Predlog arhitekture za Distribuiranu igru Haosa

Milan Bojic, Andrej Gasic, Stefan Budimac

Jun 2022

Sadrzaj

1 Arhitektura

U osnovnoj verziji sistema moze se jasno izdvojiti tri razlicita stanja sistema:

- IDLE stanje
- Stanje izrade fraktala
- Stanje izrade visestrukih fraktala

1.1 IDLE stanje

Tokom izgradnje sistema tokom IDLE stanja sistem se formatira kao uvezana ciklicna lista. Cvor u sistemu je povezan sa svojim prethodnikom i svojim sledbenikom (cvor n je povezan sa cvorovima n-1 i n+1) Ove veze ce ostati i pri prelazu u drugo stanje, odnosno samo zavisi stabilnosti sistema.

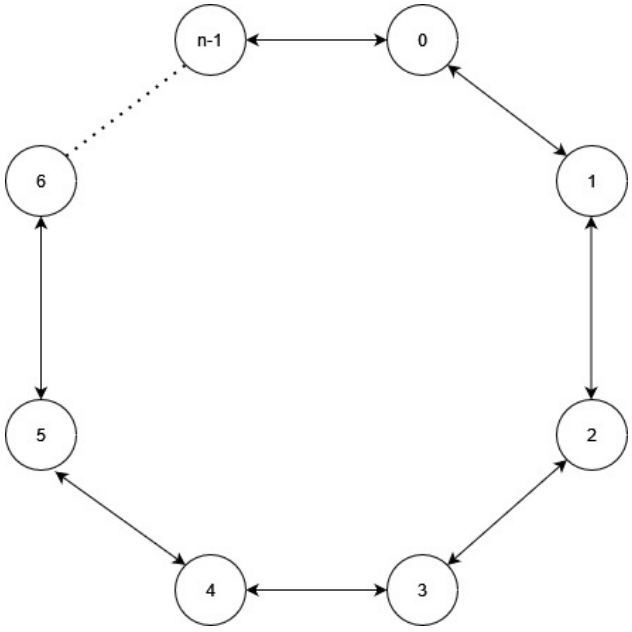


Figure 1: Diagram IDLE sistema, Veze u grafu su bidirekcijualne

Ovako napravljen sistema lici na Chorda samo bez 2^n veza. Ovo sluzi i kao veza izmedju klastera sa koja rade dva razlicita fraktala. Veza u klasteru se opisuje u sledecem stanju.

1.2 Stanje izrade fraktala

Za opis ovog dela koristicu primer sa 3 tacke u fraktalu.

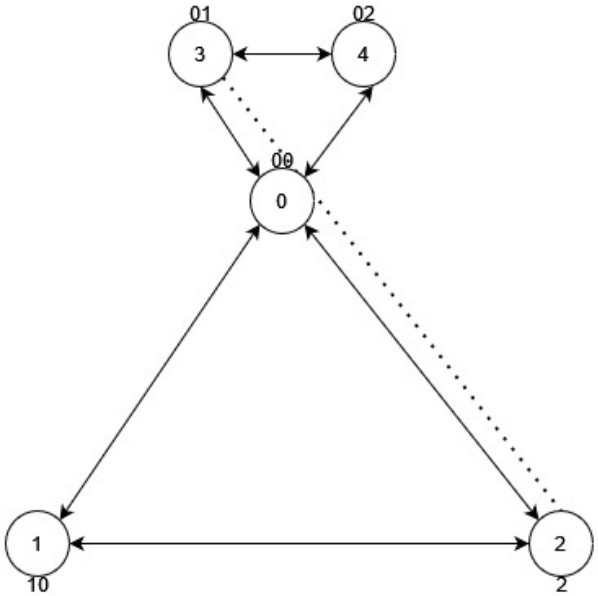


Figure 2: Diagram Klaster sistema, Veze sa strelicama nastale unutra klastera, isprekidane veze su iz IDLE stanja

Graf je blago izmenjena K-Kubni graf (eng. Hypercube graph) Dijagram klastera se moze prikazati i kao 3d graf na vise nivoa. Sto je nivo veci slika fraktala bi bila veca. Nulti nivo bi sadrzao ceo fraktal. Napomenta, stvarni cvorovi(koji su usli preko Bootstrap servisa) se nalaze na poslednjem nivou. Ostali nivoi su samo idejni, gde cvorovi imaju informacije na kom nivou se takodje nalaze, sto im govori i sa kojim cvorovima su uvezani i o njihovoj ulozi u podfraktalu. Postoje dve ideje iza algoritma povezivanja unutar klastera:

- Povezanost u podfraktalu
- Povezanost cvorova gde je razlika u fraktalID u najvise jedanoj cifri

Unutar podfraktala svi cvorovi na istom novou su povezano sto stvara kompletan podgraf K_n. Igrom slucaja ovo ce se sloziti i sa sledecim pravilom

Da bi izbegli one-point of failure unutar sistema uvodimo pravilo gde se cvorovi, na istom nivou, spajaju ako se im se fractalID razlikuje u najvise

jednoj cifri (npr. cvorovi sa 0100 i 0120 ce se spojiti, do 1100 i 1210 nece). Za primer neka u klaster, sa prethodne slike, dolazi novi cvor.

