Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

Analýza odolnosti Stellar konsenzus protokolu voči Byzantínskym chybám

Bakalárska práca

2019
Samuel Sládek

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

Analýza odolnosti Stellar konsenzus protokolu voči Byzantínskym chybám

Bakalárska práca

Študijný program:	Informatika
Študijný odbor:	2508 Informatika
Školiace pracovisko:	Katedra informatiky
Školiteľ:	RNDr. Tomáš Kulich, PhD.

Bratislava, 2019
Samuel Sládek

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Samuel Sládek
Študijný program: informatika (Jednoodborové štúdium, bakalársky I. st., denná forma)
Študijný odbor: informatika
Typ záverečnej práce: bakalárska
Jazyk záverečnej práce: slovenský
Sekundárny jazyk: anglický

Názov: Analýza odolnosti Stellar konsenzus protokolu voči Byzantínskym chybám
Analyzing Byzantine Fault Tolerance of Stellar Consensus Protocol

Anotácia: Stellar konsenzus protokol (SCP) vylepšuje štandardné konsenzus protokoly možnosťou čiastočne otvoreného členstva: Nové uzly sa môžu zúčastňovať protokolu aj keď sú známe iba niektorým existujúcim uzlom. Navyše, pokiaľ sa útočník nezmocní výrazného množstva dôveryhodných uzlov, SCP ostáva odolný voči Sybil útokom.

SCP uvádza, za akých podmienok je sieť odolná voči Byzantínskym chybám, napriek tomu doteraz nie je známy algoritmus na overenie týchto podmienok pracujúci v polynomiálnom čase.

Cieľom práce je zaradiť tento problém do triedy zložitosti. Popísať niektoré základné nutné podmienky pre bezpečnosť siete. A analyzovať kvóra, ich prieniky a odolnosť celej siete voči Byzantínskym chybám na náhodne prepojených sieťach.

Vedúci: RNDr. Tomáš Kulich, PhD.
Katedra: FMFI.KI - Katedra informatiky
Vedúci katedry: prof. RNDr. Martin Škoviera, PhD.

Dátum zadania: 25.10.2018

Dátum schválenia: 27.10.2018 doc. RNDr. Daniel Olejár, PhD.
garant študijného programu

.....
študent

.....
vedúci práce

Poďakovanie

Chcel by som sa poďakovať môjmu školiteľovi RNDr. Tomášovi Kulichovi, PhD. za návrh témy a podpory v objavovaní pre mňa nového sveta kryptomien.

Rovnako Patrikovi Bakovi za skvelé rady, postrehy a najmä možnosti vyrozprávania sa o nástrahách čakajúcich v priebehu vyhotovovania práce.

Abstrakt

Kryptomena Stellar pred pár rokmi prešla na nový protokol, ktorý má vylepšovať dovtedy známe konsenzus protokoly najmä zjednodušením možnosti vstupu do siete pre novú entitu. Táto práca analyzuje odolnosť tohto protokolu voči zlyhaniam jednotlivých entít spolupracujúcich na konsenze. Najprv ukáže, že pre danú sieť nevieme v polynomiálnom čase povedať pri koľkých zlyhaniach bude sieť stále bezpečná a ani pri koľkých zlyhaniach bude sieť stále schopná dohodnúť sa na konsenze. Neskôr sa pozrie na niektoré zaujímavé typy sietí pri ktorých vieme vyjadriť odolnosť rýchlo. Vytvorí tiež algoritmus na presné vyjadrenie odolnosti malých sietí a aj rýchlejší algoritmus na odhady zhora pre odolnosť väčších sietí. Na záver rozanalyzuje bezpečnosť a odolnosť náhodne vytvorených sietí. Túto analýzu ukončí odporúčaním volenia parametrov siete aby bola sieť schopná prežiť čo najviac zlyhaní.

