Seminarski rad

Tema: *Konsenzus algoritmi*

Predmet: *Distribuirani i paralelni algoritmi i strukture podataka*

Mentor: Dušan Gajić

Student: Stefan Aleksić

Novi Sad, novembar 2022.

Sadržaj

1 Uvod

Blokčejn (*eng. Blockchain*) ima veliki potencijal da promeni način na koji komuniciramo i obavljamo transakcije na Internetu. Neke od karakteristika ove notorne strukture podataka podrazumevaju: decentralizovanost, bezbednost, nepromenljivost i transparentnost. Jedan od većih problema, odnosno stigmi u vezi blokčejn tehnologije jeste nedostatak treće strane (*eng. third praty*) koja vrši verifikaciju transakcija, na koju se većina navikla. Samim tim se u pitanje dovodi poverenje u ovakav jedan sistem, jer naizgled može biti sklon napadima poput dvostruke potrošnje (*eng. double-spending*). Da bi ovakav sistem funkcionisao na globalnom nivou, od ključnog značaja je bezbednosni algoritam za sinhronizaciju decentralizovanog sistema, odnosno algoritam za ostvarivanje konsenzusa u distribuiranom sistemu.

Razmatrajući definiciju konsenzus algoritma na najjednostavniji način možemo uočiti da se sastoji iz dva različita pojma – *konsenzus* i *algoritam*. Konsenzus se odnosi na dogovor između dve ili više zainteresovanih strana, dok algoritam, u programskom smislu, može biti definisan kao skup pravila koja pomažu da se reši problem. Dakle, algoritam konsenzusa je okosnica blokčejn mreža. U tehničkom smislu, algoritam konsenzusa nije ništa drugo do postupak kroz koji se svi učesnici u blokčejn mreži slažu oko stanja sistema, u ovom slučaju oko stanja distribuirane knjige (*eng. ledger*) [1]. Članovi distribuiranog sistema dolaze do konsenzusa u zavisnosti od odluke većine i ostaju pri odluci koja svima ide u korist. Ovo pomaže u izgradnji poverenja između nepoznatih čvorova i čini mrežu sigurnijom, te se samim tim sistem štiti od zlonamernih napada.

Upotrebna vrednost ovih algoritama zavisi od njihove efikasnosti, odnosno od prosečne mere zadovoljavanja sledećih faktora:

1. **Postizanje sporazuma** – korisnici bi trebalo da budu u mogućnosti da postignu sporazum bez potrebe verovanja jedni drugima [2].
2. **Učešće svih** – podjednako učešće svih članova blokčejn mreže. Niko ne treba da nedostaje i treba da se računa svačiji glas [2].
3. **Jednaka vrednost** – glas svakog učesnika treba da ima istu vrednost i jednaku važnost [2].
4. **Bez dvostruke potrošnje** – bez kopiranja detalja transakcija, čime bi se trošila ista kriptovaluta više puta.
5. **Sistem otporan na greške** – mehanizam konsenzusa treba da obezbedi pouzdanost blokčejn mreže čak i u slučaju neuspeha i potencijalnih pretnji.
6. **Jednaka aktivnost** – svaki učesnik u mreži treba da bude podjednako aktivan i nijedan ne treba da ima više odgovornosti od drugog.

Nadalje sledi opis pojedinih algoritama za ostvarivanje konsenzusa u distribuiranom sistemu. Neki od njih ne zadovoljavaju navedene uslove, međutim, svaki od njih ima isti cilj – postizanje većinski priznatog stanja sistema. Bitno je napomenuti da dva trenutno najvažnija algoritma za ostvarivanje konsenzusa u aktuelnim blokčejn mrežama su *Dokaz o obavljenom poslu* (*eng. Proof of work*) i *Dokaz o zakupu* (*eng. Proof of stake*). Ovi algoritmi nisu opisani u radu, a s obzirom da se rad oslanja na poznavanje pomenutih, preporučuje se upoznavanje sa njima pre daljeg čitanja rada.

2 Dokaz o sagorevanju [3]

Dokaz o sagorevanju (*eng. Proof of burn*) je algoritam za postizanje konsenzusa, koji je izumeo Iain Stewart. Ideja iza ovog algoritma je bila da se minimizuje potrošnja električne energije i hardverskih resursa, problem visokih razmera u algoritmima zasnovanim na algoritmu *dokaza o obaveljnom poslu*. Način na koji čvor u mreži dokazuje svoju verodostojnost u ovom algoritmu jeste kroz spaljivanje svog novca, naravno u kriptovaluti koja se koristi u samoj mreži.

Javna i u mreži dobro poznata adresa, nazvana etar adresom (*eng. Eater address*) upravo služi kao mesto gde se novac šalje na sagorevanje. Razlog za to je što nijedan čvor u mreži nema pristup novčaniku koji je vezan za etar adresu, međutim, s obzirom da se radi o javnoj adresi, svi mogu da nadgledaju transakcije, odnosno količinu novca i čvor koji vrši slanje na istu.

Što je veći broj valute poslat na spaljivanje od strane nekog čvora, to je veća verovatnoća da će on biti izabran za rudarenje (*eng. mining*) sledećeg bloka. Bitno je napomenuti da s obzirom da se radi o algoritmu koji je baziran na lutriji, postoji određeni vremenski period između licitiranja za rudarenje i samog rudarenja bloka, kako odabrani čvor ne bi validirao blok koji sadrži njegov ulog i time mogao da naruši jednaka pravila za sve. Čvor prima nagradu samo ako uspešno potvrdi transakcije i kreira novi blok, inače se novac poslat na spaljivanje potpuno baca u etar.

Sama ideja iza algoritma jeste da zadrži koncept ulaganja resursa zarad rudarenja blokova, sa ograničenjem da ti resursi budu interni resursi sistema. Sagorevanje rešava problem potrošnje eksternih resursa, međutim uvodi novi problem – ponestajanje kriptovalute. Takođe, sama vrednost valute se koriguje kad god je novi blok kreiran. Ovo onemogućava rudarima da za jednokratno sagorevanje imaju konstantnu šansu da kreiraju blok. Na drugu stranu, rudar koji kreira blok biva nagrađen adekvatnom količinom kriptovalute, kako bi se određena količina novca održavala u sistemu. Ova nagrada dolazi kao deo naplate transakcija korisnicima mreže.