Iz ovih pravila moze sve izvesti sledece osobine unutar sistema(gledajuci sistem sa svim cvorovima na istom nivou). Ako je n broj cvorova fraktala i m nivoa u grafu:

$$\#_e(v) = m(n-1)$$

$$\#_v(G) = n^m$$

Dijametar ovakvog grafa je m (odnosno najduzi najkraci put izmedju dva cvora u grafu), sto je logaritamskom odnosu sa brojem cvorova u Grafu $m \approx \log n^m$

Put izmedju dva nepovezana cvora se odredjuje ovako: Prolazimo kroz FractalID cvorova sa leva na desno i pri svakom i-tom skoku skacemo na cvor koji se podudara sa ciljnim cvorem na i-toj poziciji.

(npr. saljemo sa 00000 ka 12345 cvoru: 00000 -> 10000 -> 12000 -> 12300 -> 12340 -> 12345)

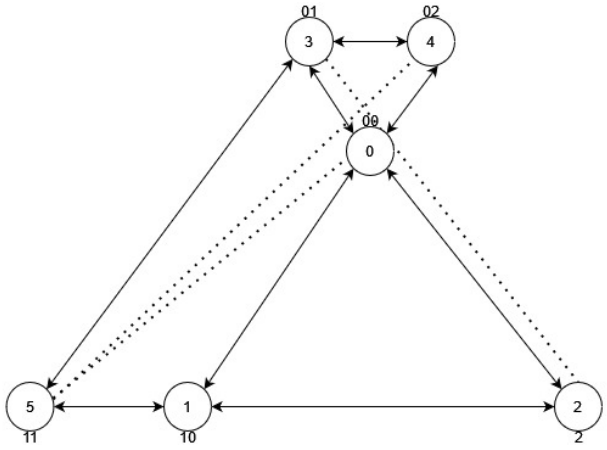
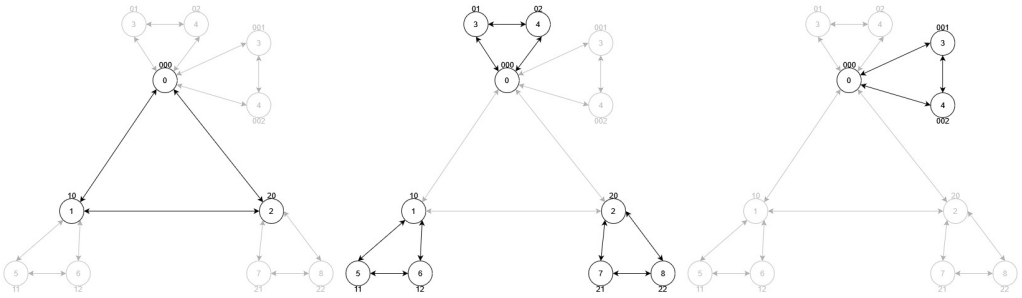


Figure 3: Dodavanje cvora u klaster, dobija fractalID 11 i povezuje sa cvorovima 3(fractalID 01) i 1(fractalID 10)



(a) Nivo 0 (b) Nivo 1 (c) Nivo 2

Figure 4: Nivoi u sistemu

1.3 Stanje izrade visestrukih fraktala

Ovde imamo dva vazna trenutka: Podela sistema po fraktalima i konekcija izmedju klastera. Gledajuci sistem kada je u IDLE stanju, ako krenemo od pocetka "lanca", za n-ti fraktal uzimamo svaki n-ti cvor da radi na zadatom fraktalu

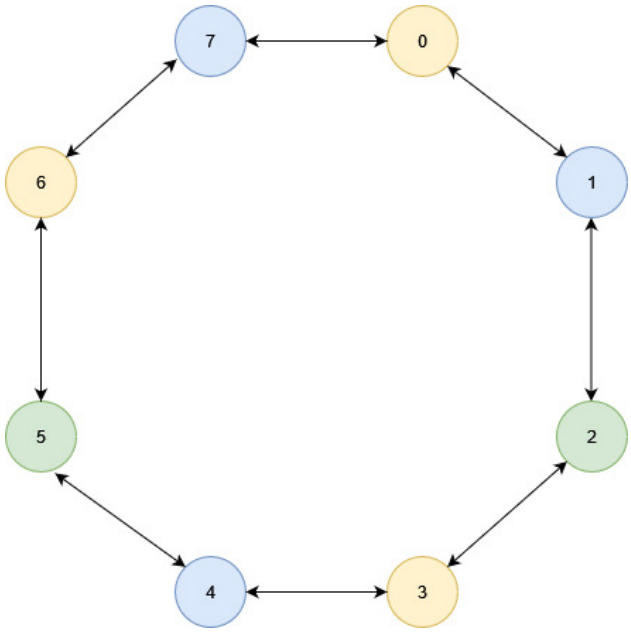


Figure 5: Nacin na koji ce se sistem podeliti na 3 fraktala. Zuti cvorovi su za Fraktal 0,Plavi cvorovi su za Fraktal 1,Zeleni cvorovi su za Fraktal 2

Kada se sistem podeli po fraktalima, nastaju klasteri cvorova. Unutra klastera koristi se pravilo **Stanje izrade fraktala**. Ovako imamo dobru povezanost u sistemu, jer ce svaki cvor (osim krajnjih) biti povezani sa cvorovima iz drugih klastera (IDLE veze).

