

# Odgovori na II kolokvijum - Računarstvo u Oblaku

## 1. Upravljanje resursima

1. Šta je upravljanje resursima? Na koje kriterijume za evaluaciju sistema direktno utiče?
  - Upravljanje resursima je osnovna funkcija sistema. Utiče na cenu, performanse i funkcionalnost.
2. Da bi se postiglo efikasno upravljanje resursima u cloudu kakva optimizacija je neophodna?
  - Višeciljna optimizacija gde se upravljanje ne sme fokusirati samo na jednu stvar.
3. Jedan od primarnih ciljeva upravljanja resursima u oblaku je obezbeđivanje elastičnosti. Koji je ključni izazov sa kojim se upravljanje resursima suočava u ovakvim okruženjima?
  - Veliko, fluktuirajuće opterećenje.
4. Koja su dva osnovna pristupa (načina) upravljanju resursima?
  - Predviđanje opterećenja i alociranje u napred, i korišćenje autoscaling algoritma koji može raditi samo ako postoje slobodni resursi i samo ako je monitoring sistem dobar pa će znati koji su to resursi koji mogu da se realociraju.
5. Zašto je centralizovani pristup upravljanju resursima tema polemika od nastanka cloud okruženja?
  - Jer se smatra da centralizovana kontrola teško može da se izbori sa brzim i nepredvidivim promenama i da obezbedi kontinualnu isporuku.
6. Koje su osnovne grupe politika upravljanja resursima u oblaku? Šta je cilj svake od njih?
  - Kontrola prihvatanja (sprečava sistem da prihvati rad na zadacima zbog neke definisane politike); Alokacija kapaciteta (kako se alociraju resursi); Balansiranje opterećenja (kako učiniti da je opterećenje jednako podeljeno na sve instance servera); Optimizacija utroška energije (kako minimizovati potrošenu energiju); Quality of Service (kako ispuniti sve zahteve iz ugovora).
7. Šta je problem raspoređivanja (scheduling) u računarskom sistemu?
  - Odlučivanje kako dodeliti resurse sistema različitim zadacima i korisnicima.
8. Šta je "raspoređivač" (scheduler) i koja je njegova funkcija?
  - Raspoređivač je program koji implementira algoritam raspoređivanja resursa.
9. Koje su ključne odluke koje raspoređivač donosi u pogledu resursa?
  - Količini resursa koju je potrebno dodeliti i trajanju te dodele resursa nekom zadatku ili korisniku.
10. Šta su ključne željene karakteristike algoritma za raspoređivanje resursa?
  - Efikasnost, pravednost, sprečavanje gladi.
11. Koji tipovi vremenskog ograničenja za izvršavanje zadataka se najčešće identifikuju? Za koje aplikacije je primerena određena vrsta vremenskog ograničenja?

- Postoje tvrda i meka ograničenja, gde tvrda ograničenja označavaju zadatke koji moraju da se završe u nekom roku i postoje penali ako se ne završe, a meka ograničenja su zadaci koji ne moraju nužno u tom roku da se završe i nemaju penale. Aplikacije multimedije imaju meka vremenska ograničenja, a aplikacije u realnom vremenu imaju tvrda vremenska ograničenja.

12. Kada se radi raspoređivanje na VM, koje dodatne faktore treba uzeti u obzir?

- Postoji veći broj raznovrsnih zadataka, konačan broj zadataka se može obavljati na svakom serveru, zadaci mogu imati drugačije potrebe u smislu vremena za izvršavanje, performanse, izolacija resursa između različitih zadataka.

13. Šta je Proportional Share raspodela resursa?

- Neki zadaci zahtevaju više CPU vremena pa im je dodeljeno proporciono više.

14. Šta je Fair Share raspodela resursa?

- Svaki zadatak bi trebalo da dobije istu količinu CPU vremena.

15. Šta je Work Conserving, a šta Non Work Conserving režim rada raspoređivača?

- Kod Work Conserving moda, cilj je održati procesor u radu pa će CPU nastaviti biti korišćen dok god ima posla. Kod Non Work Conserving moda, cilj je da je svakom zadatku dodeljen udeo i kad svaki zadatak potroši udeo, CPU prestaje sa radom.

16. Šta je Preemptive, a šta Non Preemptive metod funkcionisanja raspoređivača?

- Kod Preemptive moda CPU-a, rad jednog zadatka može biti prekinut i preraspodela resursa se radi kada dođe novi zadatak. Kod Non Preemptive moda CPU-a, rad jednog zadatka ne može biti prekinut i preraspodela resursa se radi kada zadatak završi sa korišćenjem CPU-a.