Pristup sagorevanja ima prednosti u odnosu na algoritam *dokaza o obavljenom poslu* i promoviše učešće u mreži. Za što veći broj ulagača, odnosno sagorevača, vrednost valute je stabilnija. Takođe, ideja iza slanja na nepristupačan novčanik uvodi jednakost i transparentnost u decentralizovani sistem. Loša strana je to što se i dalje troše resursi, kao i činjenica da je lako da sistemom zavladaju bogati i postanu još bogatiji. Trenutne blokčejn mreže koje koriste ovaj algoritam su: *Slimcoin* (*SLM*), *Counterparty* (*XCP*) i *Factom* (*FCT*).

3 Dokaz o ovlašćenju

Dokaz o ovlašćenju (*eng. Proof of Authority*) je posebno rasprostranjen u privatnim blokčejn sistemima. Ovaj pristup je varijacija algoritma *dokaza o zalogu*. Čvorove koji vrše validiranje blokova obično imenuju operateri sistema kao autoritete u okviru komisije same mreže, međutim mogu biti odabrani i kroz zalaganje male količine kriptovalute i svoje reputacije. Ovi čvorovi ne moraju imati zalog da bi stvarali i validirali blokove, međutim maliciozne akcije se kažnjavaju narušavanjem reputacije u mreži. Kredibilitet odabranih čvorova kao autoriteta osigurava nesmetan rad mreže. Dakle, ovlašćeni čvorovi mogu isključiti čvorove koji krše pravila distribuiranog sistema [4].

Prema algoritmu *dokaza o ovlašćenju*, čvor koji vrši kreiranje novog bloka, takozvani lider rudnika (*eng. mining leader*), se bira iz skupa čvorova koji predstavljaju autoritete sistema, i to kada se ostali ovlašćeni čvorovi slože sa odabirom. Takođe, većina čvorova koji su ovlašćeni moraju da potvrde novi blok kako bi on bio nadovezan na prethodni [5].

S obzirom da dokaz o ovlašćenosti ne koristi nikakve resurse za proces kreiranja blokova, ovaj algoritam konzumira jako malo komputacione energije. Transakcije mogu biti obrađene mnogo brže i efikasnije u odnosu na algoritma dokaza o obavljenom poslu, što samim tim povećava i samu skalabilnost sistema. Ovlašćeni čvorovi nadgledaju da svaki čvor bude pristalica trenutnog stanja sistema [3].

Jasno je da s obzirom da ovlašćeni čvorovi imaju bitnu ulogu u sistemu, kao validatori i kreatori novih blokova, moraju biti dodatno zaštićeni od napada i ostalih formi manipulacije nad sistemom kako bi se održala njegova bezbednost. Takođe, vidno je da se radi o nekoj vrsti pristupa koji liči na centralizovan pristup, te se uglavnom koristi kao rešenje za korporativne mreže, pre nego za javne. *Microsoft Azure* koristi ovaj konsenzus algoritam [4].

4 Dokaz o verodostrojnosti [4]

Dokaz o verodostojnosti (*eng. Proof of believability*) je još uvek relativno mlad i stoga još uvek nije široko korišćen pristup u poređenju sa ostalim algoritmima za ostvarivanje konsenzusa. Mehanizam u pozadini je zasnovan na istraživanju prethodnog ponašanja i doprinosa svakog od čvorova u mreži. *IOST* je razvio ovaj pristup 2018. godine i obećao bolje performanse u odnosu na algoritam *dokaza o zalogu.*

Blokovi i transakcije su validirane od strane veća koje odlučuje o verodostojnosti istih. Veće, odnosno komitet se trenutno sastoji od 17 čvorova i svaki član mreže može da učestvuje ukoliko poseduje odgovarajuće tokene. Tokeni za pristup veću se nazivaju *Servi* i analogni su *Ether*-u u *Ethereum*-u. Sastav komisije se menja na svakih desetak minuta tako što se bira 17 čvorova sa najvećim brojem *Servi* tokena u novčaniku. Sve dok su na vlasti, čvorovi koji čine komitet imaju mogućnost da kreiraju blokove i dobijaju odgovarajuće nagrade za to. Interesantan aspekt ovog algoritma jeste mehanizam da svi članovi komiteta na kraju svoje vladavine iz svog novčanika gube onoliko Servi tokena koliko se nalazi u novčaniku poslednjeg, sedamnaestog člana komiteta. Ovo znači da na kraju vladavine, sedamnaesti čvor nema nijedan *Servi* token, a ostalima je umanjen iznos koji je izgubio sedamnaesti čvor.

Sama logika mehanizma jeste ta da ukoliko se odaberu čvorovi sa najvećim balansom i svima njima umanji iznos poslednjeg, pa čak i onog ko ima najmanje tokena, ostatak mreže, odnosno čvorovi koji nisu odabrani će automatski imati barem jedno slobodno mesto u komitetu, a najverovatnije i više, kada dođe odabir novog komiteta. Ovo omogućava da se sastav komiteta konstantno menja, pa je i šansa da se vrši validiranje blokova i dobijanje nagrada za to mnogo veća u odnosu na algoritam *dokaza o zalogu*.

5 Delegirana vizantijska tolerancija grešaka (dBFT)

Pristup „delegirane vizantijske tolerancije grešaka“ (*eng. Designated Byzantine Fault Tolerance*) je zahtevan algoritam koji ima za cilj da olakša ostvarivanje konsenzusa u blokčejnu. Poreklo ovog pristupa je iz teorije igara koje se našlo u informatici pod nazivom „Problem Vizantijskih Generala” [6]. Ova dilema iz teorije igara pokazuje komunikacijske probleme koji nastaju u okviru grupe generala u toku važnog zajedničkog napada [7]. Dilema pretpostavlja da svaki general ima svoju vojsku koja je postavila logore na različitim mestima, na primer, grada. Generali moraju da se dogovore o napadu ili povlačenju. Pored toga, doneta odluka se ne može poništiti, a odluke pojedinačnih generala moraju biti sprovedene u isto vreme. Poteškoća je u tome što jedan ili više generala mogu biti izdajnici, što zauzvrat znači da bi mogli da prave lažne izjave o njihovom postupku. Da stvar bude gora, komunikacija između generala se obavlja isključivo pomoću kurira. Napad ne uspeva, ukoliko se nije postigao konsenzus [4].