Kľúčové slová: Stellar, konsenzus, byzantínske chyby, kvórum, bezpečnosť, odolnosť siete

Abstract

A few years ago, cryptocurrency Stellar switched to a new protocol to improve previously known consensus protocols in particular by simplifying network entry for a new entity. This thesis analyzes the resilience of this protocol to Byzantine failures. Firstly it shows impossibility to evaluate number of failures in which the network is still safe and the number of failures in which the network can still agree on consensus in polynomial time. Later, it considers some interesting types of networks in which we are able to calculate resilience quickly. It creates algorithm to accurately express resilience for small networks and faster one to determine the upper bound for resilience of larger networks. At the end it analyzes security and resilience of randomly created networks. This analysis ends with recommendations for choosing network parameters to make resilience as high as possible.

Keywords: Stellar, consensus, Byzantine failures, quorum, safety, network resilience

Obsah

Zoznam obrázkov

Úvod

V posledných rokoch nadobudli kryptomeny na popularite a výskum v tejto oblasti sa raketovým tempom začal posúvať. Dnes už máme tisícky rôznych kryptomien, z ktorých každá svojím spôsobom vylepšuje dnešnú existujúcu finančnú sieť. Najväčším prínosom je schopnosť prevodu peňazí bez centrálnej autority, no aj tak tu ostalo mnoho výziev, ktorými by sme dokázali zlepšiť aktuálne podmienky v sieťach jednotlivých kryptomien.

My sme sa rozhodli bližšie pozrieť na kryptomenu Stellar. Pre túto kryptomenu bol pred 3 rokmi vyvinutý nový konsenzus protokol. Tento protokol mal za úlohu zachovať všetky dobré vlastnosti doteraz vyvinutých protokolov, zrýchliť peňažné transakcie a najmä zabezpečiť otvorené členstvo pre nové entity, ktoré sa chcú do siete pridať. Poslednú vlastnosť dovtedy žiadny konsenzus protokol bez centrálnej autority zabezpečiť nevedel.

Jednou z dôležitých vlastností Stellar konsenzus protokolu je, že sa dokáže vysporiadať aj so zlyhaním niekoľkých entít a udržať sieť bez poškodenia schopnú naďalej vykonávať transakcie. Koľko takýchto zlyhaní však vie sieť prežiť závisí od rôznych parametrov siete a dodnes nie je známe ako rýchlo vieme pre konkrétnu sieť zistiť tento počet.

V našej práci si najprv predstavíme jednotlivé vlastnosti, ktoré od protokolu budeme očakávať a následne si predstavíme samotný Stellar konsenzus protokol. Potom si zadefinujeme jednotlivé metriky podľa ktorých budeme merať odolnosť siete voči zlyhaniam a ukážeme, že zmerať tieto metriky vo všeobecnosti nebudeme vedieť v polynomiálnom čase od veľkosti popisu siete. Následne sa preto pozrieme na konkrétnejšie typy sietí kde tieto metriky budeme vedieť určiť a dostaneme lepší pohľad na závislosť metrík od stavby siete. Na záver rozanalyzujeme koľko zlyhaní dokážu zhruba prežiť náhodné siete. A vyvinieme odporúčanie ako nastaviť niektoré parametre a pravidlá aby sieť mohla očakávať, že v priemernom prípade bude schopná prežiť čo najväčší počet zlyhaných entít.

Kapitola 1

Problém konsenzu

Problém konsenzu vyžaduje dohodu medzi viacerými entitami na jednej a tej istej *hodnote* (*hodnota* môže byť ľubovoľný typ informácie). Tento problém začne byť zaujímavý keď pripustíme, že niektoré z entít podieľajúcich sa na konsenze budú nespoľahlivé alebo dokonca zlyhajú. V takomto prípade budeme očakávať od riešenia (súbor pravidiel, ktoré keď budú uzly dodržiavať dosiahnu konsenzus, ďalej ho budeme nazývať *protokol*), že entity, ktoré ešte pracujú podľa protokolu budú schopné dosiahnuť konsenzus aj napriek týmto zlyhaniam.

