VILNIAUS UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
PROGRAMŲ SISTEMŲ KATEDRA

Laikinų sutrikimų poveikio sesijų valdymo algoritmui tyrimas

Impact of temporary failures on session management algorithm

Kursinis darbas

Atliko: Mantas Petrikas (parašas)
Darbo vadovas: doc. Karolis Petrauskas (parašas)

Vilnius – 2021

TURINYS

Įvadas

Tyrimo objektas

Šiame darbe analizuojamas algortimas, skirtas asinchroniškai valdyti sesijų komunikaciją tarp skirtingu paskritytos sistemos grupių (angl. „clusters") [PB18; PB19]. Algoritmas aprašytas TLA$^+$ specifikavimo kalba [Lam$^+$], jo korektiškumas yra tikrinamas atviro kodo „TLA$^+$ toolbox" įrankiu [KLR19].

Darbo aktualumas

Moderios didelio praleidumą turinčios sistemos apdoroja tukstančius, ar net milijonus užklausų vienu metu, ir greitai ir patikimai gražinti duomenis. Kad efektyviai pasiekti rezultatus, sistemos projektuojamos numatant galimę plėsti jas didiant serverių skaičių ir paskirstant apkrovą tarp jų [Son16].

Paskirstytos sistemos suteikia turi daug privalumų - padidėja sistemos prieinamumas, sumažinamas nekritinių sistemos dalių poveikis visos sistemos veiklai, suteikiama galimybė didinti sistemos pralaidumą paralelizuojant ir horizontaliai plečiant sistemos resursus. Tačiau paskirstytos sistemos turi ir minusų. Dauguma paskirstytų sistemų problemų susiję su sudėtingesne infrastrukūra ir diegimu (ang. deployment), sistemos būsėnos valdymu ir vientisumo išlaikymu visose sistemos dalyse [Thö15]. Norinti išlaikyti didelį sistemos pasiekiamumą ir operacijų greitumą, dalis sistemų naudoja asynchoninis bendravimą tarp skirtingų sistemų dalių, užtikrinant tik galutinį būsenos nuoseklumą (ang. eventual consistency) [Vog09].

TLA$^+$ yra formali, predikatų logiką paremta, aukšto lygio modeliavo kalba, leidžianti projektuoti, aprašyti ir tikrinti sistemų korektiškumą [**kuppe2019tla**]. Ši kalba yra naudojama didžiosiose kompanijose, „Intel" [BL02] „Amazon" komponijoje [NRZ$^+$15], Microsoft ir kitose. Specifikavimo kalba leidžia naudojant predikatų logiką aprašyti pradinę sistemos būseną, leistinus sistemos būsenos pakitimus ir savybės kurias modelis turi tenkinti, tada naudojant „TLA toolbox" įrankį" patikrinti ar visos sistemos būsenos tenkina pasirinktas savybes. Toks formalus modelio būsenų tikrinimas tinkrina modelio logines algortimo savybės, ir tai leidžia patikti retas sistemos busenas ir panaudos scenarijus, apie kurios nevisada pagalvoma rašant algortimus, atliekant statinę kodo analizę ar testuojant [NRZ$^+$15]. Šis metodas leidžia aptinkti galimas klaidas prieš išmeginant sistemą realioje aplinkose, taip sumažinant sistemos taisymo kaštus ir užkertant kritiniams sistemos sutrikimams, kas yra ypatingai svarbu kritinėse sistemose ar sistemose kurios siekia turėti dideli pasiekiamumą.

Moksliniame straipsnyje „Effectiveness of the asynchronous client-side coordination of cluster service sessions” aprašomas algortimas, galintis sumažinti į neveikianti mazga nukreiptų užklausų skaičių, stebinti sistemos klientų busenas ir informuojant kitus sistomos klientus apie kliento aptinktus sutrikimus. Modelyje nenumatomi laikini tiklo sutrikimai, dėl kurių kai kurie mazgai siunčiantiems užklausas gali laikinai tapti klientams gali atrodyti nepasiekiami. Šis darbas nagrinėja algoritmo elgseną, papildydamas algortimo modelį laikinų sutrikimų būsenomis.