Prevedeno na blokčejn terminologiju, generali su zapravo čvorovi mreže. Većina čvorova u distribuiranom sistemu se mora složiti i izvršiti istu akciju kako bi se mreža održala bezbednom i stabilnom. Ovo zahteva da u sistemu postoji dve trećine pouzdanih čvorova. Otpor na greške vizantijskog, odnosno proizvoljnog tipa, zapravo predstavlja neometan rad distribuiranog sistema kada jedna trećina njegovih čvorova daje nevalidne rezultate, odnosno ima maliciozne akcije [8].

Algoritam polazi sa pretpostavkom da učesnici u komunikaciji, odnosno čvorovi distribuiranog sistema imaju motiv da interaguju sa sistemom na pravilan način, bez malicioznih akcija. Delegati su zapravo čvorovi, odabrani većinski od strane ostalih učesnika kao dovoljno pošteni i njima se daje poverenje za održavanje integriteta blokčejn mreže [4].

Fertig i *Schütz* su u radu [3] opisali uloge koje delegati mogu da imaju u okviru sistema. Te uloge podrazumevaju:

- **Čvor konsenzusa** – odabrani od strane čitave mreže sa ulogom da obezbede da svaki čvor poštuje trenutno stanje sistema,
- **Čvor govornik** – vrši kreiranje novog bloka,
- **Čvorovi delegati** – vrši validiranje bloka, odnosno transakcija u okviru njega.

Kineska kompanija *Neo*, blokčejn platforma za razvoj digitalne imovine i integraciju pametnih ugovora, koristi ovaj pristup. Zanimljiva je izjava kompanije da će u budućnosti moći da izvrši više od 10.000 transakcija u sekundi. Ovo približava blockčejn brzini transakcija tradicionalnih baza podataka [9].

Iako još uvek nije toliko rasprostranjen, ovaj pristup pruža alternativu sa jednostavnijom demonstracijom upotrebe metoda rada.

6 Dokaz o proteklom vremenu [4]

Proizvođač čipova *Intel* razvio je ovaj dokaz 2016. godine kako bi smanjio potrošnju energije i drugih resursa koje je crpeo tada aktuelni algoritam *dokaza o obavljenom poslu*. Mehanizam konsenzusa *dokaza o proteklom vremenu* čini proces posebno pogodnim za blokčejn mreže sa dozvolom pristupa. U mrežama koje koriste ovakav pristup, učesnik mora da se identifikuje unapred pre nego što se može pridružiti, što ih čini hibridnim mrežama privatnih i javnih mreža.

Algoritam funkcioniše na sledeći način: Svaki čvor koji učestvuje u mreži mora sačekati nasumično odabran vremenski period, a prvi ko ispuni zadato čekanje dobija pravo da kreira novi blok. Svaki čvor u blokčejn mreži generiše nasumično vreme čekanja i prelazi u režim pripravnosti na određeno vreme. Onaj koji se „*probudi*" prvi – to jest onaj sa najkraćim vremenom čekanja – kreira novi blok na blokčejnu slanjem potrebnih informacija celoj p2p mreži. Isti proces se zatim ponavlja za kreiranje sledećeg bloka.

Zasnovan na principu lutrijskog sistema, u kojem svaki pojedinačni čvor ima jednake šanse da bude pobednik, mehanizam *dokaza o proteklom vremenu* se zasniva na distribuciji šansi za pobedu na što veći broj učesnika mreže. Prvi učesnik koji završi čekanje postaje rudar novog bloka.

Da bi ovo funkcionisalo, moraju se proveriti dva zahteva.

1. Da li je pobednik lutrije dobio zapravo nasumično vreme čekanja? U suprotnom, učesnik bi mogao namerno da izabere kratko vreme čekanja za pobedu.
2. Da li je dobitnik lutrije zaista čekao određeno vreme?

S obzirom da ovaj algoritam zahteva dozvolu pristupa, mreži je moguće pristupi samo sa određenim sertifikatom, čime se ne promoviše otvorenost kao kod javnih blokčejnova. Ugrađeni mehanizam omogućava aplikacijama da pokreću pouzdani kod u zaštićenom okruženju. Ovim se sprečavaju nelegalne promene koda koje unose malicioznost u sistem.

Hyperledger Sawtooth je platforma za kreiranje blokčejnova koji se koriste u preduzećima, a koja olakšava kreiranje aplikacija i mreža koji koriste distribuiranu knjigu. Ova platforma upravo koristi algoritam *dokaza o proteklom vremenu* za ostvarivanje konsenzusa [10].

8 HotStuff [11]

HotStuff je *BFT* (*eng. Byzantine Fault Tolerance*) protokol za replikaciju automata (*eng. state machine*). Nekoliko inovacija čini ovaj algoritam boljim od tradicionalnog *PBFT* (*eng. Practical Byzantine Fault Tolerance*) protokola. Međutim, kao i *PBFT*, radi u mreži sa delimično sinhronom razmenom poruka i minimalnim brojem ispravnih čvorova $n = 3f + 1$, gde je n broj ukupnih, a f broj neispravnih čvorova u mreži. Replikacija se oslanja na postojanje jednog lidera koji je odgovoran za koordinaciju razmene poruka između replika (*eng. leader based primary backup*)[1]. Koristi pouzdane i autentifikovane kanale za komunikaciju. Takođe, *HotStuff* koristi granične potpise (*eng. threshold signature schema - TSS*) gde svi čvorovi koriste jedan javni ključ, ali svaka replika koristi jedinstveni privatni ključ za digitalni potpis. Upotreba graničnih potpisa rezultira smanjenjem složenosti komunikacije. *HotStuff* je svetu konsenzusa predstavio neke inovacije o kojima će biti više reči u nastavku.