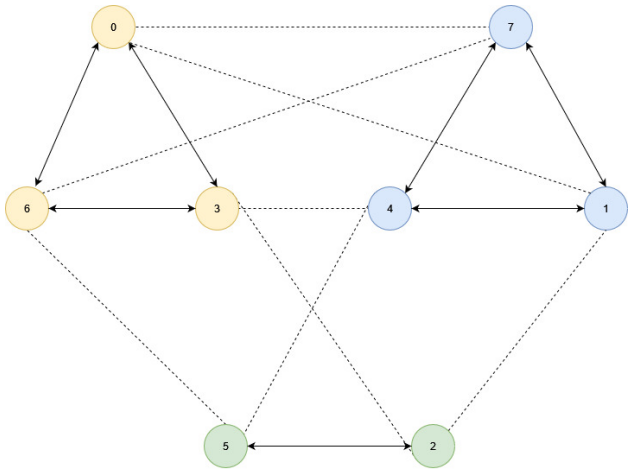


Figure 6: Izgleda sistema kada se izgradi nad tri fraktala

Da bi se poslala poruka izmedju dva cvora izmedju u dva razlicita fraktala prvo saljemo bilo kom cvoru iz fraktala gde se nalazi u fraktalu gde se nalazi ciljni cvor (to ce biti jedan dodatni skok ili dva dodatna skoka,ako nismo direktno povezani sa nekim cvorom iz tog fraktala) i nakon toga koristi se isto pravilo kao i za **Stanje izrade fraktala**.

2 Cvorovi sistema

2.1 Bootstrap

Bootstrap je cvor u sistemu koji iskljucivo sluzi za ukljuvicanje cvoraova u sistem. Ocekuje se da ce Bootstrap cvor stalno biti ukljucen, jer je neophodan da bi bilo koji cvor usao u sistem. Komunikacija sa ostatkom cvorova se svodi na ukljuvicanje cvorova u sistem i dobijanje informacija o cvoru. Informacije koje bootstrap o cvorovima u sistemu:

- ID cvora
- IP adresu
- Port

Takodje je bitno da postoji samo jedan Bootstrap cvor pokrenut u sistemu, u suprotnom sistem nece raditi.

Konfiguracija Bootstrapa

Potrebne informacije da bi se konfigurisao cvor su:

Key	Type
"port"	int
"ipAddress"	string

Koji ce pisati kao json fajl, zbog lakse prenosivosti po heterogenom sistemu

2.2 Cvor Radnik

Ovo je tip cvora ciji skup ce predstavljati sistem koji ce raditi na fraktalima. Cvor radnik ima dve glavne funkcije: Rad na fraktalima i organizacija sistema, sa ostalim cvorovima.

Konfiguracija Radnika

Potrebne informacije da bi se konfigurisao cvor su:

Key	Type
"port"	int
"ipAddress"	string
"weakLimit"	int
"strongLimit"	int
"jobs"	List of Jobs

Koji ce pisati kao json fajl, zbog lakse prenosivosti po heterogenom sistemu

Takodje Konfiguracija za **JOB** je:

Key	Type
"name"	string
"pointCount"	int
"p"	double
"width"	int
"height"	int
"mainPoints"	List of pairs of doubles

Cuvanje informacija cvorova

Unutar cvorova cuva se informacije o svim cvorovima u sistemu, a te informacije se cuvaju kao objekat oblika:

```
type NodeInfo struct {
    Id        int    `json:"Id"`
    IpAddress string `json:"ipAddress"`
    Port      int    `json:"port"`
    JobName   string `json:"JobName"`
    FractalId string `json:"FractalId"`
}
```

2.3 Komunikacija izmedju cvorova

Zbog lakse Komunikacije izmedju heterogenih cvorova, sva komunikacija ce se vrsiit preko TCP konekcija, i razmenjivace se JSON poruke.

2.3.1 Osnovna poruka

Ove ce biti superklasa svih ostalih poruka, i cilj je samo da se samo sadrzaj poruke menja, zarad bolje kompaktabilnosti komunikacije izmedju cvorova. Takodje, jedan od stvari koje vaze za svaku poruku jeste da ce imati id koji je lokalni za svaki cvor.

