - pogledi zauzimaju manje memorije i imaju manje elemenata nego glavni skup podataka
    - izračunavanje izuzetno sporo usled postojanja velike količine podataka
  - za upite koji treba da uzmu u obzir **podatke koji pristižu u realnom vremenu**
    - koriste se izračunati pogledi
    - koriste se podaci koji su primljeni od trenutka poslednjeg pristiglog podatka uključenog u izračunavanje pogleda

Arhitektura Lambda

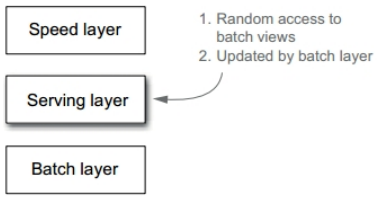
- Arhitektura Lambda - slojevi
  - sloj paketne obrade podataka (engl. *batch layer*)
    - skladišti sve podatke
      - sadrži glavni skup podataka (engl. *master dataset*)
    - paralelno se izvršavaju proizvoljne funkcije nad podacima
    - rezultat izvršenja funkcija su **pogledi nad podacima** (engl. *batch view*)
      - `batch view = function(all data)`



izvor: *Big Data: Principles and best practices of scalable real-time data systems, Nathan Marz, James Warren*

Arhitektura Lambda

- Arhitektura Lambda - slojevi
  - **uslužni sloj** (engl. *serving layer*)
    - omogućava ažuriranje i čitanje iz pogleda
    - omogućava izvršenje upita nad pogledima
      - `query = function(batch view)`



izvor: *Big Data: Principles and best practices of scalable real-time data systems, Nathan Marz, James Warren*

7

7

Arhitektura Lambda

- Arhitektura Lambda - slojevi
  - sloj paketne obrade podataka i uslužni sloj
    - poseduju skoro sve poželjne karakteristike arhitekture sistema velikih skupova podataka
    - **robusnost i tolerancija na otkaze**
      - koriste replikaciju i distribuirani sistem datoteka
      - ispravljanjem algoritma i ponovnim izračunavanjem pogleda efekti ljudske greške su svedeni na minimum
    - **mogućnost skaliranja**
      - oba sloja su u potpunosti distribuirana i lako se skaliraju dodavanjem računara
    - **generalizacija**
      - moguće je skladištiti proizvoljno strukturirane podatke i implementirati proizvoljne funkcije i algoritme obrade
    - **proširivost**
      - zasnovana na generalizaciji, dodavanje novog pogleda proširuje mogućnosti sistema

8

8

Arhitektura Lambda

- Arhitektura Lambda - slojevi
  - sloj paketne obrade podataka i uslužni sloj
    - **podrška za *ad hoc* upite**
      - sloj paketne obrade podataka podržava ove upite jer su svi podaci u jednom sistemu datoteka
    - **lako održavanje**
      - najviše truda je potrebno uložiti u podešavanje i održavanje distribuiranog sistema datoteka
      - uslužni sloj ne podržava direktan upis podataka pa ga je izuzetno lako održavati
    - **mogućnost pronalaženja grešaka**
      - uvek postoji jasan trag o izvršenju neke funkcije što olakšava pronalaženje greške
        - ne ažuriraju se direktno skladišteni podaci

9

9

Arhitektura Lambda

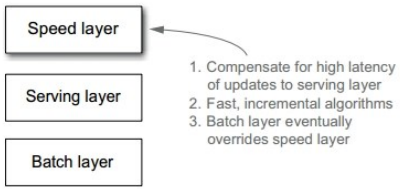
- Arhitektura Lambda - slojevi
  - sloj paketne obrade podataka i uslužni sloj
    - ne zadovoljavaju **brzo čitanje i pisanje**
      - jer je izračunavanje pogleda dugotrajan proces
      - potrebno uvesti novi sloj u arhitekturu kako bi se zadovoljila potreba za izvršenjem upita nad podacima pristiglim u realnom vremenu

10

10

Arhitektura Lambda

- Arhitektura Lambda - slojevi
  - **sloj obrade podataka u realnom vremenu** (engl. *speed layer*)
    - najkompleksniji sloj
    - skladišti tek pristigle (nove) podatke koji se ne nalaze u pogledima
      - `realtime view = function(realtime view, new data)`
    - izračunavanje novih pogleda briše podatke iz ovog sloja
      - koji su uključeni u same poglede
      - omogućava ispravljanje greške u ovom sloju ponovnim izračunavanjem pogleda i brisanjem snimljenih podataka u sloju
    - omogućava skraćenje vremena potrebnog za dobijanje rezultata upita

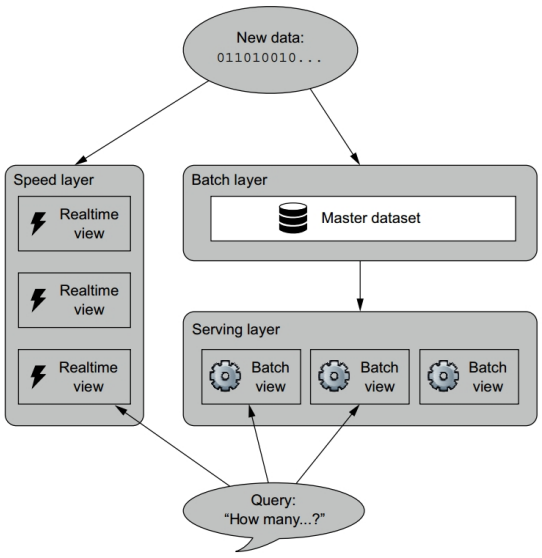


izvor: *Big Data: Principles and best practices of scalable real-time data systems, Nathan Marz, James Warren* 11

11

Arhitektura Lambda

```
batch view = function(all data)
realtime view = function(realtime view, new data)
query = function(batch view, realtime view)
```



izvor: *Big Data: Principles and best practices of scalable real-time data systems, Nathan Marz, James Warren* 12