Sertifikat kvoruma (*eng. quorum certificate*) – struktura podataka koja predstavlja kolekciju kriptografskih potpisa proizvedenih od $n - f$ čvorova koji označavaju da je potreban prag potpisa (*eng. signature threshold*) postignut. Drugim rečima, kolekcija glasova iz $n - f$ čvorova koja potvrđuje da je kvorum saglasan sa prosleđenim podacima.

Linearna promena stanja (*eng. Linear view change*) – promena stanja u *HotStuff* protokolu zahteva samo $O(n)$ poruka. To je deo normalnog rada sistema, umesto posebnog potprotokola. U najgorem slučaju kada lideri otkažu jedan za drugim, cena komunikacije poraste na $O(n^2)$ – kvadratna složenost. Za razliku od *PBFT*-a, gde je uloga lidera dodeljena jednom čvoru, ovde uloga lidera rotira između čvorova na svake tri runde, ukoliko se ne desi prevremeni otkaz trenutnog lidera. U linearnoj promeni stanja, nakon vremena za stabilizaciju sistema (*eng. Global Stabilization Time – GST*), svaki ispravan izabrani vođa šalje samo $O(n)$ poruka za potpisivanje i distribuciju novog stanja, što znači da u najgorem slučaju, kada lideri krenu da otkazuju jedan za drugim, kompleksnost komunikacije dostiže kulminaciju sa $O(n^2)$. Detaljniji izgled ovih poruka odnosno samog toka komunikacije će biti objašnjen kasnije.

Optimistični odziv (*eng. Optimistic Responsiveness*) – omogućava svakom ispravnom lideru nakon *GST*-a da sačeka potpise od samo prvih $n - f$ čvorova, kako bi se obezbedio napredak, umesto da se čeka na po $n - f$ potpisa svake od replika. Ovo znači da radi brzinom mreže umesto da za prelazak u drugu fazu nepotrebno čeka na još poruka iz drugih čvorova.

Kvalitet lanca (*eng. Chain quality*) – svojstvo koje omogućava pravičnost i životnost (*eng. fairness and liveness*)[2] u sistemu kroz ciklično smenjivanje lidera.

Sakrivena brava (*eng. Hidden lock*) – problem koji se javlja kada lider tokom validiranja ne sačeka vreme isteka runde, već prevremeno donese odluku, primivši adekvatan broj poruka ($n - f$). Najbolja brava (*eng. highest value lock*) u tom slučaju možda ne stigne do lidera, što rezultira situacijom u kojoj lider nije svestan brave sa najvećim brojem potpisa. Ako lider tada predlaže nižu vrednost dok neki

[1] Takođe poznata i kao primarno-sekundarna replika (*eng. primary-secundary replica*).
[2] Životnost (*eng. liveness*) predstavlja prenos trenutnih podataka, odnosno razlikuje aktuelne podatke od ponovnog slanja prethodno generisanih podataka.

drugi čvorovi već imaju bolju vrednost brave, to može dovesti do problema sa životnošću (*eng. liveness issue*). Čvorovi će uvek čekati bravu koja ima veću ili istu vrednost koju oni poseduju, ali vođa nije svestan najveće vrednosti brave i nastaviće da šalje nižu vrednost, što dovodi do kršenja uslova trke (*eng. race condition*) i životnosti. *HotStuff* je rešio ovaj problem dodavanjem runde zaključavanja prekursora (*eng. precursor lock*) pre runde zaključavanja brave. Ideja iza ovoga je da lider primi odgovore od $2f + 1$ čvora, koji prihvataju zaključavanje prekursora. Na taj način, lider će dobiti odgovor od njih i naučiti najvišu zaključanu vrednost. Iz tog razloga, lider nije u obavezi da čeka Δ vremena (delta – gornja granica kašnjenja isporučivanja poruke) i može naučiti najbolju vrednost brave sa $n - f$ odgovora.

Pejsmejker (*eng. pacemaker*) – *HotStuff* na inovativan način odvaja mehanizme sigurnosti i životnosti. Sigurnost je obezbeđenja kroz pravila glasanja i potvrđivanja za učesnike u mreži. S druge strane, životnost je odgovornost posebnog modula, koji se zove pejsmejker (*eng. pacemaker*). Ovaj modul obezbeđuje da je novo biranje vođe pravično i jedinstveno. Štaviše, pejsmejker garantuje napredak nakon dostizanja *GST*-a. Prva odgovornost koju ima je da dovede sve ispravne replike, kao i jedinstvenog vođu na isti nivo za dovoljno dug period. Za sinhronizaciju, replike postepeno povećavaju vreme čekanja dok se ne postigne napredak. Kako bi ovaj mehanizam funkcionisao, neophodan je sinhroni model razmene poruka. Takođe, sam proces izbora lidera se zasniva na jednostavnoj rotirajućoj paradigmi, gde je određeni raspored, obično *round-robin*, praćen od strane svih replika tokom odabira novog lidera. Pejsmejker takođe obezbeđuje da predlog vođe bude prihvaćen od strane replika.

Učešće replika i topologija mreže – *HotStuff* organizuje čvorove u topologiju zvezde. Ova postavka omogućava vođi da šalje ili prikuplja poruke direktno u ili sa svih drugih čvorova, što rezultuje smanjenu složenost razmene poruka. Jednostavnije rečeno, koristi se obrazac komunikacije „jedan-ka-svima“ (*eng. one-to-all*). Ako je lider odgovoran za svu ovu obradu, lako može nastati problem velikog opterećenja jedne tačke, što može usporiti mrežu. Problem može biti još veći, ukoliko se radi o neispravnom lideru. Međutim, ukoliko vođa predloži nevalidan blok, biće odbijen od strane drugih ispravnih čvorova, koji neće izvršiti njegovo potpisivanje, te vođa neće skupiti dovoljno potpisa da bi upisao blok. Ukoliko se nastavi sa predlaganjem nevalidnog bloka i njegovo neuspešno upisivanje, tajmer dodeljen vođi će isteći i biće zamenjen sledećom replikom po rasporedu. Ako je većina mreže iskrena, ispravan vođa će na kraju preuzeti i predložiti korektan blok. Takođe, za dodatnu sigurnost, vođa se obično često rotira između replika svakih nekoliko rundi, što može nadoknaditi bilo koji zlonamerni napadi usmereni na vođu. Ovo svojstvo obezbeđuje pravičnost, što pomaže da se postigne bolji kvalitet lanca.