Key	Type	Explanation
"id"	int	Id poruke, vezana za posiljalaca
"massageType"	string	Tip poruke koji se salje
"sender"	NodeInfo	Originalni posiljalac
"reciver"	NodeInfo	Primalac rute
"route"	List of ints	Ruta kojom je poruka prosla
"massage"	string	Sadrzaj poruke

Sender i reciver su objekti koji sadrzi potrebne informacije o cvorovima. Ta polja su: Id,IpAddress,Port.

Predlog za Java implementaciju: Svaka vrsta poruka da ima svoju klase, a Osnovna poruka da bude abstraktna klasa. Da postoji metoda toJson koji pretvara objekat poruke u json string, za slanje.

2.3.2 Info poruka

Osnovno "prosirenje" osnovne poruke, ovde sama poruka je obican string koji govori neku ne struktuiranu informaciju primaocu. Trenutno se ne preporucuje za neko veliko koriscenje.

Posljalac: Cvor
Primalac: Cvor
Okidac: Nema
Akcija primanja: Ispisivanje poruke u log

2.3.3 InfoBroadcast poruka

Slicno kao i prethodno s tom razlikom da ne postoji jedan primalac vec se poruka salje celom sistemu

Posljalac: Cvor
Primalac: Nema
Okidac: Nema
Akcija primanja: Ispisivanje poruke u log

2.4 Poruke za Bootstrap

2.4.1 Hail

Poruka se salje kada neki cvor hoce da udje u sistem.

Posljalac: Radnik
Primalac: Bootstrap
Okidac: Zelja cvora da udje u sistem
Akcija primanja: Provera i slanje Contact poruke cvoru koji hoce da udje u sistem

2.4.2 Contact

Bootstrap salje poruku da obavesti da novi cvor koji je njegov kontakt u sistemu. Prosledjuje se kontakt informacije cvora sa najvecim ID-jem.

Posljalac: Bootstrap
Primalac: Radnik
Okidac: Prolazak provere novog cvora sa Hail poruke
Akcija primanja: Kontaktiranje cvora koji ce nas uvesti u sistem
Template poruke: NodeInfo json

2.4.3 Reject

Bootstrap salje poruku da obavesti da novi cvor da, posle provere nece dobiti prilaz sistemu.

Posljalac: Bootstrap
Primalac: Radnik
Okidac: Nije prosla provera unutar Bootstrapa
Akcija primanja: Smrt cvora

2.4.4 Join

Radnik salje poruku kada udje u sistem i dobije svoj jedinstveni ID.

Posljalac: Radnik
Primalac: Bootstrap
Okidac: Radnik je usao u cvor
Akcija primanja: Ubacivanje cvora u tabelu cvorova
Template poruke: id

2.4.5 Leave

Radnik salje poruku kada pokusava da izadje iz sistema.

Posljalac: Radnik
Primalac: Bootstrap
Okidac: Radnik pokusava da izadje iz sistema
Akcija primanja: Izbacivanje cvora iz tabele cvorova
Template poruke: id

2.4.6 Purge

Bootstrap salje komandu svim cvorovima u sistemu da se ugase

Posljalac: Bootstrap
Primalac: Radnik
Okidac: Primanje komande purge na CLIu
Akcija primanja: Gasenje cvora

2.5 Poruke za Radnike

2.5.1 SystemKnock

Radnik pokusava da udje u sistem, pa salje ovu poruku ka svom Kontaktu,

Posljalac: Radnik
Primalac: Radnik
Okidac: Primljena Contact poruka od strane Bootstrap
Akcija primanja: Slanje Welcome poruke ka novom cvoru
Template poruke: ipAddress:port

2.5.2 Welcome

Radnik salje poruku kada novi cvor pokusa da udje u sistem, a cvor je njegov kontakt za ostatak sistema. Salje mu se novi jedinstveni ID, kao i tabelu sa informacijama o ostatku sistema.

Posljalac: Radnik
Primalac: Radnik
Okidac: Kontakt prima SystemKnock poruke
Akcija primanja: Vracanje informacije novom cvoru

2.5.3 Entered

Radnik salje poruku kada udje u sistem, i slicno broadcastu, salje ostatku sistema svoje informacije.

Posljalac: Radnik
Primalac: Nema
Okidac: Radnik je usao u sistem
Akcija primanja: Updateovanje tabele

2.5.4 ConnectionRequest

Cvor trazi bidirekcijalnun konekciju od drugog cvora. Ovo ce se uglavno koristiti za inicijalne konekcije pri IDLE stanju sistema. Takodje treba dodati koji smer konekcije se trazi (prethodnik sledbenik)

Posljalac: Radnik
Primalac: Radnik
Okidac: Radnik trazi konekciju
Akcija primanja: Provera requesta i slanje resposna,Azuriranje IDLE konekcije u suprotnom smeru od requesta
Template poruke: smer

2.5.5 ConnectionResponse

Cvor koji je primio request za konekciju, proverava uslove za konektovanje, pa se vraca odgovor

Posljalac: Radnik
Primalac: Radnik
Okidac: Radnik odgovara request za konekciju
Akcija primanja: Ako je pozitivno, azurira se konekcija u zadatom smeru
Template poruke: boolean:smer

2.5.6 Quit

Radnik pokusava da izadje iz sistema. Poruka se salje svima, kao Broadcast

Posljalac: Radnik
Primalac: Nema
Okidac: Radnik pokusava da izadje iz sistema
Akcija primanja: Updateovanje tabele

2.5.7 ClusterKnock

Radnik pokusava da udje u klaster i uzme svoj prostor za kalkulacije.