Aby sme vylúčili triviálne riešenia (všetky entity odsúhlasia konsenzus na vopred dohodnutej konštante a podobne), výsledná dohodnutá hodnota musí závisieť na vstupe daných entít, preto pred začatím procesu dohadovania konsenzu musia entity navrhnúť kandidátov. *Kandidáti* budú hodnoty z ktorých entity následne vyberú jedného, na ktorom sa všetci zhodnú a ustanovia konsenzus. Celý konsenzus protokol musí mať teda tri fázy: entity navrhnú kandidátov na hodnoty, entity spolu komunikujú a na konci každá entita oznámi, na ktorom kandidátovi na hodnotu sa dohodli, pričom všetky nezlyhané entity oznámia toho istého kandidáta.

Tento problém je jeden z najzákladnejších problémov informatiky. Mnohí informatici ho riešili už niekoľko desaťročí a dokázali prísť na mnohé zaujímavé dôsledky (napríklad *dôkaz nemožnosti konsenzu* [10]). Takto zadaný problém je však veľmi všeobecný a dá sa riešiť na rôznych modeloch, kde si konkretizujeme naše entity, spôsob komunikácie medzi nimi a ďalšie vlastnosti siete entít. A teda napriek *dôkazu nemožnosti konsenzu* [10] dokážeme dnes za istých podmienok spravovať distribuované systémy.

1.1 Model siete v našej práci

Teraz si zadefinujeme model siete, ktorý budeme uvažovať v celej práci. Napriek tomu, že náš model bude teoretický, všetky rozhodnutia konkretizovania modelu sa budeme snažiť robiť na základe skúseností z praxe aby prípadná implementácia do reálneho

distribuovaného systému nebola obtiažna, ale zároveň nebudeme brať mnohé reálne problémy do modelu, ktoré priamo nesúvisia s problémom konsenzu a iba by komplikovali konsenzus a dôkazy správnosti v ňom.

V našej práci nás bude tento problém zaujímať hlavne v distribuovaných systémoch. A preto v našom modeli entity budú jednotlivé počítače v sieti. V celej práci budeme počítače (alebo ľubovoľné entity) v sieti nazývať *uzly*.
Sieť je množina uzlov, ktoré sú poprepájané každý s každým a vedia si teda medzi sebou vymieňať ľubovoľné správy. Nebudeme dávať žiadne obmedzenia na veľkosť správ a ani ich podobu. Dokonca nebudeme mať žiadne predpoklady na poradie doručenia správ. Avšak budeme predpokladať, že správy budú v konečnom čase doručené v nezmenenej podobe (takýto predpoklad nám v reálnej sieti vie ponúknuť kvalitné šifrovanie). Celá komunikácia bude prebiehať asynchrónne a teda nepredpokladáme žiadnu synchronizáciu uzlov.

Takisto v našom modeli povolíme byzantínske chyby uzlov. Čo znamená, že na správanie uzlov nebudeme dávať žiadne predpoklady. Toto zodpovedá aj reálnemu svetu, kde uzol môže zlyhať po hardvérovej alebo softvérovej chybe alebo dokonca nepriateľ vie upraviť vykonávaný kód (na tento upravený kód už nemôžeme mať žiadne predpoklady) na uzle. V poslednom z prípadov keď nepriateľ upraví vykonávaný kód na uzle budeme hovoriť, že útočník sa „zmocnil“ uzlu.

A teda uzly budeme vždy triediť do dvoch základných kategórií, tie čo sa správajú podľa dohodnutého protokolu (budeme hovoriť, že sa držia protokolu) a tie ktoré porušujú protokol (nebudeme rozlišovať z akého dôvodu). Uzlu ktorý porušuje protokol budeme hovoriť, že *zlyhal*.