HotStuff etape

Faza pripreme (*eng. prepare*)
Kada novi vođa akumulira $n - f$ potpisa od čvorova za promenu stanja, protokol počinje sa novim vođom. Vođa obrađuje ove poruke da bi odredio najnoviju granu u kojoj je prisutan sertifikat najvećeg kvoruma sa porukom PRIPREMA.

Faza pre-potvrde (*eng. pre-commit*)
Čim lider akumulira $n - f$ pripremnih glasova, on stvara sertifikat kvoruma pod nazivom „*sertifikat kvoruma pripreme*“. Vođa emituje ovaj sertifikat drugima čvorovi kao poruku PRE–POTVRDE. Kada

čvor primi poruku PRE–POTVRDE, odgovara glasanjem PRE–POTVRDE. Potvrda o kvorumu ukazuje da je neophodan prag čvorova potvrdio zahtev za daljom obradom.

Faza potvrde (*eng. commit*)
Kada vođa sakupi $n - f$ glasova pre potvrđivanja, on stvara sertifikat PRE-POTVRDE kvoruma i emituje ga drugim čvorovima kao poruku POTVRDE. Kada čvorovi primaju ovu poruku POTVRDE, oni odgovaraju svojim glasanjem POTVRDE. U ovoj fazi, čvorovi zaključavaju sertifikat kvoruma PRE-POTVRDE da bi se osigurala bezbednost algoritma čak i ako dođe do promene stanja.

Faza odlučivanja (*eng. decision*)
Kada lider dobije $n - f$ glasova POTVRDE, on kreira kvorum sertifikat POTVRDE. Zatim, ga emituje drugim čvorovima u ODLUČITE poruku. Kada čvorovi prime poruku ODLUČITE, oni izvršavaju zahtev jer ova poruka sadrži već urezani sertifikat da su se svi složili. Novo stanje počinje kada dođe do prelaza stanja zbog prihvatanja poruke ODLUČITE i izvršavanjem zahteva.

Ovaj konsenzus protokol je među jednostavnijima i uveo je dosta inovacija koje nadomestuju nedostatke njegovih prethodnika. Trenutna mreža koja koristi ovaj konsenzus je *Cypherium*.

9 Raft [12]

RAFT je dizajniran kao odgovor na nedostatke koji se javljaju u Lamportovom notornom algoritmu Paksos (*eng. Paxos*) [13]. RAFT je engleska skraćenica od *Replicated And Fault Tolerant* (Repliciran i otporan na greške). Autori RAFT-a su imali glavni cilj da razviju protokol koji je lak za razumeti i implementirati. Ključna ideja iza RAFT-a je da omogući replikaciju stanja automata (*eng. state machine replication*) sa perzistentnim dnevnikom evidencije događaja (*eng. persistent log*). Stanje automata se utvrđuje na osnovu dnevnika podataka evidencije događaja. RAFT omogućava rekonfiguraciju klastera što znači da promena članova sistema ne ometa rad servisa. Štaviše, s obzirom da podaci dnevnika mogu znatno narasti, kada se radi o sistemima sa velikim propustom (*eng. high throughput*), RAFT omogućava kompresiju podataka dnevnika kako bi se olakšao problem skladištenja ogromne količine podataka, kao i sporo pokretanje nakon što čvor otkaže.

RAFT radi nad modelom sistema koji ima sledeće pretpostavke:
- Nema grešaka vizantijskog tipa.
- Nepouzdana mrežna komunikacija.
- Asinhrona komunikacija i procesori.
- Deterministički automat na svakom čvoru koji počinje u istom početnom stanju.
- Čvorovi imaju trajna skladišta podataka koja podržavaju *WAL (Write-ahead logging)*, što podrazumeva upisivanje podataka na kraj skladišta (*eng. append*) u *ACID* sistemima, što znači da će se svaki upis u skladište odigrati pre otkaza čvora.
- Klijent mora striktno da komunicira samo sa aktuelnim liderom. Ovo je odgovornost klijenta, s obzirom da klijenti prepoznaju sve čvore što je omogućeno inicijalnom konfiguracijom.

RAFT je (asimetrični) protokol zasnovan na lideru, gde se jedan čvor bira kao lider odnosno vođa klastera. Ovaj lider prihvata zahteve klijenata i upravlja replikacijom podataka dnevnika. U RAFT klasteru može biti samo jedan vođa, ukoliko se desi njegov pad, onda se vrši odabir novog lidera.

Postoje tri uloge u RAFT klasteru koje čvorovi mogu imati:
- ***Lider*** – prima zahteve klijenata, upravlja replikacijom podataka dnevnika i upravlja komunikacijom sa sledbenicima.
- ***Čvor sledbenik*** – pasivan po prirodi i odgovara samo na pozive udaljene procedure (*eng. Remote Procedure Call – RPC*). Nikada ne pokreće nikakvu komunikaciju.
- ***Kandidat*** – uloga koju koristi čvor koji pokušava da postane lider pokrećući proces glasanja.

Vreme u RAFT-u je logično podeljeno na mandate (*eng. term*). Mandat ili epoha je u osnovi monotono rastuća vrednost koja ima ulogu logičkog časovnika za postizanje delimičnog uređenja događaja u odsustvu globalnog časovnika. Svaki mandat počinje izborom novog lidera, gde se jedan ili više kandidata takmiče da steknu ulogu vođe klastera. Kada je lider izabran, on služi kao vođa do kraja mandata. Ključna uloga mandata je da identifikuje zastarele informacije, na primer, ustajale lidere. Svaki čvor skladišti vrednost tekućeg mandata. Kada se trenutni mandati razmenjuju između čvorova, proverava se da li je trenutna vrednost mandata jednog čvora manja od vrednosti mandata drugog čvora; ukoliko jeste, tada čvor sa manjom vrednošću ažurira svoju vrednost na veću. Kada kandidat ili lider sazna da je njegov trenutni mandat zastareo (vrednost mandata je manja od trenutne vrednosti epohe), on prelazi u režim sledbenika. Svi zahtevi sa zastarelom vrednošću mandata se odbijaju.