Posljalac: Radnik
Primalac: Radnik
Okidac: Radnik pokusava da udje u klaster
Akcija primanja: Slanje ClusterWelcome poruke

2.5.8 ClusterWelcome

Poruka sa informacijama o klasteru za novi cvor u klasteru. Salje se kada cvoru dobije ClusterKnock poruku. Salje se fractalID kao i informacije potrebne o klasteru.

Posljalac: Radnik
Primalac: Radnik
Okidac: Pirmanje ClusterKnock poruke
Akcija primanja: Slanje EnterCluster poruke
Template poruke: {"fractalID":string,"ClusterInfo":json}

2.5.9 EnteredCluster

Radnik salje poruku ka cvorovima klastera o svom ulasku u klaster.

Posljalac: Radnik
Primalac: RadniciKlastera
Okidac: Radnik je usao u klaster
Akcija primanja: Updateovanje table o cvorovima u sistemu
Template poruke: {"NodeInfo":json}

2.5.10 ClusterConnectionRequest

Cvor trazi bidirekcijalnun konekciju od drugog cvora. Ovo ce se koristiti za konekcije unutar klastera.

Posljalac: Radnik
Primalac: Radnik
Okidac: Radnik trazi konekciju
Akcija primanja: Provera requesta i slanje resposna,Azuriranje tabelu konekcija u klasteru

2.5.11 ClusterConnectionResponse

Cvor koji je primio request za konekciju, proverava uslove za konektovanje, pa se vraca odgovor

Posljalac: Radnik
Primalac: Radnik
Okidac: Radnik odgovara request za konekciju
Akcija primanja: Ako je pozitivno, azurira se konekcija u zadatom smeru
Template poruke: boolean

2.5.12 ImageInfoRequest

Cvor trazi informacije o tackama od svojih podredjenim cvorova. Ovo moze da se radi rekurzivno kroz "stablo" klastera. Odnosno ista poruka ce se slati kroz graf sve dok se ne dodje do cvorova bez podredjenih cvorova.

Posljalac: Radnik
Primalac: Radnik
Okidac: Komanda result ili primanje iste poruke od nadredjenog cvora
Akcija primanja: Slanje ImageInfoRequest ka podredjenim cvorovima ili slanje ImageInfo poruke

2.5.13 ImageInfo

Radnik salje informacije o svojim tackama ka nadredjenom cvoru, uglavnom se to desava kada nadredjenom cvoru trazi tu informaciju.

Posljalac: Radnik
Primalac: Radnik
Okidac: Primanje ImageInfoRequest poruke
Akcija primanja: Sumiranje primljenih tacaka
Template poruke: Job object json

2.5.14 StartJob

Kada Unutar klastera dodje do podele posla, cvor koji deli posao salje ovu poruku. Posto cvor dete zna pri ulasku u klaster, kojim ce se poslom baviti, nista mu se nesalje kao poruka

Posljalac: Radnik
Primalac: Radnik
Okidac: Cvor roditelje je odlucio da podeli posao
Akcija primanja: Poceatk izrade posla

2.5.15 StartJobGenesis

Cvor koji je dobio komandu da pocne izradu fraktala, salje poruku cvoru koji ce biti nulti cvor unutar novokreiranog klastera. Taj cvor ce poceti da radi nad fraktalom.

Posljalac: Radnik
Primalac: Radnik
Okidac: Korisnik je zapoceo novi posao
Akcija primanja: Poceatk izrade posla
Napomen: Treba da se salje objekat posla sa ranije generisanim tackama

2.5.16 ApproachCluster

Cvor koji je dobio komandu da pocne izradu fraktala, salje poruku cvoru koji ce biti deo klastera. Taj cvor ce poslti ClusterKnock poruku kontakt cvoru

Posljalac: Radnik
Primalac: Radnik
Okidac: Rasporedjivac je odradio da cvor pripada klasteru
Akcija primanja: Slanje ClusterKnock kontakt cvoru

2.5.17 StopShareJob

Ova poruka se salje kada dolazi do reorganizacije sistema (dodavanja ili brisanje fraktala). Kada se poruka dobije radnik zaustavalj posao i salje informacija o njemu. Takodje se salje i posao koji je doveo do promene sistema tj. raspodele klastera

Posljalac: Radnik
Primalac: Nema
Okidac: Korisnik je promenio broj fraktala u sistemu
Akcija primanja: Zaustavljanje posla i slanje njegovih informacija
Napomena: Ako je krerian dinamicki posao u poruci ce se on proslediti da bi updatovali svoj niz poslova.