1.2 Od teoretického modelu k bankovému sektoru

Na prvý pohľad nečakane, môžeme konsenzus využiť vo finančnej sieti. Schvaľovanie konsenzu zodpovedá schvaľovaniu finančných transakcií a uzly sú jednotliví používatelia peňazí. Stav siete je vlastne zoznam uskutočnených transakcií (konkrétne implementácie ukladania a manipulovania so stavom necháme na jednotlivých protokoloch). Tento zoznam sa zvykne nazývať *bloková reťaz*, keďže každú transakciu vieme zapísať ako jeden blok dát o nej a tieto transakcie vieme zoradiť za sebou do reťaze, ktorú postupne rozširujeme a na jej koniec zapisujeme nové transakcie. Dnešná celosvetová finančná sieť v takomto jednoduchom ponímaní neexistuje. Na svete existuje veľa rôznych mien a prevody peňazí medzi menami nie sú až také jednoduché ako prevody v jednej mene.

Celkovo svet vyzerá ako mnoho malých (alebo väčších) sietí s centrálnou autoritou, ktorou sú banky, ktoré nám naše transakcie musia schváliť. A teda prevody peňazí z jedného konca sveta na druhý sú náročné. Neskôr si popíšeme aké sú nevýhody tohto

systému s jednou autoritou (nazývané aj centralizované systémy) a aké sú možnosti náhrady tohto systému.

Na záver podkapitoly si zhrňme všetky očakávania od finančnej siete, ktoré aktuálna finančná sieť nespĺňa a v ďalšej podkapitole si predstavíme protokoly na nájdenie konsenzu, ktoré sa snažia aspoň niektoré vlastnosti spĺňať (a naopak tým zlým sa vyhýbať). V kapitole 2 potom predstavíme *Stellar konsenzus protokol*, ktorý bude všetky tieto vlastnosti spĺňať a budeme sa mu v práci venovať hlbšie.

- **Jedna globálna** – chceme nahradiť veľa aktuálnych sietí jednou celosvetovo použiteľnou

- **Decentralizovaná** – nechceme nechať štát a banky kontrolovať náš finančný tok, chceme nech všetci zapojený v sieti dohliadajú na dodržovanie pravidiel

- **Nízka latencia** – nech už posielame peniaze kdekoľvek chceme, aby bola transakcia vykonaná v čo najkratšom čase

- **Otvorené členstvo v sieti** – žiadne prekážky k vytvoreniu si účtu, tak ako dnes v mnohých bankách, možnosť otvoriť si účet kdekoľvek na svete v čo najkratšom čase

- **Otvorené členstvo medzi schvaľovateľmi** – žiadne prekážky ku zapojeniu sa do schvaľovaciemu procesu

- **Flexibilná dôvera** – najradšej by sme si sami určili, komu dôverujeme a neboli zaviazaný bankovým inštitúciám, ktoré môžu rôznymi spôsobmi ovplyvňovať finančnú sieť

- **Odolnosť voči zlyhaniam** – sieť by nemala prestať fungovať aj napriek zlyhaniam niektorých uzlov v sieti, sieť je odolnejšia keď aj zlyhanie viacerých uzlov neovplyvní jej chod

1.3 Centralizovaný protokol a jeho problémy

Najčastejším a najjednoduchším riešením tohto problému je **centralizácia**. Systém navrhneme tak, že obsahuje jeden hlavný uzol, označme ho **A**. Tento uzol je autorita a udržiava stav celého systému. Teraz pokiaľ chce iný uzol zmeniť stav systému alebo uložiť dáta musí nadviazať spojenie s uzlom **A** a požiadať o zmenu stavu. Uzol **A** túto žiadosť zvaliduje (aby nebola porušená konzistentnosť, alebo ináč narušený chod systému) a tento stav uloží. Ostatné uzly už budú potom informované o novom stave pri akejkoľvek komunikácií s uzlom **A**. Uzol **A** je teda zárukou toho, že údaje v systéme sú korektné a systém vie stále vykonávať svoju úlohu. Nech sa teda útočník zmocní ľubovoľného uzla okrem **A**, alebo nech ľubovoľný uzol zlyhá, systém môže stále pokračovať v svojej úlohe. Obrovská nevýhoda však je, že práve uzol **A** je slabina celého systému. Keď zlyhá uzol **A** celý systém nie je schopný pokračovať. Ďalšou veľkou nevýhodou je, že všetky požiadavky systému musí validovať práve uzol **A** a teda môže dôjsť k jeho preťaženiu.