Darbo tikslas

Tikslas: Išanalizuoti sesijų valdymo algoritmą, įvertinti laikinių tinklo sutrikimų poveikį ir apžvelgti būdus kaip galima būtų patobulinti algoritmą.

Darbo uždaviniai

- Atlikti mokslinės literatūros analizę, nagrinėjant panašius sesijų valdymo algoritmus.
- Išanalizuoti sesijų valdymo algoritmo modelį.
- Suformuluoto savybę, įrodancią, kad algoritmas veiks tinkamai, nepavyks prisijungti prie vienintelio aktyvaus tiekėjo mazgo.
- Naudojant TLA+ toolbox programinį paketą įrodyti, kad egzistuoja sistemos būsena kurioje pateikta savybė nėra tenkima.
- Pasiūlyti kaip būtu galima patobulininti algoritmą, kad pateikta savybė būtų tenkinama.

Tyrimo metodai

Darbe atliekama asynchoninio sesijų valdymo algoritmo, aprašyto TLA$^+$ modeliavimo kalba analizė, praktiškai tikrinamas jo savybių tesingumas naudojant „TLA$^+$ toolbox” įrankį, testuojant skirtingus modelio parametrus. Taip pat, darbe apžvelgiama paskistytų asynchroniškai bendraujančių systemų valdymo algortimų mokslinės literatūra.

Darbo struktūra

Pirmajame skyriuje aprašomas sesijų valdymo algoritmo modelis. Antrajame skyriuje aprašomas aprašyto modelio papildymas tinklo sutrikimais, ir poveikis sistemos modeliui. Trečiajame skyriuje nagrinėjama algoritmo patobulimo galimybės.

1. Literatūros ir esamų metodų apžvalga

Siame skyriuje apžvalgiama su sesiju valdymu susijusi literatūra

2. Sesiju panaudojamas

Sesijos saugo vartotojo informacija apie dabartine saveikos busena su programa. Tai naudojama internetinese programose,

Įprasta internetine programa išlaiko kiekvieno prisijungusio vartotojo sesiją tol, kol vartotojas yra prisijungęs. Sesijos būsena yra tai, kaip programos prisimena vartotojo tapatybę, suasmeninimo informaciją, prisijungimo duomenis, ar naujausius veiksmus. Sesijoje kaupiami duomenys leidzia greiciau pasiekti reikiamus duomenis, daznai

Sesijos gyvavimo laikas pasibaigia, kai vartojas specifiškai iššaukia atsijungimo veiksmą arba tam tikra laiko tarpą kvieciami su sesija susije veiksmai. Priklausimai nuo sesijos paskirties, kai kurie duomenys gali būti išsaugoti duomenų bazėje, vėlesniam naudojimui, arba trumpalaikės informacija gali būti sunaikinama pasibaigus sesijai.

Sesijos būsena yra panaši į spartinančiąją atmintį (ang. cache), skiriasi tik duomenų valdymo modelis. Spartinančioji atmintis yra tolerantiška duomenų praradimui ir ją bet kada galima atkurti iš pirminės duomenų bazės, tačiau atnaujinant spartinančios atminties duomenis reikia atnaujinti ir pagrindinę duomenų talpyklą. Sesijos duomenis sukuriami prasidedant sesijai, ir vėliau nėra garantuojama kad sesijos duomenys nepasikeis t.y. juos bus galima atkurti duomenų šaltinio. Sesijos duomenys dažniausiai išsaugomi į pagrindinę talpyklą tik pasibaigus sesijai. Sesijos duomenys gali būti laikini arba nuolatiniai - t.y. pasibaigus vartotojo sesijai duomenys gali būti sunaikinami arba išsaugoti vėliasniam naudojimui..