2.5.18 StoppedJobInfo

Poruka za slanje informacija o poslu koji je upravo zaustavljen. Ovo je parnjak uz StopShareJob poruku.

Posljalac: Radnik
Primalac: Radnik
Okidac: Primanje StopShareJob poruke
Akcija primanja: Cuvanje informacija o poslu za reorganizovanje sistema

2.5.19 AskForJob

Ako cvor udje u sistem dok se radi na nekom fraktalu treba da udje u klaster.

Posljalac: Radnik
Primalac: Radnik
Okidac: Usao u sistem ali nije u klasteru
Akcija primanja: Slanje ApproachCluster poruke

2.5.20 UpdatedNode

Ako cvor menja neke informacije o sebi, on salje celom sistemu svoje promene

Posljalac: Radnik
Primalac: Nema
Okidac: Promena Radnika
Akcija primanja: Updateovanje tabele sistema
Template poruke: NodeInfo object json

2.5.21 JobStatusRequest

Cvor salje poruku ka svim cvorovima u sistemu kada dobije STATUS komandu od korisnika.

Posljalac: Radnik
Primalac: Radnik
Okidac: Radnik je dobio STATUS komandu
Akcija primanja: Slanje informacija o poslu preko JobStatus poruke

2.5.22 JobStatus

Cvor salje poruku ka svim cvorovima u sistemu kada dobije STATUS komandu od korisnika.

Posljalac: Radnik
Primalac: Radnik
Okidac: Radnik je dobio STATUS komandu
Akcija primanja: Slanje informacija o poslu
Template poruke: JobStatus object json
Napomena: Razlikujemo ovde Job i JobStatus strukturu. Struktura JobStatus treba da je oblika:

```
type JobStatus struct {
    Name            string          `json:"name"`
    PointsGenerated int             `json:"pointsGenerated"`
    WorkingNodes    int             `json:"workingNodes"`
    PointsPerNodes  map[string]int  `json:"pointNodes"`
}
```

3 Kljucne funkcionalnosti

3.1 Dodavanje cvora u sistem

Da bi cvor usao u sistem, bootstrap cvor proverava tabelu postojecih cvorova da li se ponavlja par (ip adresa,port), ako se ne ponavlja cvor ulazi u sistam. Bootstrap salje novi cvor ka poslednjem cvoru u uvezanoj listi.

Ako se neki fraktal izracunava onda ce novi cvor biti prikljucen u klaster po istom pravilu kao i inicijalnoj raspodeli u **Stanje izrade visestrukih fraktala**. Odnosno, ako novi cvor dobije k ID, i prethodni cvor sa k-1 ID-jem i deo je i-tog klastera, onda ce novi cvor pripasti (i+1)-om klasteru (ili 0 klasteru, ako je i-ti klaster i poslednji klaster).

3.2 Izrada jednog fraktala

Zbog jednostavnosti, prvo ce se razmatrati izrada jednog fraktala od strane celog sistema. Ako na sistem, od m cvorova, udje fraktal sa N tacaka, dodeljivanje ce biti u redosledu ulancane liste. Gde ce se prvi cvor postati

osnova klastera, i dobice fractalID 0, i svaki sledeci cvor iz liste ce se pridodati prethodno dodatom cvoru gde ce mu biti zadat fractalID, odakle ce slati zahteve za konekcije unutra klastera. Takodje, svaki cvor dobije korditane od N tacaka fraktala. Kada se ceo klaster sastavi, pocece izrada fraktala.

Pocetna kordinata tacke ce biti u prostoru koji pripada nultom cvoru u klasteru. Dok se tacka nalazi u istom podprotsoru, cvor kome pripada on ce da zapisuje te tacke u svojoj "memoriji". U trenutku kada tacka napusti podprostor cvora, cvor salje poruku cvoru koji zaduzen za taj podprostor. I tako u krug.

3.3 Izrada vise fraktala

Ako na sistem, sa m cvorova, dodje k Fraktala sa $N_0, N_1, ..., N_k$ tacaka. Raspodela cvorova po poslovima, tj. u klastera, ce se raditi po principu Round-robin. Unutar svakog klastera je organizovanje isto kao i za izradu jednog fraktala, jer su klasteri dok obradjuju fraktale nezavisni jedni od drugih. Komplikacije nastaju samo ako dodje do promene unutar celog sistema, i to najvise pri otkazima cvorova.

3.4 Ulazak cvora u vec postojeci klaster

Ako cvor udje u sistem, dok postoji fraktal koji se obradjuje, i bude prikljucen klasteru, bice poslat ka cvoru sa 0 fractalID-em. Tu ce dobiti informacije o klasteru. Posto nulti cvor klastera ima informacije o svim cvorovima unutar klastera, novom cvoru ce biti dodeljen preliminarni fractalID i bice prosledjen u graf klastera. Cvor ce doci do cvora ciji fractalID ce se razlikovati sa fractalID novog cvora samo u krajnje desnoj cifri(ili je ceo ID prefiks). Kada udje u klaster, novi cvor je u IDLE stanju, a cvor "roditelj" povecava broj cvorova koji su u IDLE stanju(a da su u njegovom podsistemu). Kada broj IDLE cvorova postane jednaka $N - 1$ tada se deli prostor na N jednakih podprostora i deli se sa cvorovima koji cekaju, posle toga oni pocinju da rade na svojim podfraktalima.