1.4 Známe decentralizované protokoly

Dôkaz prácou

Jedným z prvých pokusov o zlepšenie finančného systému je Bitcoin [12]. Bol to krok dobrým smerom, keďže ako prvá kryptomena dokázala fungovať na decentralizovanej myšlienke a teda sa vyhla používaniu jednej centrálnej autority.
Napriek tomu nedokázal vyriešiť všetky problémy popísané vyššie.
Je založený na myšlienke dôkazu prácou [3]. Veľkou nevýhodou je jeho neekologickosť, keďže plytvá zdrojmi. Existuje odhad z roku 2014, kedy Bitcoin skonzumoval toľko elektrickej energie ako celá krajina Írsko [13]. Totiž na schválenie transakcie musí schvaľovateľ vyriešiť ťažkú matematickú šifru, čo tiež má za následok pomalosť transakcií. Takisto nie každý sa môže stať schvaľovateľom, keďže schvaľovateľ musí disponovať veľkou výpočtovou silou aby dokázal vyriešiť matematické šifry.

Dôkaz bohatstvom

Jeho alternatívou je napríklad protokol dôkazu bohatstvom [7]. Tento protokol sa vyhýba problému protokolu dôkazu prácou, ktorého transakcie boli pomalé.
Jeden z najväčších problémov je takzvaný „nič v stávke“ problém (angl. nothing-at-stake). Keďže schvaľovanie transakcií je lacné (časovo aj výpočtovo), nič nebráni schvaľovateľom podporovať rôzne blokové reťaze (transakčné histórie).

Praktická byzantínska tolerancia voči chybám

Ďalším diametrálne odlišným prístupom je protokol *praktickej byzantínskej tolerancie voči chybám* [1]. Tento protokol na rozdiel od vyššie spomínaných nemá žiadne predpoklady na správanie sa nepriateľských uzlov (toto správanie označujeme ako byzantínske a uzly, ktoré sa tak správajú označujeme ako Byzantíncov). Takisto dosahuje zaručene nízku latenciu.

Nespĺňa však našu požiadavku o otvorenom členstve. Sieť sa spolieha na to, že útočník nemôže rýchlo a jednoducho pridať veľa uzlov do siete a tak sa zmocniť dostatočne veľkého množstva uzlov, čím by vedel ovplyvňovať hodnoty na ktorých sieť dosiahne konsenzus. Takýmto útokom sa tiež zvykne hovoriť *Sybil útoky*. A preto zaradenie uzla do siete musí prejsť schvaľovacím procesom. Vo všeobecnosti, siete používajúce tento protokol, nechávajú schvaľovanie členstva na centrálnej autorite.

Kapitola 2

Stellar konsenzus protokol

V tejto časti si predstavíme protokol [11], ktorý bude spĺňať všetky popísané požiadavky v kapitole vyššie a bude zaručené, že všetky správne fungujúce uzly sa dohodnú na rovnakej hodnote.

Každý uzol si bude udržiavať svoju kópiu blokovej reťaze. Jednu transakciu (a metadáta o nej) v blokovej reťazi budeme nazývať *blok*. Cieľom tohto konsenzus protokolu teda je aby sa celá sieť vždy dohodla na obsahu nového bloku a všetky uzly si ju mohli zapísať do svojej blokovej reťaze. Budeme tiež predpokladať, že každý uzol dodržujúci protokol nikdy nebude tvrdiť dve protichodné tvrdenia.