12

Arhitektura Lambda - Sloj paketne obrade

13

Sloj paketne obrade

- Elementi sloja paketne obrade
  - glavni skup podataka
    - model podataka
    - skladištenje podataka
  - paketna obrada podataka
    - izvršavanje funkcija nad podacima u glavnom skupu

14

14

Sloj paketne obrade

- Glavni skup podataka (engl. *master dataset*)
  - jezgro bilo koje ASVSP
  - predstavlja izvor istine za ceo sistem
    - mora biti pažljivo osmišljen, upravljan i održavan
      - kako bi se povećala otpornost kako na ljudske tako i na greške tehnološke prirode
    - jedini deo arhitekture Lambda koji **ne sme da dođe u neispravno stanje**
      - podaci u ostalim slojevima se mogu izračunati iz ovog sloja
  - poseduje dve komponente
    - **model podataka**
    - **skladište podataka**

15

15

Sloj paketne obrade

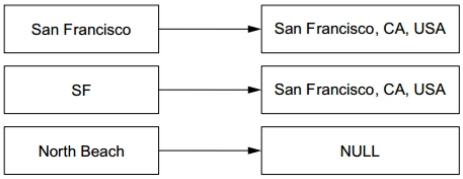
- Osobine podataka u velikim skupovima podataka
  - **podaci se čuvaju u osnovnom obliku** (engl. *rawness*)
  - **podaci su nepromenljivi** (engl. *immutability*)
  - **podaci su zauvek istiniti** (engl. *perpetuity*)

16

16

Sloj paketne obrade

- Podaci se čuvaju u osnovnom obliku
  - nad podacima se ne vrši agregacija već se čuvaju u izvornom obliku
    - po potrebi se prilagođavaju formatu skladištenja
  - kako bi bilo moguće odgovoriti na što veći broj pitanja
    - što je oblik podataka manje menjan, to je moguće odgovoriti na više pitanja
  - **skladištenje nestrukturiranih podataka**
    - kada se algoritam strukturiranja može menjati
  - skladištenje strukturiranih podataka
    - kada se algoritam strukturiranja ne menja
    - **parsiranje i semantička normalizacija**
  - više informacija ne implicira nužno kvalitetnije skladištenje podataka
    - primer: *HTML ili tekst bloga*



izvor: *Big Data: Principles and best practices of scalable real-time data systems, Nathan Marz, James Warren*

17

17

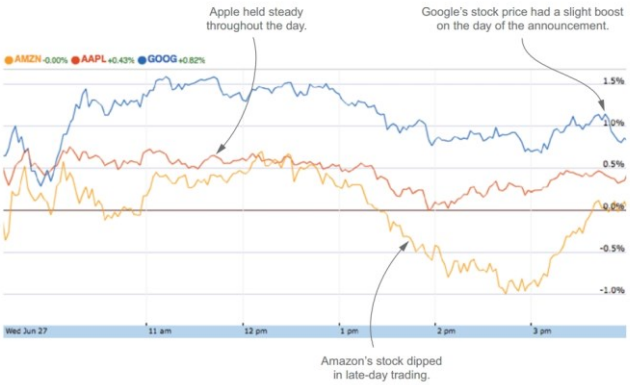
Sloj paketne obrade

- Podaci se čuvaju u osnovnom obliku - primer

Company	Symbol	Previous	Open	High	Low	Close	Net
Google	GOOG	564.68	567.70	573.99	566.02	569.30	+4.62
Apple	AAPL	572.02	575.00	576.74	571.92	574.50	+2.48
Amazon	AMZN	225.61	225.01	227.50	223.30	225.62	+0.01

Financial reporting promotes daily net change in closing prices. What conclusions would you draw about the impact of Google's announcements?

agregirani podaci



podaci u osnovnom obliku ("sirovi" podaci)

18

izvor: *Big Data: Principles and best practices of scalable real-time data systems, Nathan Marz, James Warren*

18

Sloj paketne obrade

- Podaci su nepromenljivi
  - glavni skup podataka se samo uvećava
    - dodavanjem novih torki
    - podaci se obično ne ažuriraju niti se brišu
  - doprinosi **otpornosti na ljudske greške**
    - jer podaci ne mogu biti izgubljeni (brisanjem ili izmenom)
  - doprinosi **jednostavnosti**
    - jer ne zahteva indeksiranje podataka
  - zahteva mnogo više mesta za skladištenje

19

19

Sloj paketne obrade

- Podaci su zauvek istiniti
  - podatak koji je jednom tačan mora biti uvek tačan
  - svim podacima se dodaje **vremenska odrednica**
    - koja određuje vremenski trenutak kada je taj podatak bio tačan
  - brisanje podataka je retko
    - ako je moguće, treba ga izbeći ili napraviti filtrirane kopije podataka umesto brisanja
    - brisanje nije znak neistinosti podataka već njihove niske vrednosti
    - npr. oslobađanje prostora za skladištenje, ukoliko smo ograničeni sa resursima
      - engl. *garbage collection*
    - npr. usled zakonskih regulativa
      - GDPR

20

20

Sloj paketne obrade

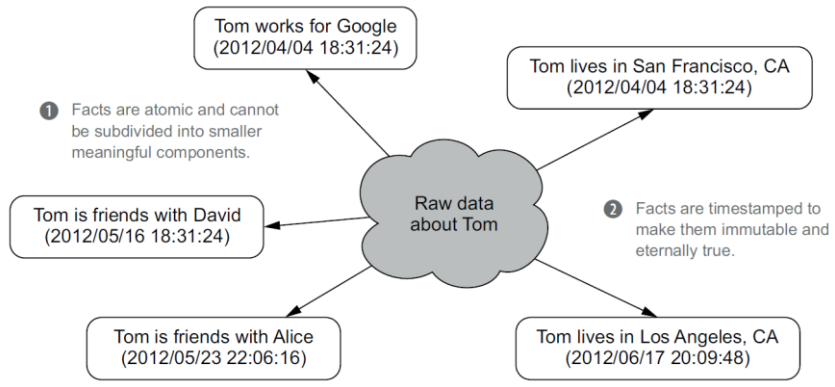
- Model podataka zasnovan na činjenicama (engl. *fact-based data model*)
  - skladištenje podataka u glavnom skupu se svodi na
    - dekonstrukciju podataka na činjenice (engl. *facts*)
      - koje se ne mogu dalje raščlaniti
      - **nedeljivost činjenica** (engl. *atomic facts*)
    - dodavanje vremenske odrednice svakoj činjenici
      - vremenska određenost činjenica (engl. *timestamped facts*)
    - dodeljivanje identifikatora svakoj činjenici
      - **mogućnost jedinstvene identifikacije činjenica** (engl. *fact identifiability*)
      - najčešće se koriste veštački ključevi
  - svi skladišteni podaci se skladište kao činjenice koje zadovoljavaju prethodno navedene tri osobine
  - glavni skup podataka je zasnovan na ovom modelu