RAFT protokol radi koristeći dva unapred definisana poziva udaljene procedure:

- *NadoveziUnos(…) (eng. AppendEntries)* – RPC koji se poziva od strane lidera za duplikaciju podataka evidencije, a takođe se koristi kao signal otkucaja srca (*eng. Heartbeat signal*),
- *ZahtevajGlasanje(…) (eng. RequestVote)* – RPC koji pozivaju čvorovi kandidati kako bi sakupili glasove.

RAFT etape

RAFT se sastoji od dve faze. Prva faza je faza odabira lidera, a druga je duplikacija podataka evidencije. U drugoj fazi, vođa prihvata zahteve klijenata, ažurira dnevnik podataka evidencije i šalje signale otkucaja srca kako bi sve sledbenike obavestio o njegovom postojanju, odnosno da se nije srušio.

Izbor lidera

Mehanizam otkucaja srca se koristi kao okidač za proces izbora lidera. Svi čvorovi se pokreću kao sledbenici. Proces ulogu sledbenika izvršava sve dok prima važeće RPC-ove od vođe ili procesa kandidata. Ako sledbenik ne dobije signal otkucaja srca od vođe za određen vremenski period, onda dolazi do isteka tajmera koji okida takozvani „*tajm-aut izbora*“. Tajm-aut vremensko ograničenje je nasumično podešeno između 150 i 300 ms.

Nakon isteka tajmera, čvor sledbenik preuzima ulogu kandidata i pokušava da postane lider pokretanjem izbornog protokola. Kandidat povećava svoju vrednost mandata, glasa za sebe, resetuje izborni tajmer i traži glasove od drugih sledbenika putem *RequestVote() RPC*-a. Ako dobije glasove većine čvorova, onda postaje vođa i počinje da šalje signale otkucaja srca drugim čvorovima, koji su sada sledbenici. Ako je drugi kandidat pobedio i postao validan lider, onda bi ovaj kandidat počeo da dobija signale otkucaja srca i vratiće se u ulogu sledbenika. Ako niko ne pobedi na izborima i izborima nastupi tajmaut, izborni proces počinje ponovo sa novim mandatom.

Bitno je naglasiti da kada proces sledbenik dobije *RequestVote() RPC,* on upoređuje podatke svog dnevnika, sa podacima dnevnika kandidata i samo ukoliko je ta vrednost manja ili jednaka primljenoj vrednosti procesa kandidata, moći će da glasa za njega, u protivnom, odbija zahtev. Takođe, zahtev je odbijen i ukoliko je vrednost mandata (epohe) procesa kandidata manja od trenutne vrednosti mandata koju poseduje sledbenik.

Replikacija dnevnika

Faza replikacije dnevnika RAFT-a je jednostavna. Prvo, klijent šalje komande/zahtev lideru da ih izvrši nad repliciranim automatom. Lider zatim komandi dodeljuje vrednost mandata i indeks tako da komanda može biti jedinstveno identifikovana u dnevnicima koje poseduju čvorovi.

Zatim ovu komandu nadovezuje svom dnevniku. Kada vođa ima novi unos u svom dnevniku, istovremeno šalje zahteve za repliciranjem ovog unosa preko *AppendEntries() RPC*-a svim čvorovima sledbenicima.

Tek kada je vođa u stanju da sprovede komandu nad većinom čvorova sledbenika, to jest, kada je većina čvorova potvrdila komandu, komanda se smatra učinjenom nad klasterom. Sada lider izvršava komandu nad svojim automatom i vraća rezultat klijentu. Takođe, lider obaveštava i sledbenike da je unos izvršen preko *AppendEntries() RPC*-a, i sledbenici izvršavaju prosleđenu komandu nad svojim automatima.

Ukoliko iz nekog razloga *AppendEntries() RPC* ne uspe nad manjinom sledbenika u sistemu, lider će pokušavati nad njima da izvrši komande ukoliko je većina već potvrdila novo stanje. Ukoliko ih ne obave, nije problem, jer većina diktira sistemom i bitno je da oni imaju konzistentnu repliku stanja nad kojom su se prethodno dogovorili, odnosno za istu glasali (digitalnim potpisima).

Kada sledbenik primi *AppendEntries() RPC* za replikaciju podataka dnevnika, prvo proverava vrednost mandata poziva, ukoliko je manja od trenutne, odbija poziv i vraća *false*. Takođe će odbiti poziv ukoliko vrednost indeksa prosleđene komande ne postoji ili nije sukcesivna od one koju sledbenik poseduje, a vrednost mandata je korektna. Tako da se vrši isključivo nadovezivanje samo onih komandi čiji mandat odgovara trenutnom, a indeksi su sukcesivni. Konačno, za slučaj da se kroz RPC primi komanda čiji se indeks nalazi u dnevniku sledbenika, ali vrednosti mandata razlikuju, sve komande od tog indeksa bivaju poništene, odnosno prepisane primljenom komandom.

Ukoliko postoji kandidat ili sledbenik proces koji se srušio, protokol će pokušati da ih učini konzistentnim ponovnim okidanjem *AppendEntries() RPC*-a.

Onda kada je komanda potvrđena u klasteru, RAFT obezbeđuje da neće biti izgubljena, bez obzira na greške u mrežnoj komunikaciji, ponovnom pokretanju sistema, ili otkazima. Međutim, greške vizantijsog, odnosno arbitarnog tipa nisu podržane.