3.5 Ulazak u sistem

Cvor salje HAIL poruku ka Bootstrap cvoru. Bootstrap radi proveru da li cvor sa istim ip adresom i portom vec postoji u sistemu, ako ne postoji salje mu se CONTACT poruka sa informacijama o cvoru koje ce biti kontakt

za ulazak sistema. Tada novi cvor salje SYSTEMKNOCK poruku ka svom kontaktu. Kada kontakt primi poruku on slje informacije o ID-ju koji ce cvor da preuzme kao i tabelu sistema (informacije o cvorovima). Kada cvor dobije WELCOME poruku on trazi konekcije next i prev, takodje salje informacije sistemu da je usao u sistem (broadcast ENTERED poruku) i Bootstrap cvoru (JOIN poruka).

3.6 Izlazak iz sistema

PROCESSING...

3.7 Podela slike na vise cvorora

Post je slika fraktal, mozemo da iskoristimo osobinu fraktala da se unutar fraktala nalazi fraktali, ili fraktal sadrzi samo sebe. Posto u Igri haosa imamo N kljucnih tacaka, a da bi se podelila slika treba da $N-1$ cvorova da ceka, mi mozemo da podeli sliku na N delova u odnosu na kljucne tacke. Svaki cvor ce dobiti po jednu kljucnu tacku, i ostatak velike slike ce se skalirati u odnosu na dobijenu tacku. Preporuka za skaliranje da bude $\frac{1}{N-1}$, odnosno da zavisi od broja kljucnih tacaka fraktala. Ovako podeljena slika ce dovesti do nezavisni cvorova pri obradi slika.

Posto ce se pri podeli slike, tacke imati apsolutne koordinate, pri spajanju slika, mozemo samo da spojimo sve slike u jednu sliku, odnosno da spojimo sve slike od podslika u jedanu listu i cinece veliku sliku.

Napravljeni su nekoliko primera za laksu ilustraciju

- Skalirani trougao (LINK)
- Skalirani cetvorougao (LINK)
- Skalirani petougao (LINK)

3.8 Obrada slike

Pocetna tacka fraktala je poslednja tacka u listi tacaka(ovde se nalaze i kljucne i generisane tacke) Ako imamo fraktal sa N tacaka i u trenutku imamo tacku T,koja sluzi kao generisana tacka, sledeca tacka ce se generisati na sledeci nacin:

- Nasumicno biramo jednu od N kljucnih tacaka.
- Sledeca tacke ce se nalaziti na putu izmedju izabrane tacke i tacke T. Slicno kao i skaliranje tacke.
- Novogenerisana tacka ce postati novo T, za sledecu iteraciju.

Primer generisanja nove tacke:(Link)

3.9 Rutiranje

Rutiranje poruka se moze svesti na dva nacina rutiranja zavisno u kakvom okruzenju u kojem je cvor:

- Rutiranje kroz prsten
- Rutiranje kroz klaster preko fractalID-a

Najoptimalnije (tj. najmanji broj skokova pri slanju poruka) bi bilo da cvorovi imaju neki oblik inteligencije da sam odluci koji nacin rutiranja ce odabrati. I zbog organizovanosti sistema ovo se moze lako odrediti. Tako da za svaki nacin rutiranja mozemo lako da odredimo distancu izmedju cvorova.

Rutiranje kroz prsten

Posto svaki cvor ima tabelu svih cvorova distanca izmedju dva cvora moze se lako odrediti pomocu aritmetika:

$$dist(u,v) = \min\left(u.ID - v.ID, (size(G) - u.ID) + v.ID\right)$$
$$G - grafsistem; u, v \in G; u.ID > v.ID$$

Rutiranje kroz klaster

Iz slicnog razloga kao i prethodno rutiranje postoji lak nacin da se odredi distanca izmedju dva cvora koristeci fractalID. Nacin na koji ce se naci distanca izmedju dva cvora jeste preko edit distance dva fractalID-a.

Put izmedju dva nepovezana cvora se odredjuje ovako: Prolazimo kroz FractalID cvorova sa leva na desno i pri svakom i-tom skoku skacemo na cvor koji se podudara sa ciljnim cvorem na i-toj poziciji.

Ovde samo treba obratiti paznju na put izmedju dva cvora koji nisu i istom klasteru. U tom slucaju predlog je da se prvo nadje najblzi cvor koji pripada trazenom klasteru.

Tako da na kraju cvor odrediti dve distance za slanje i odabrace onu sa najmanjim brojem skokova.