21

21

Sloj paketne obrade

- Model podataka zasnovan na činjenicama
  - nedeljivost i vremenska određenost činjenica



22

izvor: *Big Data: Principles and best practices of scalable real-time data systems, Nathan Marz, James Warren*

22

Sloj paketne obrade

- Skladištenje glavnog skupa podataka
  - jednom kada je identifikovana šema podataka, podatke koji dolaze u sistem treba uskladištiti tako da odgovaraju definisanoj šemi
  - poželjne osobine skladišta podataka u svetlu potrebnih operacija nad podacima
    - operacija pisanja podataka
      - efikasno dodavanje novih podataka
      - mogućnost skaliranja skladišta podataka
    - operacija čitanja podataka
      - podrška za paralelno procesiranje podataka
    - obe operacije
      - mogućnost upravljanja skladištem i uticanja na njegovu cenu
      - obezbeđenje nepromenljivosti podataka

23

23

Sloj paketne obrade

- Skladištenje glavnog skupa podataka
  - distribuirani sistemi datoteka
    - datoteke predstavljaju niz bajtova sekvencijalno snimljenih na disk
      - na više računara uz dodatnu replikaciju
    - potpuna kontrola nad datotekama
      - sa stanovišta uticanja na njihovu veličinu kompresijom
      - sa stanovišta kontrole pristupa i sprečavanja nedozvoljenih operacija
    - razlikuju se u odnosu na standardne, nedistribuirane sisteme datoteka
      - postoji više ograničenja u radu sa datotekama
        - često nije moguće modifikovati datoteku nakon kreiranja
        - često nije moguće pisati u sredinu postojeće datoteke
      - postojanje mnoštva manjih datoteka može da bude pogubno po performanse sistema datoteka
    - postoji više implementacija
      - npr. *Hadoop Distributed File System* (HDFS)

24

24

Sloj paketne obrade

- Izvršavanje upita u sloju paketne obrade
  - cilj bilo kojeg sistema koji upravlja podacima je da pruži odgovore na postavljena pitanja
  - izvršavanje upita se svodi na izvršavanje funkcija nad celim skupom podataka
    - izvršavanje funkcija je paralelno
    - često se koristi paradigma MapReduce
      - za kreiranje paralelno izvršivih algoritama

25

25

Sloj paketne obrade

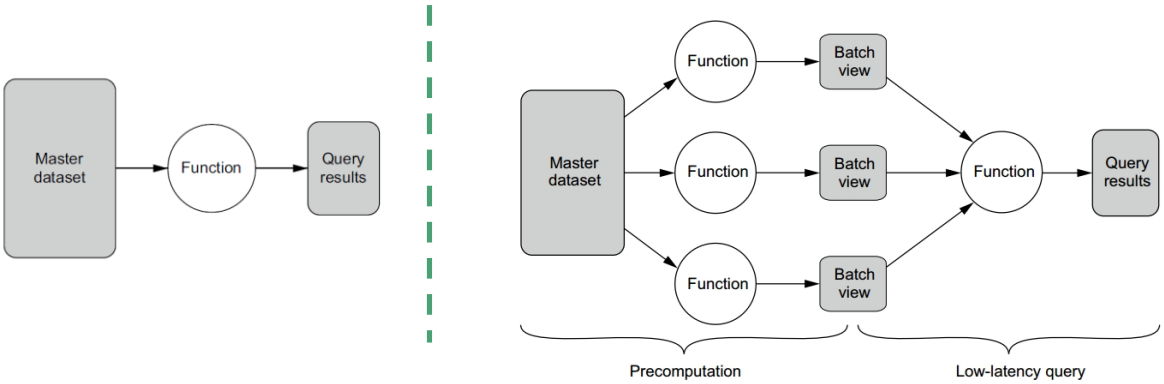
- Izvršavanje upita u sloju paketne obrade
  - za bilo koji upit moguće je napraviti pogled u sloju paketne obrade
    - koji će dozvoliti brže izvršenje upita u uslužnom sloju
    - **ne izračunavaju se pogledi za svaki mogući upit** i svaku moguću vrednost parametara upita
      - izračunavaju se pogledi koji mogu biti korišćeni za različite upite
        - ili različite vrednosti parametara
      - pogledi predstavljaju samo međukorak u izračunavanju rezultata upita

26

26

Sloj paketne obrade

- Izvršavanje upita u sloju paketne obrade



27

izvor: *Big Data: Principles and best practices of scalable real-time data systems, Nathan Marz, James Warren*