Svaki unos u dnevnik podataka evidencije sadrži: vrednost mandata nad kojim je izdata, indeks komande, kao i samu komandu nad automatom. Vrednošću mandata se utvrđuje nekonzistentnost podataka, daje se neki vid vremenske markice u kojoj je izvršena komanda. Indeks identifikuje poziciju unosa u dnevnik. Komanda je očigledno zahtev koji je klijent pozvao na izvršenje.

Garancije i ispravnost

Garancije koje RAFT omogućava:

- **Korektnost izbora**
    - **Sigurnost izbora** – u svakom mandatu može biti izabran najviše jedan lider.
    - **Izborna živost (*eng. liveness*)** – neki kandidat će kad-tad postati lideri.
- **Lider može samo da pridodaje** – lider može samo da vrši upisivanje novih podataka u dnevnik. Nema dozvolu prepisivanja ili brisanja unosa iz dnevnika. Sledbenicima je dozvoljeno prepisivanje lokalnih replika samo ukoliko se identifikuje nekonzistentnost.
- **Podudaranje podataka evidencije** – ako dva unosa iz dva dnevnika, svaki na svom čvoru, imaju isti indeks i vrednost mandata, onda su ovi dnevnici identični u svim prethodnim unosima i čuvaju istu komandu.
- **Celovitost lidera** – unos u dnevnik potvrđen u određenom mandatu će uvek biti prisutan u dnevniku budućih lidera, odnosno lidera sa većom vrednošću mandata. Čvorovi sa nepotpunim dnevnicima nikada neće biti odabrani za lidera.

- **Bezbednost automata** – ako je čvor primenio unos u dnevnik za dati indeks na svom automatu, nijedan drugi čvor nikada neće primeniti drugačiji unos u dnevnik za isti indeks.

Izborna korektnost zahteva sigurnost i život. Sigurnost znači da je najviše jedan lider dozvoljen po mandatu, dok živost zahteva da neki kandidat mora da pobedi i postane vođa na kraju. Da bi se osigurala sigurnost, svaki čvor glasa samo jednom u mandatu i to perzistira na skladištu. Za pobedu na izborima potrebna je većina, odnosno ne postoje dva različita kandidata koji će istovremeno dobiti većinu.

Do podeljenih glasova može doći tokom izbora lidera. Ako dva čvora budu izabrana istovremeno, tada može doći do takozvanog „*podela glasa*“ (*eng. split vote*). RAFT koristi tajmer sa nasumičnom vrednošću za rešavanje ovakvih situacija. Ovo pomaže jer nasumična vremenska ograničenja dozvoljavaju samo jednom čvoru da kada tajmer istekne, može da se probudi i pobedi na izborima pre isteka drugog čvora. U praksi, ovo dobro funkcioniše ako je nasumično odabrano vreme veće od vremena latencije na mreži.

Podudaranje podataka dnevnika postiže visok nivo konzistentnosti između dnevnika. Pretpostavljamo da vođa nije maliciozan. Lider nikada neće dodati više od jednog unosa sa istim indeksom u istom terminu. Provere konzistentnosti dnevnika obezbeđuju da su svi prethodni unosi identični. Lider vodi evidenciju o najnovijem indeksu koji je izvršio u svom dnevniku. Lider emituje ove informacije u svakom *AppendEntries() RPC*-u. Ako čvor sledbenik nema unos u svom dnevniku sa istim indeksnim brojem, neće prihvatiti dolazni unos. Međutim, ako sledbenik prihvati *AppendEntries() RPC,* lider zna da su dnevnici identični na oba kraja. Dnevnici su generalno konzistentni osim ako nema grešaka na mreži. U tom slučaju, provera konzistentnosti dnevnika obezbeđuje da čvorovi na kraju sustignu podatke sistema i postanu konzistentni. Ako evidencija nije konzistentna, vođa će ponovo preneti nedostajuće unose sledbenicima koji možda nisu ranije primili poruku ili su se srušili, a sada su se oporavili.

Rekonfiguracija i kompresija podataka dnevnika su dve korisne karakteristike RAFT-a. Nisam raspravljao o njima jer nisu direktno povezani sa osnovnim konsenzusnim protokolom. O njihovim detaljima se može pročitati u okviru RAFT reference [12] izdvojene na kraju rada.

Kao što je rečeno na početku, RAFT služi kao prokol za očuvanje stanja replika, ne specifično kao alat u blokčejn tehnologijama. Trenutna produkciona korist ovog protokola jeste u očuvanju replika baza podataka poput: *CouchDB, MongoDB, YugabyteDB, TiDB*…

10 Pregled

algoritam	PoW	PoS	PoET	dBFT	HotStuff	PoBurn	PoA	PoBelievability
tip blokčejna	javni	privatni i javni	privatni i javni	privatni	privatni	javni	privatni i javni	privatni
ostvarivanje transakcije	lutrija	lutrija	lutrija	kvorum	kvorum	lutrija	lutrija	lutrija
brizina transakcije	niska	niska	srednja	visoka	visoka	niska	visoka	visoka
konzumacija resursa	visoka	srednja	visoka	srednja	niska	srednja	niska	niska
skalabilnost peer mreže	visoka	visoka	visoka	srednja	srednja	visoka	srednja	visoka
otpornost na greške	<51% moći obrade	<51% udela	nepoznato	<33% neispravnih replika	<33% neispravnih replika	<51% udela	nepoznato	nepoznato
implementacija	Bitcoin	Ethereum, NAVcoin, Neo, Lisk…	Hyperledger Sawtooth	Hyperledger Fabric, Neo	Cypherium	Slimcoin	VeChain, Xodex, Microsoft Azure	IOST

11 Zaključak

Kako se internet danas široko koristi, veliki broj digitalnih sadržaja se deli na mreži. Ali, dok se sadržaj deli, dosta toga se i krade. Postoji nekoliko načina preuzimanja vlasništva nad digitalnim sadržajem, ali ti načini uključuju intervenciju treće strane kao što su vlade, banke, itd. Postoje i druge tehnologije koje omogućavaju određen nivo bezbednosti digitalnog sadržaja, ali imaju svoje nedostatke, pa stoga nisu toliko pouzdane. Decentralizovan način može se čak koristiti kako bi se eliminisalo veliko vreme potrebno za dobijanje vlasništva i obezbeđenje sadržaja. Blokčejn tehnologija može biti od velike pomoći u postizanju ovoga zbog njenih karakteristika kao što su bolja transparentnost, smanjeni troškovi, nepromenljivost, vreme, evidentiranje, itd [14].