17. Šta je Workload management (WM)? Koja su dva osnovna pristupa WM?

- Workload management je sistem koji aplikacijama precizno dodeljuje resurse bilo koje vrste. Aplikacije govore koje performanse su im potrebne, a WM dodeljuje resurse na osnovu toga. Postoji statički WM (resursi se aplikaciji dodeljuju jednom) i dinamički WM (resursi se prilagođavaju trenutnim potrebama aplikacije).

18. Šta je BVT algoritam?

- Borrowed Virtual Time. Pravična raspodela, dodeljuje CPU vreme zadacima sa najmanjom vrednošću virtuelnog vremena. Preemptive, WC. Ne podržava NWC pa ograničava izolaciju.

19. Šta je SEDF algoritam?

- Simple Earliest Deadline First. Svaki zadatak specificira vremenski odsečak, period i da li može dobiti ekstra vreme (WC mod). Preemptive, WC i NWC. Pravičnost zavisi od dužine perioda. Nedostaje globalno balansiranje na multiprocesorskim okruženjima.

20. Šta je Credit Scheduler algoritam?

- Obezbeđuje da ni jedan fizički CPU nije u idle režimu ako ima stvarnog posla na virtuelnim CPU-ovima. Za svaku VM, raspoređivač dodeljuje težinu i limit. Limit je 0 = WC režim rada za tu VM. Limit nije 0 = NWC režim rada za tu VM. Svaki virtuelni

CPU dobija 30ms za izvršavanje pre nego što biva zamenjen. Non preemptive, WC i NWC. Globalno balansiranje na multiprocesorskim okruženjima.

## 2. Aplikacije i paradigme

21. Koji su ključni ekonomski faktori koji čine aplikacije u oblaku prihvatljivim za korisnike?
  - Veoma male početne investicije u infrastrukturu. Korisnik instant može da iskoristi ubrzanje, dobijajući približno  $n$  puta brzi program ( $n$  - predpostavljeni broj deljivosti programa). Programeri se većinski fokusiraju na funkcionalne zahteve jer se ne brinu o infrastrukturi. Imaju just-in-time-infrastructure. Ogromna elasticnost.
22. Koji su ključni ekonomski faktori koji čine aplikacije u oblaku prihvatljivim za ponuđače?
  - Jako velika iskoriscenost njihovih resursa. Uspeh ponudjaca jako zavisi od toga da li će privuci ogromnu masu korisnika sa dobrim uslugama. Vise je usmereno ka enterprise korisnicima nego pre, gde je bilo usmereno samo istrazivacima.
23. Koja je ključna osobina aplikacija koje su izrazito pogodne za iskorištavanje prednosti okruženja u oblaku?
  - Glavna osobina aplikacija koje dobro iskoristavaju prednosti okruženja u oblaku je to da se te aplikacije dobro dele na segmente koji se dobro paralelizuju.
24. Koji su ključni izazovi za aplikacije u oblaku?
  - Disbalans između računarskih resursa, brzine I/O i propusnosti komunikacionih kanala. Iako postoji elasticnost, podaci moraju biti smesteni blizu procesnih jedinica na kojim će se izvršavati. Bezbedonosna izolacija za multitenant sisteme. Otkazi cvorova na kojima aplikacija radi. Koliko logovati, nekad može biti previše, nekad premalo.
25. Koji su tipične grupe aplikacija u oblaku?
  - Processing pipelines, batch processing i web aplikacije.
26. Ključna osobina distribuiranih, pa i aplikacija u oblaku je koordinacija aktivnosti. Koji načini postoje za koordinisanje aktivnosti?
  - Staticki i dinamicki (workflow se menja tokom izrade). Podela po načinu koordinacije tokom izvršavanja: strong i weak coordination models.
27. Šta je ideja koordinacije na principu konačnog automata?
  - Servis se sagledava kao konacni automat, koji izvršava odredjene operacije kada dobije komandu od klijenta, a izvršavanje ga dovodi u sledece definisano stanje.
28. Šta je ZooKeeper?
  - ZooKeeper je distribuirani koordinacioni servis koji se zasniva na modelu konacnog automata.
29. Koji su osnovni principi funkcionisanja ZooKeeper-a i koje garancije on pruža?
  - ZooKeeper se skida i instalira na vise servera. Osnovni principi funkcionisanja ZooKeeper-a su to da serveri u grupi biraju vodeceg. Baza se replicira na svaku instancu i replike su konzistentne. Klijent se može povezati na bilo koji server u grupi. ZooKeeper garantuje: atomicnost, sekvencijalnu konzistentnost update-a (update-i

se primenjuju redosledom kojim su poslati), klijenti vide jednu sliku sistema, trajnost update-a i pouzdanost.