3. Resursu valdymas

Norinti sumažinti programos prieeinamumo laikotarpį,

4. Komunikacijos klaidos

Duomenų perdavimo sutrikimai gali atsitiki dėl per didelės tinklo apkrovos, aptarnaujančios serverio perkrovos, techninės ar programinės įrangos klaidos, paslaugų teikimą trikdančių (angl. denial-of-service) atakų [ABL+08].

Session pool management

Resource management Redudancy Resource sharing Session management Network faults Net split False negatives

t56

5. Algortimo analizė

Šiame skyriuje aprašomas darbe nagrinėjamas asynchroniško sesiju valdymo algoritmo [PB18] modelis, pagrindiniai algoritmo veikimo principai, daromos prielaidos, galimi modelio ir realios sistemos neatitikimai.

Modelis sudarytas iš 3 sistemų grupių. Kientų-vartotojų (angl. consumers), kurie atitinka siekiančių gauti informacija ar laikiantys tam tikro rezultato, tiekėjų (angl. providers), kurie aptarnauja kientus teikdami duomenis ar atlikdami funkcijas. Bendravimas tarp vartotojų ir tiekėjų vyksta sesijų pagalba ir koordinatorių (angl. coordinators), valdančių vartotojų ir tiekėjų komunikaciją. Tiek vartotojų, tiek vartojų grupėje esantys mazgai (angl. nodes) patys nežino apie kitų to pačio tipo mazgų egzistavimą, ir neturi valdančio proceso (angl. master node), kuris galėtų valdyti bendravimą tarp jų.

Tinklo topologija modelyje nėra detalizuojama, tačiau daroma prielaida kad visi koordinuojantys mazgai žino apie visus kitus koordinuojančius mazgus, o sesijos gali sudaryti ryšį su bet kuriuo tiekėjo mazgu. Tai nebūtinai atitinka realiaus panaudojimo atvejus, nes paskirtų sistemų dalys gali būti veikti skirtinguose pasaulio vietose ir nebutinai būti vienodai pasiekiamos skirtuose tinkluose (ang. subnets) tačiau sistemos topologijos problematika nėra nagrinėjama šiame darbe.

5.1. Modelio būsena

Modelis aprašomas TLA$^+$ modelio kalba, ir jo savybių tikrinui naudojamas TLA+ toolbox įrankis, todėl norint patikriniti modelio teisingumą reikia apriboti galimų tinklo mazgų kombinacijų skaičių. Tai daroma keičiant modelio konfiguracija, keičiant sesijų, tiekėjų ir vartotojų skaičių prieš atliekant formalų modelio tikrinimą. Šie parametrai nesikeičia modelio tikrinimo metu, tačiau gali būti pakeisti prieš paleidžiant skirtingas simuliacijas. Modelyje laikoma kad kiekvienas vartotojas gali turi vienoda sesijų kiekį, tačiau sesija nebūtinai gali būti prisijungusi prie vartotojo. Kiekvienas vartotojo mazgas turi savo koordinatorių.

5.2. Tiekėjų būsenos pakeitimai

Modelyje tiekėjai turi 2 būsenų perėjimus. Jei tiekėjas veikia, jis gali tapti neveikiančiu ir atvirščiai- neveikiantis gali tampti veikiančiu. Sąlygos dėl kurių tiekėjai gali nustoti veikti ar vėl pradėti veikti nėra apibrėžtos ir modelyje nemodeliuojamos. Tiekėjai patys nepraneša aptarnaujanties mazgams apie savo būsenos pakeitimus, žinučių apie klientų būsenų pasikeitimus sekimu rupinasi prie tiekėjo prisijungę vartotojai.

5.3. Vartojų būsenos pakeitimai

Modelyje vartotai turi 3 būsenų pakeitimus: vartotojo sesijos prisijungimą prie veikiančio tiekėjo, sesijos atsijungimą gavus žinutę iš koordinatoriaus apie neveikiantį tiekėją, sesijos atsijungimą nustojus veikti tiekėjui.

Komunikacija tarp tiekėjo ir vartotojo šiame algortimo modelyje nėra modeliuojama, nes normalus sistemos veikimas nekeičia sistemos būsenos, nes laikoma kad klientas yra visada lieka prisijungęs prie to pačio tiekėjo.