27

Sloj paketne obrade

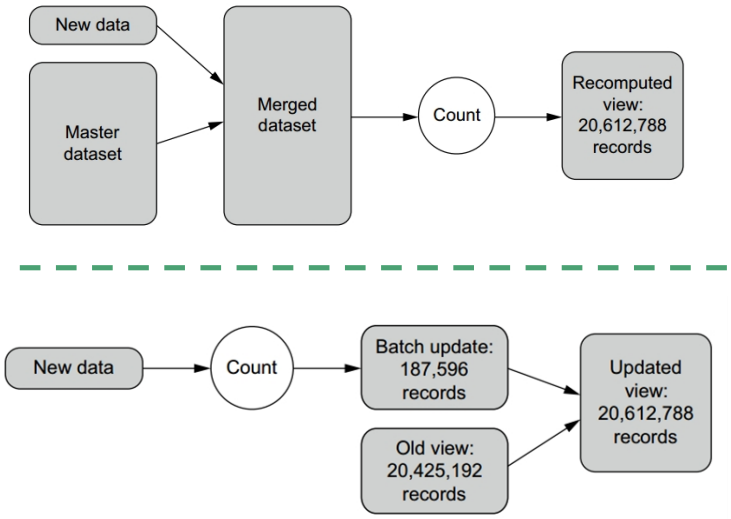
- Izvršavanje upita u sloju paketne obrade
  - ažuriranje pogleda
    - **algoritmi sa kompletnim izračunavanjem** (engl. *recomputation algorithms*)
      - ažurira se prvo glavni skup podataka
      - ponovo se izračunavaju svi pogledi
    - **inkrementalni algoritmi** (engl. *incremental algorithms*)
      - pogledi se direktno ažuriraju na osnovu novih vrednosti
    - odabir algoritma zavisi od tri parametra
      - performantnost
      - otpornost na ljudske greške
      - generičnost algoritma
    - algoritmi sa kompletnim izračunavanjem su **uvek** potrebni
      - kako bi se obezbedila otpornost na ljudske greške

28

28

Sloj paketne obrade

- Algoritmi za ažuriranje pogleda
  - algoritmi sa kompletnim izračunavanjem
    - slika gore
  - inkrementalni algoritmi
    - slika dole



29

izvor: *Big Data: Principles and best practices of scalable real-time data systems, Nathan Marz, James Warren*

29

Sloj paketne obrade

	algoritmi sa kompletnim izračunavanjem	***inkrementalni algoritmi***
performantnost	zahteva procesiranje celokupnog glavnog skupa podataka, što zahteva veliku procesnu moć	zahteva manje procesne moći ali može rezultovati većim pogledima
otpornost na ljudske greške	izuzetno otporni na ljudske greške usled stalnog ponovnog izračunavanja pogleda	ne obuhvata podršku za lako ispravljanje grešaka i ispravke se obično rade *ad hoc* i često uključuju pretpostavke o korektnim vrednostima
generičnost algoritma	visoko generički algoritmi koji se mogu koristiti za rešavanje više problema i čija kompleksnost se ogleda u obradi glavnog skupa	algoritmi niske generičnosti koji se često prave namenski za neki domen primene

30

30

Sloj paketne obrade

- Izvršavanje upita u sloju paketne obrade
  - poželjne karakteristike
    - **mogućnost skaliranja** (engl. *scalability*) - osobina sistema da dodavanjem novih resursa očuva performanse pod uvećanim opterećenjem
      - **opterećenje** (engl. *load*) - između ostalog, obuhvata trenutnu količinu podataka, količinu podataka koja dospeva svakog dana i broj zahteva koje aplikacija opslužuje u sekundi
    - **linearno skaliranje** gde se performanse sistema linearno uvećavaju sa dodavanjem novih računara u klaster
      - troškovi rastu proporcionalno sa porastom opterećenja
      - sistemi koji nemaju mogućnost linearnog skaliranja, nisu praktično održivi
  - paradigma MapReduce
    - za izgradnju paralelno izvršivih algoritama
    - performanse MapReduce algoritama linearno rastu sa dodavanjem novih računara u klaster

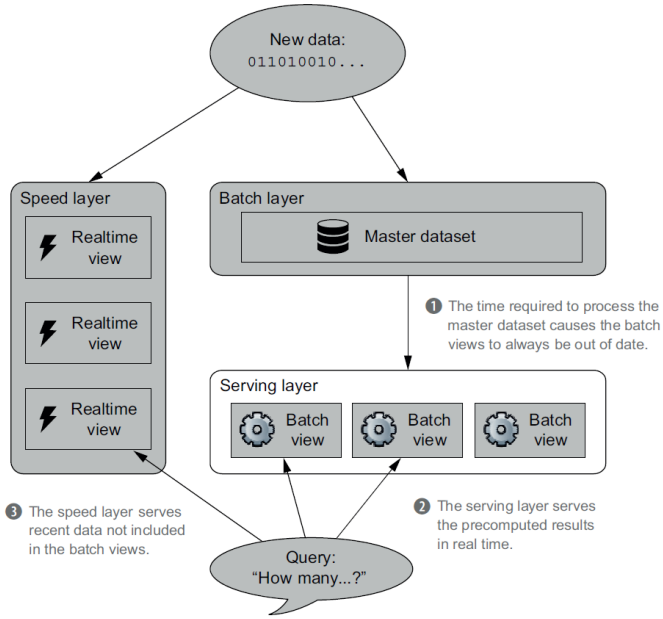
31

31

Arhitektura Lambda - Uslužni sloj

32

Uslužni sloj



izvor: *Big Data: Principles and best practices of scalable real-time data systems, Nathan Marz, James Warren*

33

33

Uslužni sloj

- Uslužni sloj
  - usko povezan sa slojem paketne obrade
  - sadrži poglede
    - koji su sračunati u sloju paketne obrade
    - nikada nemaju najsvežije podatke
      - usled konstantnog priliva novih podataka
      - svežina do na određeni trenutak u vremenu
  - sadrži indekse nad pogledima, napravljene u cilju
    - skraćenja vremena davanja odgovora na postavljeni upit
    - smanjenja potrebnih resursa za davanje odgovora na upit
  - distribuiran je na veliki broj računara radi mogućnosti skaliranja sistema
    - indeksi se takođe distribuiraju