30. Šta je MapReduce programski model?

- MapReduce se zasniva na ideji paralelog procesiranja aplikacija koje obradjuju veliku kolicinu podataka i omogucavaju proizvoljnu podelu posla. Podaci se dele u blokove, svaki blok se dodeli jednoj instanci procesa, kada sve instance zavrse posao rezultati se spajaju.

31. Koje su ključne komponente u MapReduce programskom modelu?

- Map funkcija (radi racunanje) i reduce funkcija (spaja rezultat racunanja).

32. Šta je uloga mastera u MapReduce? Šta je uloga Workera?

- U MapReduce modelu, uloga mastera je da dodeljuje Map i Reduce zadatke slobodnim serverima, donosi M+R odluka i cuva MxR stanja workera. Uloga workera koji radi Map je da cita dobijene podatka i pretvara ih u medjurezultat. Uloga workera koji radi Reduce je da podatke iz bafera u kojima su Map funkcije pisale, agregira.

33. Kako MapReduce resava problem otkaza čvorova?

- Postoje dva tipa otkaza: otkazi worker cvorova i otkazi master cvorova. Za otkaze worker cvorova: master periodično salje heartbeat proveru da vidi da li je worker otkazao. Reduce workeri koji su osteceni otkazom Map workera se preraspoređuju. Za otkaze master cvorova: obrada se prekida i pocinje iznova.

34. Zašto je lokalnost podataka bitna za performanse MapReduce programskog modela?

- Sto su podaci blizi, Map i Reduce workeri ce ih brze obradivati i bice lakse citanje i upisivanje u buffer.

35. Koji mehanizam MapReduce programski model koristi da bi poboljšao performanse kada u mreži postoje čvorovi sa problemima u obradi (stragglers)?

- Kada je operacija blizu kraja (ostali su samo spori cvorovi), master cvor pokrece rezervne obrade preostalih podataka. Onda, koji god cvor da prvi obradi podatke, njegov rezultat se uzima.

36. Koje dodatne izazove donosi upotreba MapReduce programskog modela na velikim heterogenim klasterima?

- Dosadasnje resenje stragglers cvorova je bilo da pokrecemo nove rezervne cvorove nad jos neobradjenim podacima. Ovo je problem kod velikih heterogenih klastera jer je performansa jako sarolika.

37. Od kojih pretpostavki o načinu funkcionisanja čvorova se polazi u homogenim okruženjima?

- Cvorovi obavljaju posao približno istom brzinom. Zadaci napreduju konstantnom brzinom tokom izvsavanja. Nema troskova pokretanja spekulativnog zadatka. Progres jednog zadatka je približno jedan njegovom udelu u ukupnom poslu (**jako debilno napisano**). Zadaci se izvsavaju u talasima. Razliciti zadaci iste kategorije sadrze okvirno istu kolicinu posla.

38. Koje pretpostavke o načinu rada čvorova ne važe u heterogenim virtuelizovanim okruženjima? (**rec virtuelizovana nije spominjana na slajdovima ali je prakticno isto znacenje**)

- Ne vaze predpostavke o slicnoj brzini cvorova i konstantnoj brzini izvršavanja zadataka. Previše spekulativnih zadataka oduzima resurse drugim zadacima. Zadaci iz različitih generacija se izvršavaju konkurentno: srednji indikator progressa se jako menja.
39. Za koje tipove okruženja je progress score dovoljno dobra mera za odlučivanje o spekulativnim zadacima?
- Za homogena okruženja (**ne postoji direktan odgovor na ovo pitanje, ali ovo je moja pretpostavka**).
40. Koja mera se koristi za raspoređivanje spekulativnih zadataka u heterogenim virtualizovanim okruženjima?
- Longest Approximate Time to End (LATE) mera (**opet, ne postoji direktan odgovor na ovo pitanje, ali ovo je moja pretpostavka**).
41. Koja je osnovna ideja LATE raspoređivača zadataka?
- Napraviti backup tj. replika zadatke za one kojima je procenjeno vreme do kraja najduže.