5.4. Koordinatoriaus būsenos pakeitimai

Modelyje koordinatoriai turi 3 būsenų pakeitimus: žinučių iš sesijų apie neaktyvius tiekėjus apdoriją, pranešimą kitiems koordinatoriams apie neveikiančius tiekėju mazgus, tikrinimą ar tiekėjai tapo aktyviais.

Koordinuojantis mazgas, gavęs pranešimą apie neveikiantį tiekėją, išsisaugo savo busenoje tiekėja kaip neveikiantį ir asynchroniškai praneša visoms savo valdomoms sesijoms apie neveikiantį mazgą.

Koordinatorius žinantis apie neveikiantį mazga, nuolatos asynchroniškai tikrinina tiekėjo būsena ir sužinojęs apie pradėjusi veikti mazgą, atnaujina savo būseną, pažyminti tiekėją kaip veikiantį.

Klientai, iš koordinuojančios gavę žinutę apie neveikiantį tiekėjo mazgą, kiekvienai sesijai atskirai asynchronišku būdu nurodo atsijungti nuo kliento. Atsijungusios sesijos gali prisijungti prie kito veikiančio tiekėjo. Modelyje apraščiame darbe [PB18] daroma prielaida, kad sesijoje vieno tiekėjo pakeimas kitu yra leidžiamą operacija, nors pabrėžiama, kad realiame scenarijuje tiekėjo mazgo pakeitimas gali pasireikšti sistemos veikimo suletėjimu ar nenuoseklių (ang. inconsistent) duomenų atsiradimu, todėl šios operacija turėtų būti kaip įmanoma labiau vengiama.

6. Tinklo sutrikimų modeliavimas

Analizuojant modelį, buvo iškelta hipotezė, kad algoritmas gali neveikti jei tinkle, kuriame vyksta komunikacija, įvyks komunikacijos tarp skirtingų mazgų sutrikimai. Šie sutrikimai nagrinėjamame TLA $^+$ modelyje nėra modeliuojami, todėl jis buvo papildytas busenų pakeitimais.

Rezultatai ir išvados

Literatūra

[ABL+08] Narjess Ayari, Denis Barbaron, Laurent Lefevre ir Pascale Primet. Fault tolerance for highly available internet services: concepts, approaches, and issues. *IEEE Communications Surveys & Tutorials*, 10(2):34–46, 2008.

[BL02] Brannon Batson ir Leslie Lamport. High-level specifications: lessons from industry. *International Symposium on Formal Methods for Components and Objects*, p. 242–261. Springer, 2002.

[KLR19] Markus Alexander Kuppe, Leslie Lamport ir Daniel Ricketts. The tla+ toolbox. *arXiv preprint arXiv:1912.10633*, 2019.

[Lam+] Leslie Lamport ir k.t. Tla+ toolset, 2009. *URL http://www. tlaplus. net/tools/tla-toolbox/.-Cited on*:91.

[NRZ+15] Chris Newcombe, Tim Rath, Fan Zhang, Bogdan Munteanu, Marc Brooker ir Michael Deardeuff. How amazon web services uses formal methods. *Communications of the ACM*, 58(4):66–73, 2015.

[PB18] Karolis Petrauskas ir Romas Baronas. Asynchronous client-side coordination of cluster service sessions. *International Baltic Conference on Databases and Information Systems*, p. 121–133. Springer, 2018.

[PB19] Karolis Petrauskas ir Romas Baronas. Effectiveness of the asynchronous client-side coordination of cluster service sessions. *Databases and information systems X: 13th international Baltic conference on databases and information systems, DB&IS 2018, held in Trakai, Lithuania, in July 2018*, p. 95–108, 2019.

[Son16] Rahul Soni. *Nginx*. Springer, 2016.

[Thö15] Johannes Thönes. Microservices. *IEEE software*, 32(1):116–116, 2015.

[Vog09] Werner Vogels. Eventually consistent. *Communications of the ACM*, 52(1):40–44, 2009.