2.1 Kvórum a typy uzlov

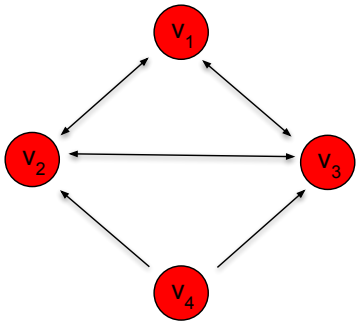
Keď sa uzol dozvie od dostatočného množstva iných uzlov informáciu, uverí jej a bude predpokladať, že žiadny uzol dodržujúci protokol nikdy nebude tvrdiť nič protichodné. Takúto množinu uzlov pre uzol v budeme nazývať kvórový rez. Keďže sa nám po čase môže stať, že niektoré uzly zlyhajú, dovolíme mať každému vrcholu viac kvórových rezov.

Stellar systém budeme nazývať dvojicu $< \mathbf{V}, \mathbf{Q} >$ kde $\mathbf{V}$ bude zoznam uzlov a $\mathbf{Q}$ bude funkcia priradzujúca uzlom *kvórové rezy*.
Presnejšie:

$$\mathbf{Q} : \mathbf{V} \to 2^{2^V} \setminus \emptyset \quad \forall v \in \mathbf{V}, \forall q \in \mathbf{Q}(v), v \in q$$

Teda každý vrchol obsahuje sám seba v každom svojom kvórovom reze. Kvórový rez bez vrcholu, ktorému prislúcha budeme nazývať *vlastný kvórový rez*.

Kvórum je množina uzlov $U \subseteq V$ v Stellar systéme práve vtedy keď $U \neq \emptyset$ a U obsahuje kvórový rez pre každý jeho prvok, teda $\forall v \in \mathbf{U}\ \exists q \in \mathbf{Q}(v)$, že $q \subseteq U$



$$\mathbf{Q}(v_4) = \{\{v_2, v_3, v_4\}\}$$
$$\mathbf{Q}(v_2) = \mathbf{Q}(v_3) = \mathbf{Q}(v_4) = \{\{v_1, v_2, v_3\}\}$$

Obr. 2.1: Príklad volenia kvórových rezov od uzlov

Kvórum je množina uzlov dostatočná na dosiahnutie konsenzu. Totiž táto množina obsahuje pre každý svoj prvok kvórový rez, ktorý ho dokáže presvedčiť. Na obrázku 2.1 je znázornená jedna možnosť ako si uzly mohli navoliť kvórové rezy. Všimnime si, že napríklad množina $\{v_2, v_3, v_4\}$ je síce kvórový rez uzlu v_4, ale kvórum netvorí, keďže neobsahuje jediný kvórový rez v_2. Naopak množina uzlov $\{v_1, v_2, v_3\}$ je kvórový rez uzlov v_1, v_2, v_3 a teda aj tvorí kvórum.

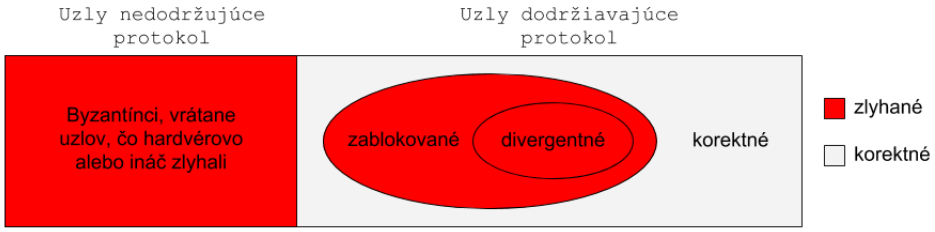
Uzly budeme rozdeľovať do dvoch hlavných skupín – na uzly držiacie sa protokolu a nedržiace sa protokolu. Medzi uzly nedodržiavajúce protokol zaraďujeme buď Byzantíncov (napríklad nepriateľ sa zmocnil uzlu) alebo uzly, ktoré akýmkoľvek spôsobom zlyhali. Cieľom protokolu je aby uzly dodržujúce protokol dosiahli konsenzus. Toto vieme popísať dvomi vlastnosťami.