### 3. Mesos i Omega

42. Zašto je na velikim klasterima, na kojima se pokreću različite vrste aplikacija neophodno imati dodatni servis koju upravlja dodelom resursa aplikacijama? Zašto je oslanjanje na raspoređivač određene vrste aplikacija ne bi bilo dobro rešenje za ostale?
- Potrebno je imati dodatni servis koji upravlja dodelom resursa aplikacijama jer je jako tesko da jedno izvrsno okruzenje, koje je specifično za određenu vrstu aplikacija, upravlja resursima na velikim klasterima.
43. Šta je statička podela klastera? Dobre strane? Loše strane?
- Staticka podela klastera je kada se jedan tip aplikacije izvršava na jednoj particiji. Lose iskoriscenje resursa, tvrd princip particionisanja ne mora da se poklapa sa promenljivim potrebama aplikacije, nepodudarnost izmedju granularnosti statickih particija i potreba trenutno aktivnih aplikacija.
44. Šta je Mesos? Koji je princip rada Mesos-a?
- Omogucava izvršavanje različitih okruženja na jednom klasteru. Tanak deljeni sloj omogucava deljenje resursa na finom nivou granularnosti izmedju različitih okruzenja, okruzenjima pruza interfejs preko kog oni pristupaju resursima u klasteru.
45. Koje su osnovne komponente Mesos-a?
- Mesos master, mesos slave na svakom cvoru, frameworks, framework schedulers.
46. Koja je uloga Mesos mastera, a koja Mesos slave komponente?
- Master: upravlja deljenjem resursa izmedju različitih okruzenja na finom nivou granularnosti. Slave: izvršavanje taskova.
47. Da li aplikativni scheduler-i i dalje obavljaju svoju ulogu u Mesos-u?
- Da, Mesos master odlucuje koliko ce pojedinacna okruzenja dobiti resursa, a aplikativni scheduler-i tih okruzenja odlucuje kako ce ih koristiti.
48. Zašto su za Mesos bitne liste filtera?

- Pravila definisana u listi filtera se jako brzo evaluiraju.
49. Šta je Omega? Da li kod Omega okruženja postoji komponenta koja centralno donosi odluku o dodeli resursa?
- Omega je rasporedjivac za velike klastere racunara. Ne postoji centralizovani rasporedjivac.
50. Na koji način Omega obezbeđuje informaciju o zauzetim resursima raspoređivačima pojedinih aplikacija?
- Rasporedjivac pojedinačne aplikacije donosi odluku, atomično je upisuje u deljenu master state tabelu, samo jedan od tih upisa može da uspe (atomičnost), rasporedjivac azurira svoju lokalnu kopiju i ako je potrebno, radi ponovo preraspoređivanje.

## 4. Sistemi skladistenja (storage systems)

51. Šta je GFS? Koji je osnovni princip rada?
- Google File System, koristi hiljade sistema za skladistenje, omogućavajući ogromno skladište (PB) za raznovrsne potrebe.
52. Koje tipične obrasce pristupa podacima u fajlovima pretpostavlja GFS?
- Velika "streaming" citanja i mala nasumična citanja.
53. Koja je uloga master-a, a koja chunkserver-a kod GFS?
- Master održava sve metapodatke sistema datoteka: kontrolu pristupa, mapiranje iz datoteka u blokove, lokacije blokova, itd. Chunkserver čuva blokove.
54. Šta je BigTable?
- BigTable je distribuirano skladište za strukturane podatke koje je dizajnirano da se skalira na velike skupove podataka.
55. Po kom principu se BigTable deli na tablete?
- Jedan tablet je jedan leksikografski sortiran opseg. Bliski tableti se čuvaju na istom serveru. Dobar izbor ključa omogućava dobru lokalnost.
56. Za koje potrebe BigTable koristi GFS, SSTable i Chubby?
- GFS se koristi za skladistenje podataka, SSTable se koristi za čuvanje samih BigTable podataka (napredni algoritmi referenca), Chubby servis se koristi za zaključavanje.

## 5. Serverless

57. Šta je osnovna ideja Serverless arhitekture?
- Serverless arhitektura omogućava da se aplikacije ne održavaju stalno na serverima. Mogu znatno smanjiti operativne troškove, ali trajno vezuju za neku platformu.
58. Koja dva osnovna pristupa Serverless možemo uočiti? BaaS? FaaS?
- BaaS je "Backend as a Service", a FaaS je "Functions as a Service".
59. Ključne karakteristike FaaS pristupa?

- Omogućava mogućnost izvršavanja backend koda bez opterećenja održavanja servera i dugotrajnih serverskih aplikacija. Funkcije se aktiviraju i deaktiviraju po potrebi.

#### 60. Prednosti i mane FaaS?

- Prednosti: horizontalno skaliranje je u potpunosti automatizovano, smanjeni operativni troškovi, smanjena kompleksnost isporuke. Mane: čvrsto vezivanje za platformu cloud provajdera, multitenant problemi, bezbednost, nema stanja, postoji limit dužine izvršavanja, hladni start.