34

34

Uslužni sloj

- Uslužni sloj - performanse
  - indeksi najviše utiču na performanse celog sloja
  - dve osnovne metrike koje treba uzeti u obzir prilikom kreiranja indeksa
    - **vreme odgovora** (engl. *latency*) - vreme potrebno da sistem da odgovor na postavljeni upit
    - **propusnost** (engl. *throughput*) - broj upita na koje može biti dat odgovor u određenom vremenskom intervalu
  - strategija kreiranja indeksa zavisi od upita i pogleda nad kojim se vrše upiti
    - distribuirana priroda indeksa zahteva pažljivo planiranje i projektovanje
    - distribuirani indeks sa sortiranim vrednostima i grupisanjem ključeva po računarima u distribuiranoj arhitekturi
      - dovodi do mnogo boljih performansi i manjeg broja čitanja sa diska
        - u odnosu na nesortirani i negrupisani indeks

35

35

Uslužni sloj

- Uslužni sloj - normalizacija i denormalizacija
  - **normalizacija**
    - smanjuje redundansu
    - usporava izvršenje upita
    - primenjuje se na podatke **u sloju paketne obrade**
  - **denormalizacija**
    - bolje performanse pri izvršenju upita
    - kompleksno održavanje baze podataka i očuvanje konzistentnosti redundantnih podataka
    - primenjuje se na podatke **u uslužnom sloju**
      - uz dodatne agregacije i transformacije
      - kako bi se unapredila performantnost
      - ne predstavlja problem jer je uvek moguće ponovo kreirati poglede iz glavnog skupa podataka

36

36

Uslužni sloj

- Uslužni sloj - baza podataka
  - pogledi se skladište u bazi podataka
  - odabir baze podataka zavisi od ispunjenosti sledećih zahteva:
    - **mogućnost paketnog upisa** (engl. *batch writable*)
      - potreba za kompletnom zamenom pogleda u bazi
      - jer se pogledi kompletno izračunavaju i dospevaju sračunati u uslužni sloj
    - **mogućnost skaliranja** (engl. *scalable*)
      - pogledi mogu biti proizvoljne veličine pa je potrebno da distribuiran uslužni sloj ima **mogućnost direktnog čitanja** (engl. *random reads*)
      - uz pomoć indeksa potrebno je omogućiti direktan pristup delovima pogleda radi što bržeg izvršenja upita
    - **otpornost na greške** (engl. *fault-tolerant*)
      - obuhvata nastavak rada nakon otkaza računara u klasteru

37

37

Uslužni sloj

- Uslužni sloj - baza podataka
  - **ne zahteva direktno upisivanje** (engl. *random writes*) u bazu podataka
    - nije potrebno delimično menjati poglede
    - značajno poboljšanje performansi
      - ne zahteva se defragmentacija baze podataka
        - kako bi se smanjio neiskorišćen prostor
      - ne zahteva sinhronizaciju operacija čitanja i pisanja
        - kako se ne bi čitale polovično upisanje vrednosti
        - moguća bolja optimizacija operacije čitanja

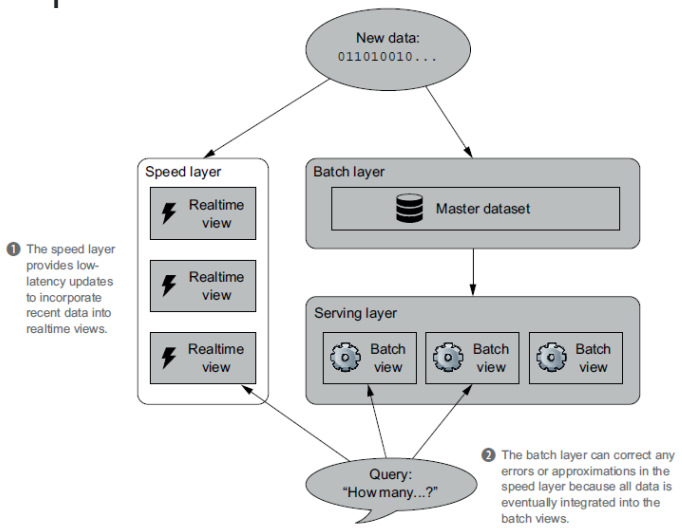
38

38

Arhitektura Lambda - Sloj obrade podataka u realnom vremenu

39

Sloj obrade podataka u realnom vremenu



izvor: *Big Data: Principles and best practices of scalable real-time data systems, Nathan Marz, James Warren*

40

40

Sloj obrade podataka u realnom vremenu

- Sloj obrade podataka u realnom vremenu
  - sloj paketne obrade podataka i uslužni sloj
    - uspešno odgovaraju na sve zahteve postavljene pred sistem koji radi sa podacima
    - osim brzog pisanja i čitanja novih podataka
  - ovaj sloj omogućava brzo pisanje i čitanje najnovijih podataka
    - sloj obrade podataka u realnom vremenu je zasnovan na **inkrementalnim algoritmima**
      - umesto na algoritmima paketne obrade podataka
  - radi samo sa podacima koji još nisu uključeni u poglede u uslužnom sloju
    - privremeni podaci
      - nestaju onog trenutka kada su uključeni u poglede
    - ukupna količina podataka u ovom sloju je mnogo manja od glavnog skupa podataka
      - lakša manipulacija nad podacima
      - inkrementalni algoritmi mogući
        - omogućeni manjom količinom podataka
        - kompleksnija implementacija i održavanje

41

41

Sloj obrade podataka u realnom vremenu

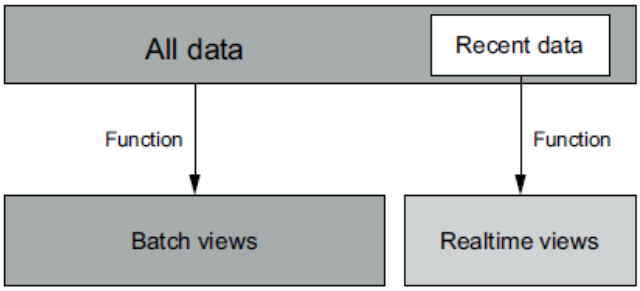
- Izračunavanje **dopunskih pogleda** (engl. *realtime views*)
  - predstavljaju poglede sa najnovijim podacima
    - koji se ne nalaze u pogledima sloja za paketnu obradu
    - ažuriraju se ubrzo po prispeću novih podataka
      - ubrzo = par milisekundi do par sekundi
      - u zavisnosti od domena primene
  - pristup 1: pokretanje funkcije nad celim skupom novih podataka
    - `realtime views = function(all new data)`
    - radi po istim principima kao i u sloju paketne obrade
    - za svaki pristigli podatak ponovno se izračunavaju pogledi
      - visoko opterećenje resursa sistema
      - dugo čekanje na odgovor od ovog sloja (do par minuta)
  - pristup 2: inkrementalno ažuriranje dodatnog pogleda najnovijim podacima
    - koji nisu bili uključeni u prethodni pogled
    - **u opštem slučaju mnogo bolji pristup**