Postoje različiti konsenzus algoritmi koji se mogu koristiti za ovo, kao što su *dokaz o zakupu, Vizantijska tolerancija grešaka, dokaz o obavljenom radu*, itd. Ovo su najčešće korišćeni konsenzusni algoritmi. Algoritmi kao što je dokaz o učinku (*eng. Proof of Contribution*) su takođe uvedeni da bi se obezbedila zaštita digitalnog sadržaja [15]. *Ethereum* je jedna od najzrelijih blokčejn platformi, koja koristi algoritam *dokaza o zakupu* za postizanje konsenzusa, a koristi se baš zato što zahteva manju potrošnju resursa, dok je pritom relativno brži od algoritma *dokaza o obavljenom radu*, a takođe i efikasniji. Stoga je to dobra alternativa za obezbeđivanje digitalnog sadržaja na decentralizovan način [16]. Takođe, *Ethereum* je podržan moćnim programskim jezikom, tj. *Solidity*, koji može da se nosi sa mnogim složenim proračunima, što ga čini najboljim izborom za kreiranje aplikacija o vlasništvu nad digitalnim sadržajem.

Veliki broj različitih modela konsenzusa daje prvi utisak o bogatstvu opcija za dizajn. Dokaz o radu funkcioniše tako što svi čvorovi koji učestvuju rešavaju kripto-grafičke slagalice. Ovo zahteva mnogo računarske snage za validaciju blokova. Dodatno, košta mnogo energije. Bez odgovarajućeg i skupog hardvera, gotovo je nemoguće potvrditi blok. Prednost ovog pristupa je pre svega zaštita od hakerskih napada: Uspešno izveden napad mora da se odvija istovremeno na najmanje više od 50% svih čvorova koji učestvuju. Velika mreža poput *Bitcoin*-a sa više od 9400 čvorova, takav napad čini nemogućim. Međutim, za mreže mnogo manjih razmera, ovaj algoritam nije preporučljiv. Svaki od kasnijih pristupa koji su se razvijali su pokušavali da reše poznate probleme prethodnika kreirajući nove probleme za rešavanje svojim sledbenicima. Dakle, svaki od njih ima svoje prednosti i mane. Na projektantu je pre svega da odabere tip blokčejn mreže, na osnovu potreba, a dalje da isplanira koji od ovih, ili možda čak i više njih, u zavisnosti od etape razvoja mreže, može najefikasnije da iskoristi.

12 Reference

[1] S. Lakshmi, M. Siva, M. Sindhu i M. Sethumadhavan, „Survey of consensus protocols on blockchain applications,“ 2017.

[2] S. Kaur, S. Chaturvedi, A. Sharma i J. Kar, „A Research Survey on Applications of Consensus Protocols in Blockchain,“ Hindawi, Jaipur, 2021.

[3] A. Schutz i o. Fertig, Blockchain für Entwickler: Das Handbuch für Software Engineers. Grundlagen, Programmierung, Anwendung. Mit vielen Praxisbeispielen, Berlin: Rheinwerk Computing, 2019.

[4] K. Adam, Blockchain Technology for Business Processes, Berlin: Springer, 2022.

[5] S. D. Angelis, L. Aniello, R. Naldoni, F. Lombardi, A. Marheri i V. Seassone, „PBFT vs Proof-of-Authority: Applying the CAP Theorem to Permissioned Blockchain,“ University of Southampton, Rome, 2017.

[6] L. Lamport, R. Shostak i M. Pease, „The Byzantine Generals Problem,“ SRI International, 1982.

[7] M. Holler, S. Napel i G. Illing, Einführung in die Spieltheorie, Berlin: Springer, 2019.

[8] E. A. Akkoyunlu, K. Ekanadham i R. V. Hubert, „Some costraints and tradeoffs in the design of network communicaions,“ Stony Brook, State University, New York, 1975.

[9] neo, „Neo White Paper,“ 2022. [Na mreži]. Available: https://docs.neo.org/v2/docs/en-us/basic/whitepaper.html.

[10] Hyperledger, „Sawtooth,“ 2022. [Na mreži]. Available: https://sawtooth.hyperledger.org/docs/1.2/.

[11] I. Bashir, Blockchain Consensus: An Introduction to Classical, Blockchain, and Quantum Consensus Protocols, London: Springer, 2022.

[12] D. Ongaro i J. Ousterhout, „In Search of an Understandable Consensus Algorithm,“ Stanford University, Philadelphia, 2014.

[13] L. Lamport, „Paxos Made Simple,“ 2001.

[14] G. Zyskind, O. Nathan i A. Pentland., „Decentralizing privacy: Using blockchain to protect personal data,“ IEEE, 2015.

[15] H. Song, N. Zhu, R. Xue, J. He, K. Zhang i J. Wang, „Proof-of-contribution consensus mechanism for blockchain and its application in intellectual property protection,“ Information Processing & Management, 2021.

[16] J. Wang, S. Wang, J. Guo, Y. Du, S. Cheng i X. Li, „A summary of research on blockchain in the field of intellectual property,“ Procedia Computer Science, 2019.

[17] V. Bhatnagar, V. Bali, L. Malik, S. Arora, U. Shrawankar i D. Vivek, Blockchain for Smart and Green Society: Promise, Practice and Application, Oxon: CRC PRess, 2022.

[18] J. Bonneau i N. Heninger, Financial Cryptography and Data Security, New York: Springer, 2020.

[19] I. Bentov, C. Lee, A. Mizrahi i M. Rosenfeld, „Proof of Activity: Extending Bitcoin’s Proof of Work via Proof of Stake,“ 2014. [Na mreži]. Available: https://eprint.iacr.org/2014/452.pdf.

[20] V. Madisetti i A. Bahga, Blockchain Applications: A Hands-On Approach, Georgia: VPT, 2017.