Bezpečnosť Množina uzlov v Stellar systéme je *bezpečná*, ak žiadne dva uzly z nich neschvália pre jeden konkrétny blok iné hodnoty. Teda nebudú mať rôzne verzie transakčnej histórie.

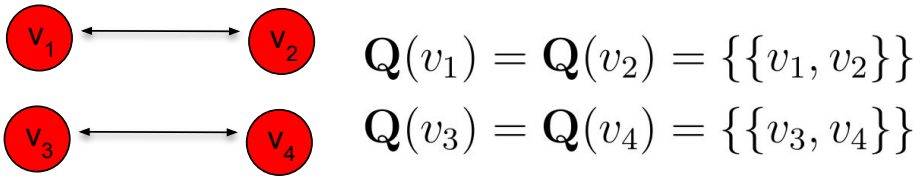
Životaschopnosť Množina uzlov v Stellar systéme je *životaschopná*, ak vie schvaľovať nové bloky do svojej blokovej reťaze, bez potreby spolupráce zlyhaných uzlov. Rovnako vieme definovať *životaschopný* uzol.

Obe tieto definície sú len popisom vlastností, ktoré chceme dosiahnuť. Neskôr v práci si aj zadefinujeme ako vieme zistiť, kedy *Stellar systém* tieto vlastnosti má, už len z popisu samotného systému.

Teraz uzly dodržujúce protokol rozdelíme do troch skupín. Uzly, ktoré budú porušovať podmienku bezpečnosti budeme označovať ako *divergentné*. Všimnime si, že toto je veľmi ťažké exaktne definovať, keďže nevieme, ktorá bloková reťaz, je tá správna a ktorá divergovala (každý uzol si totiž drží vlastnú kópiu a povedať objektívnu pravdu je náročné). Avšak tento pojem budeme potrebovať len na intuitívne pochopenie fun-



Obr. 2.2: Rozdelenie uzlov do kategórie podľa schopnosti zúčastňovať sa na konsenze



Obr. 2.3: Sieť bez prieniku kvór môže divergovať

govania siete a teda s predstavou, že *divergentný* uzol, je taký uzol, ktorý schválil niečo iné ako tie „dobré“ uzly, si plne vystačíme.

Uzly, ktoré síce sú bezpečné ale nie sú životaschopné zase budeme označovať ako *zablokované*. A nakoniec uzly, ktoré sú aj bezpečné aj životaschopné budeme označovať ako *korektné*.

Všimnime si, že aj uzly dodržujúce protokol môžu byť divergentné, stačí, že hlasovaním spolu s Byzantíncami sa im poškodila bloková reťaz. Práve takýmto situáciám sa bude Stellar protokol snažiť vyhnúť. Vizualizáciu katogorizácie uzlov môžeme vidieť na obrázku 2.2.

2.2 Vlastnosti kvór a uzlov

Keby si uzly určovali svoje kvóra úplne ľubovoľne, sieť by vôbec nemusela byť prepojená a teda by mohla byť divergentná. Preto budeme požadovať aby kvóra spĺňali niektoré vlastnosti.

Prienik kvór Stellar systém má prienik kvór práve vtedy ak každá dvojica kvór má aspoň jeden spoločný uzol.

Na obrázku 2.3 môžeme vidieť príklad siete bez prieniku kvór. Disjunktné množiny $\{v_1, v_2\}$ a $\{v_3, v_4\}$ sú kvóra bez prieniku. Takže tieto množiny sú samé o sebe schopné schvaľovať bloky a predlžovať blokovú reťaz. Keďže ale nemajú prienik nemusia schváliť rovnaký blok a teda sieť bude divergovať.

Teraz si definujme operáciu mazania množiny uzlov zo Stellar systému.