42

42

Sloj obrade podataka u realnom vremenu

- Pristup 1: pokretanje funkcije nad celim skupom novih podataka

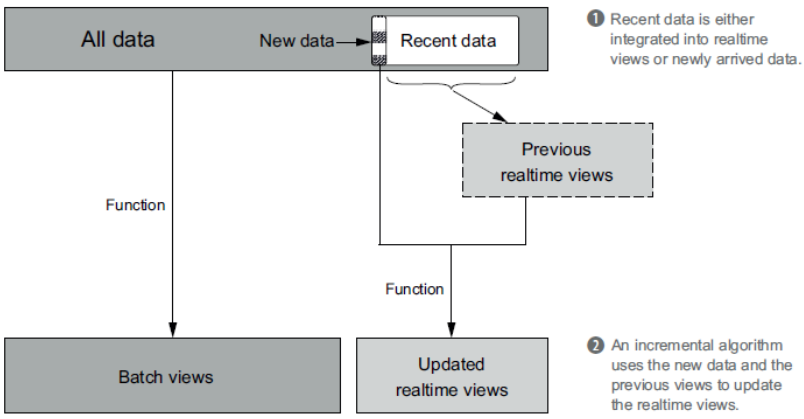


A simple strategy mirrors the batch/serving layer and computes the realtime views using all recent data as input.

izvor: *Big Data: Principles and best practices of scalable real-time data systems, Nathan Marz, James Warren*

Sloj obrade podataka u realnom vremenu

- Pristup 2: inkrementalno ažuriranje dodatnog pogleda najnovijim podacima



❶ Recent data is either integrated into realtime views or newly arrived data.

❷ An incremental algorithm uses the new data and the previous views to update the realtime views.

izvor: *Big Data: Principles and best practices of scalable real-time data systems, Nathan Marz, James Warren*

Sloj obrade podataka u realnom vremenu

- Sloj obrade podataka u realnom vremenu - baza podataka
  - dodatni pogledi se skladište u bazi podataka
  - odabir baze podataka zavisi od ispunjenosti sledećih zahteva:
    - mogućnost **direktnog čitanja** (engl. *random reads*)
      - uz pomoć indeksa potrebno je omogućiti direktan pristup delovima pogleda
        - radi što bržeg izvršenja upita
    - mogućnost **direktnog upisa** (engl. *random writes*)
      - kako bi se podržali inkrementalni algoritmi za ažuriranje dodatnih pogleda
    - mogućnost **skaliranja** (engl. *scalable*)
      - unapređenje performansi čitanja i pisanja u dodatne poglede
    - **otpornost na greške** (engl. *fault-tolerant*)
      - obuhvata nastavak rada nakon otkaza računara u klasteru
  - pogodne NoSQL baze podataka
    - npr. *Cassandra* za indeksiranje i čuvanje pogleda, *ElasticSearch* za pretragu

45

45

Sloj obrade podataka u realnom vremenu

- Sloj obrade podataka u realnom vremenu - baza podataka
  - odabir baze podataka definiše **kako** se dodatni pogledi čuvaju
  - **šta** se čuva u dodatnim pogledima?
    - zbog inkrementalnog izračunavanja, često nemaju istu strukturu kao pogledi izračunati u paketnom režimu
      - u ovim slučajevima se vrlo često koriste **aproksimacione funkcije**
      - tačan rezultat će biti na snazi prilikom sledećeg ponovnog izračunavanja pogleda u uslužnom sloju
  - **konvergentna tačnost** (engl. *eventual accuracy*)
    - u arhitekturi Lambda omogućena postojanjem dva sloja sa podacima

46

46

Sloj obrade podataka u realnom vremenu

- Inkrementalno izračunavanje dodatnih pogleda
  - bolje performanse u odnosu na algoritme sa kompletnim izračunavanjem
    - brže izračunavanje dodatnih pogleda (ažuriranjem)
  - mane u odnosu na algoritme sa kompletnim izračunavanjem
    - manja otpornost na ljudske greške
    - manja generičnost i mogućnost primene u različitim domenima i na različitim problemima
  - potrebno posebno obratiti pažnju na implementaciju inkrementalnih algoritama
    - u prisustvu **konvergentne konzistencije**