4 Specijalne funkcionalnosti

4.1 Raspored poruka pri kreiranju klastera

Klasteri pocunju da se kreiraju kada jedan od cvorova u sistemu dobije komndu START/STOP za neki fraktal. Taj cvor ce biti Rasporedjivac cvorova po klasterima. Prvo on salje svim cvorovima u sistemu poruku StopShareJob, kojima se svim cvorovima kaze da zaustave svoje izrazde fraktala i vrate informacija o istim. Kada cvor Rasporedjivac dobije StoppedJobInfo od svih cvorova u sistemu. On prolazi kroz poslove koji rade i Round Robin nacinom salje poruku StartJobGenesis cvorovima i oni ce biti Nulti cvorovi (FractalId = 0) u svojim klasteriam. Nakon toga ostatku sistema, Rasporedjivac ce slati ApproachCluster poruku sa informacijama o kontaktu za klaster u koji treba da udju. Rasporedjivanje po klasterima je takodje Round robin.

Kada cvor dobije ApproachCluster poruku on salje svom kontaktu ClusterKnock kao zahtev za ulazak u klaster. Tada kontakt salje ClusterWelcome. Kada cvor udje u klaster on prvo salje ClusterConnectionRequest svim cvorovima ciji je FractaID EditDistanc=1, i salje svim cvorovima EnteredCluster poruku, da bi obavestio sve da je usao u klaster.

4.2 Ulazak cvora u sistem ako postoji rasporedjeni klasteri

Posto pri ulasku u sistem svaki cvor ima kontakt cvor, isti kontakt cvor ce mu proslediti, uz Welcome poruku, i ApproachCluster poruku sa informacija o novom kontaktu za klaster. Posto cvor ima znanje o svoim poslovima on zna kojem klasteru ce da posalje (ID mod [broj radecih poslova]).

4.3 Raspodela FractalID

Posto su fractalID-evi ne rade po konvencijalnim principima matematike, vec prate neka svoja pravila. Ovde ce biti objasnjena. Pri raspodeli fractalID postoje intervali(svi fractalID-evi su iste duzime) gde raspodela prati pravila n-arnih sistema. Specijalni slucajevi su kada treba da se doda nova cifra u fractalID, tada treba pratitit dva pravila: Ne dodeljuju se brojevi sa zadnjom cifrom 0, jer ce ti brojevi pripasti cvorovima roditeljima koji su vec u sistemu(pri raspodeli poslova medju decom, cvor roditelj ce promeniti svoj fractalID iz X to X0); Kada dodje do overflowa: svi brojevi se menjaju sve cifre se pretvaraju u 0 (sem poslednje koja postaje 1).
Npr. neka je N = 3 Raspodela fractala ce ici ovako:
0 => 1 => 2 => 01 => 02 => ... => 22 => 001
Link ka Demo kodu napisanom u Python-u: LINK
Takodje postoji problem poredjenja brojeva i to se resava na sledeci nacin:

- Ako je broj duzi on je i veci
- Ako su brojevi iste duzine onda se porede kao i obicni n-arni brojevi

5 Komande

5.1 Komande za bootstrap

5.1.1 quit

Argumenti: None
Opis: Bootstrap se gasi
Primer: quit

5.1.2 purge

Argumenti: None
Opis: Bootstrap se gasi citav sistem
Primer: quit

5.2 Komande za worker

5.2.1 quit

Argumenti: None
Opis: Worker se gasi
Primer: quit

5.2.2 status

Argumenti: X id (X - naziv posla, id - fractalId)
Opis: Prikazuje stanje svih započetih izračunavanja - broj tačaka na svakom fraktalu. Naznačava za svaki fraktal koliko čvorova rade na njemu, fraktalni ID, i koliko tačaka je svaki čvor nacrtao. Ako se navede X kao naziv izračunavanja, onda se dohvata status samo za njega. Ako se navede posao i fraktalni ID, onda se dohvata status samo od čvora sa tim ID.
Primer: status Fractal1 011

5.2.3 start

Argumenti: X (X - naziv posla)
Opis: Započinje izračunavanje za zadati posao X. X može da bude simboličko ime nekog posla navedenog u konfiguracionoj datoteci. Ako se X izostavi, pitati korisnika da unese parametre za posao na konzoli. Proveriti da je ime posla jedinstveno, kao i da su svi parametri validnih tipova. Ako je ovo K-ti posao u sistemu, neophodno je da ima makar K čvorova aktivno. Ako nema K čvorova aktivno, ne startovati posao.
Primer: status Fractal0

5.2.4 result

Argumenti: X id(X - naziv posla, id - fractalId)
Opis: Prikazuje rezultate za završeno izračunavanje za posao X. Korisnik može, a ne mora da navede fraktalni ID za rezultat. Ako se izostavi, onda se dohvata rezultat za ceo posao, u suprotnom samo za taj fraktalni ID. Slika treba da se eksportuje kao PNG.
Primer: result Fractal0 023

5.2.5 stop

Argumenti: X (X - naziv posla)
Opis: Zaustavlja izračunavanje za posao X. Fraktal u potpunosti nestaje iz sistema, i čvorovi se preraspoređuju na druge poslove. **Primer**: status Fractal0