47

47

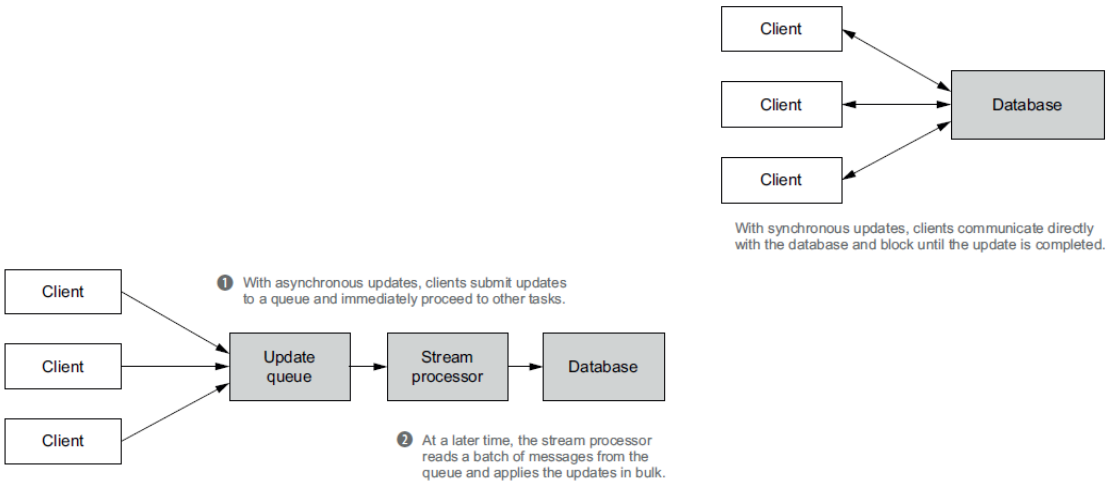
Sloj obrade podataka u realnom vremenu

- Asinhnrono i sinhrono ažuriranje dodatnih pogleda
  - sinhrono ažuriranje
    - blokira/zaključa resurse dok se ažuriranje vrednosti na završi
    - klijentski programi direktno komuniciraju sa BP
  - asinhrono ažuriranje
    - sva ažuriranja se smeštaju u red čekanja
    - red čekanja se procesira naknadno
    - sporije od sinhronog ažuriranja jer zahteva dodatne korake pre ažuriranja BP
    - omogućava mnogo veću propusnost prilikom ažuriranja
      - BP se ažurira paketno, primenom višestrukih operacija iz reda čekanja
    - red čekanja “upija” nagli porast saobraćaja
      - omogućava da sistem neometano funkcioniše u prisustvu velikog broja zahteva (engl. *traffic spikes*)

48

48

Sloj obrade podataka u realnom vremenu



izvor: *Big Data: Principles and best practices of scalable real-time data systems, Nathan Marz, James Warren*

49

49

Sloj obrade podataka u realnom vremenu

- Asinhrono i sinhrono ažuriranje dodatnih pogleda
  - postoji potreba za obe vrste ažuriranja dodatnih pogleda
  - sinhrono ažuriranje
    - u transakcionim sistemima
      - kod kojih je potrebna tačna koordinacija serverskog dela sa korisničkim interfejsom
  - asinhrono ažuriranje
    - u analitičkim sistemima
      - kod kojih se zahteva bolje upravljanje opterećenjem sistema

50

50

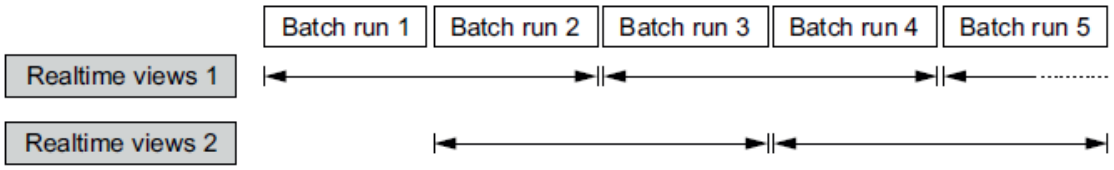
Sloj obrade podataka u realnom vremenu

- Trajanje dodatnih pogleda
  - pitanje: **do kada traje dodatni pogled?**
  - dodatni pogledi su tranzijentni
    - brišu se nakon što se izračunaju novi pogledi u uslužnom sloju
    - brišu se samo podaci koji su učestvovali u izračunavanju pogleda uslužnog sloja
  - baze podataka ne podržavaju ovakav mehanizam
    - Memcached, Redis imaju vremensko isticanje zapisa u fiksno postavljenim vremenskim trenucima
      - nije primenljivo na ovaj slučaj
      - potrebno postaviti trajanje dodatnog pogleda u zavisnosti od promenljivog vremena izračunavanja pogleda u uslužnom sloju
  - potreban generički mehanizam
    - koji održava **dva skupa dodatnih pogleda**
      - naizmenično se uzimaju jedan pa drugi nakon svakog izračunavanja pogleda u uslužnom sloju

51

51

Sloj obrade podataka u realnom vremenu



izvor: *Big Data: Principles and best practices of scalable real-time data systems, Nathan Marz, James Warren*

52

52

Arhitektura Kappa

53

Arhitektura Kappa

- Prednosti i mane Lambda arhitekture
  - prednosti
    - nepromenljivost izvornih podatka
    - visok stepen otpornosti na greške i mogućnosti skaliranja
    - dobar odnos između brzine i pouzdanosti
  - mane
    - višestruka implementacija istog algoritma obrade podataka
      - za svaki sloj u kojem se podaci obrađuju
      - teško identifikovati greške u tom kôdu
      - teško obezbediti integraciju sa novim servisima
    - održavanje i administriranje nekoliko raznorodnih distribuiranih platformi
    - ponovno procesiranje može biti izuzetno zahtevno sa stanovišta potrebnih resursa
    - model podataka u arhitekturi Lambda je teško prenosiv i teško ga je reorganizovati

54

54

Arhitektura Kappa

- Diskusija o arhitekturi Lambda
  - tvrdnja da je paketna obrada preciznija i moćnija od obrade tokova podataka
    - zasnovana je na zrelosti tehnologija i radnih okvira koji se koriste
    - na današnjem stepenu razvoja možda više i nije validna
      - alati za obradu tokova podataka mogu garantovati jaku semantiku rezultata izvršenja
  - najveći problem arhitekture Lambda jeste postojanje dve verzije kôda
    - dodatno, izuzetno je teško postići iste rezultate u dva različita distribuirana sistema
      - npr. Hadoop i Storm
    - kôd se obično piše imajući u vidu specifičnosti okruženja u kojem se izvršava
    - potencijalna rešenja
      - R1: napraviti jedan apstraktni jezik za objedinjenu implementaciju algoritma obrade koji je moguće izvršiti na oba distribuirana sistema
      - R2: koristiti samo jedan distribuirani sistem za paketnu i obradu u realnom vremenu

55

55

Arhitektura Kappa

- Diskusija o arhitekturi Lambda
  - jedan distribuirani sistem za paketnu i obradu u realnom vremenu
    - u zavisnosti od slučaja korišćenja
      - Hadoop ako vreme odziva nije presudan faktor
      - Storm ako je vreme odziva od izuzetnog značaja
    - na današnjem stepenu razvoja, sistemi za obradu tokova podataka su u mogućnosti da odgovore na zahteve paketne i obrade podataka u realnom vremenu
      - sva obrada se svodi na obradu tokova podataka
      - omogućavaju paralelnu obradu podataka
      - mogu da čuvaju podatke i nakon što su obrađeni
        - u nekom stalnom skladištu podataka

56

56

Arhitektura Kappa

- Arhitektura Kappa - osnove pristupa
  - koristiti Kafku ili neki drugi izvor toka podataka
    - mora biti u mogućnosti da čuva podatke neki duži vremenski period
    - recimo 30 dana od dospeća podatka
  - za izračunavanje pogleda nad kojim se vrše upiti koristiti obrađivače toka podatka
    - u slučaju ponovnog izračunavanja pogleda kreira se nova instanca obrađivača
      - koji se izvršava paralelno sa postojećom instancom i uzima u obzir sve prethodno dospele podatke
    - nakon izračunavanja pogleda upiti se izračunavaju nad novim pogledom
    - stara instanca obrađivača se zaustavlja, stari pogledi se brišu iz baze podataka
  - ponovno izračunavanje pogleda
    - isključivo kada dođe do promene kôda obrađivača tokova podataka

57

57

Arhitektura Kappa

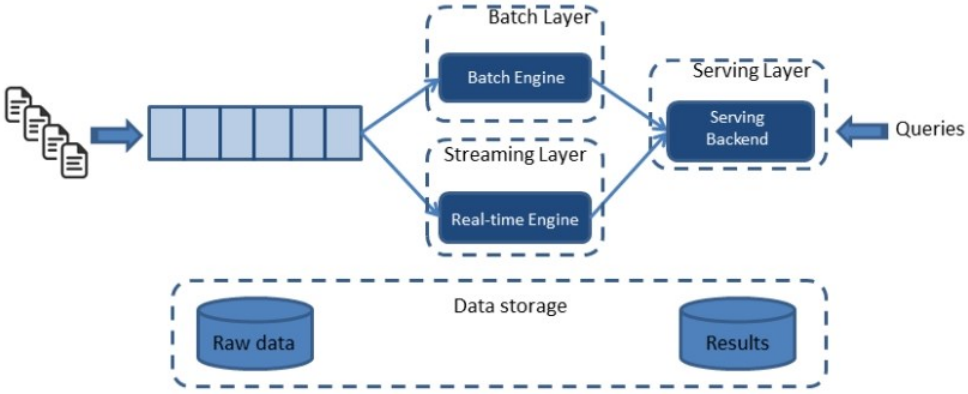
- Arhitektura Kappa - osnovni postulati
  - **sve je tok podataka**
    - paketna obrada postaje podskup obrade toka podataka
  - **nepromenljivi podaci**
    - izvorni podaci ostaju nepromenljivi dok se pogledi izračunavaju
  - **jedan radni okvir za obradu**
    - postoji samo jedan sistem/platforma za obradu podataka
    - implementacija, održavanje i unapređivanje aplikacije su pojednostavljeni
  - **mogućnost ponovljene obrade**
    - moguće je ponoviti tok izvornih podataka od bilo kojeg traženog trenutka
      - koji je obuhvaćen unapred definisanim trajanjem čuvanja podataka

58

58

Arhitektura Kappa

- Arhitektura Lambda



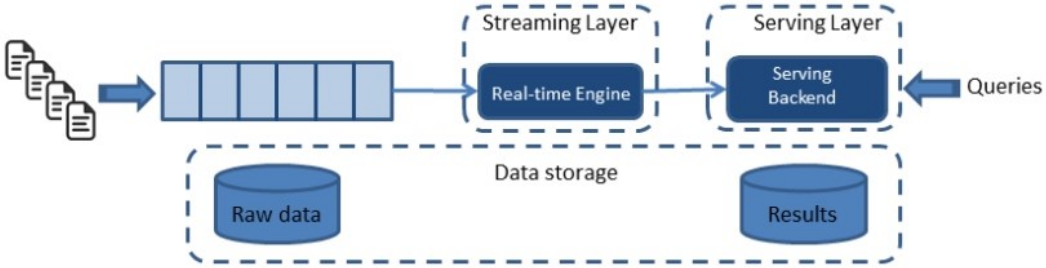
59

izvor: *Applying the Kappa architecture in the telco industry, Nicolas Seyvet, Ignacio Mulas Viela*

59

Arhitektura Kappa

- Arhitektura Kappa



60

izvor: *Applying the Kappa architecture in the telco industry, Nicolas Seyvet, Ignacio Mulas Viela*

60

Arhitektura Kappa

- Arhitektura Kappa - primena
  - nije zamena za arhitekturu Lambda
    - već alternativa u slučaju kada paketna obrada ne zadovoljava zahteve u domenu primene i rada sistema
  - moguće je primeniti arhitekturu Kappa:
    - za izgradnju delova socijalnih mreža
    - u IoT domenu
    - za obradu podataka od monitoring sistema

61

61

Arhitektura Kappa

- Arhitektura Kappa - prednosti i mane
  - prednosti
    - olakšava razvoj sistema kod kojih se mogu lako primeniti inkrementalni algoritmi obrade podataka
    - ponovno procesiranje je neophodno samo kada se promeni kôd obrađivača
    - upiti se izvršavaju samo nad jednim pogledom
  - mane
    - nedostatak sloja paketne obrade može dovesti do grešaka u obradi podataka
    - ažuriranje baze podataka koja nije optimizovana za paketnu obradu, može biti problematična sa stanovišta performansi

62

62

Literatura

- Nathan Marz, James Warren - *Big Data: Principles and best practices of scalable real-time data systems*
- Jay Kreps - *Questioning the Lambda Architecture* https://www.oreilly.com/ideas/questioning-the-lambda-architecture

63