Operácia zmaž Ak máme Stellar systém $< \mathbf{V}, \mathbf{Q} >$ a zmažeme z neho množinu uzlov $B \subseteq V$, dostaneme systém $< \mathbf{V}, \mathbf{Q} >^B = < \mathbf{V} \setminus B, \mathbf{Q}^B >$, kde $\mathbf{Q}^B = \{q \setminus B \,|\, q \in \mathbf{Q}(v)\}$.

Takto si ľahko vieme zúžiť systém na množinu uzlov, ktoré chceme ďalej uvažovať a pomocou toho analyzovať odolnosť siete aj po prehlásení niektorých uzlov za nedôveryhodné. Ako sme si už hovorili tak aby bolo kvórum schopné dosiahnuť konsenzus a budovať rovnakú blokovú reťaz, musí mať prienik kvór. Avšak ak je tento prienik práve byzantínsky uzol, tak naša sieť bude aj tak divergentná, keďže práve on môže poslať správu jednému kvóru že hlasoval *za* schválenie nového bloku a druhému poslať správu, že hlasoval *proti* a oba môžu schváliť protichodné tvrdenia. Preto aby naša sieť bola odolná aj voči zlyhaniam niektorých uzlov, bude musieť mať väčšie prieniky kvór. Pre lepšie popísanie tejto problematiky si dodefinujeme *nepotrebné množiny*. Budeme chcieť mať sieť rozloženú tak, aby všetky zlyhané uzly boli v nepotrebnej množine a teda sieť bola schopná prijímať konsenzus aj bez nich.

Nepotrebná množina Ak máme Stellar systém $< \mathbf{V}, \mathbf{Q} >$, tak množina uzlov B je nepotrebná práve vtedy keď

- $< \mathbf{V}, \mathbf{Q} >^B$ má prienik kvór
- $\mathbf{V} \setminus B$ je $\emptyset$ alebo kvórum v $< \mathbf{V}, \mathbf{Q} >$

Prvá podmienka chráni sieť pred divergenciou, ktorú by mohli spôsobovať uzly z B, tým že by schvaľovali protichodné tvrdenia. Tým, že prienik kvór ostane zachovaný aj keď uzly z B uvažovať nebudeme, sieť naozaj divergovať nebude. Ak sieť danú podmienku spĺňa zvykneme hovoriť, že sieť má prienik kvór aj *napriek množine B*.

Naopak druhá podmienka zachováva životaschopnosť siete bez ohľadu na uzly z B, keďže zaručuje, že aj bez uzlov z B vedia ostatné uzly schvaľovať nové bloky. Splneniu tejto podmienky zvykneme hovoriť životaschopnosť aj *napriek množine B*.

Čo si môžeme všimnúť je, že uzly musia balansovať vo veľkosti vyberaných kvórových rezov. Keď si totiž uzly vyberú veľké kvórové rezy, tak vzniknú veľké kvóra. Toto potom vedie k veľkým prienikom kvór (musí zlyhať veľa uzlov na zničenie prieniku kvór), čo nám zaručuje malú šancu nepriateľa zničiť bezpečnosť siete. Avšak oveľa ľahšie ohrozí jej životaschopnosť, keďže zlyhanie už len málo uzlov ovplyvní veľa kvór. Podobne naopak pri malých kvórach vzniknú menšie kvórové prieniky, čo zjednoduší nepriateľovi ohroziť bezpečnosť siete, avšak bude musieť zlyhať viac uzlov aby to ohrozilo životaschopnosť siete (aby zlyhali uzly v čo najviac kvórach). Uzly v sieti teda musia nájsť balans pre čo najväčšiu obranyschopnosť siete.

Na záver tejto podkapitoly sme už teda pripravený si definovať *bezpečnosť* a *životaschopnosť* siete na základe vlastností konkrétneho *Stellar systému*, aby zároveň boli splnené definície a očakávania spomenuté na začiatku kapitoly.

Bezpečnosť Stellar systém je *bezpečný* práve vtedy keď má prienik kvór.

Životaschopnosť Množina uzlov A v Stellar systéme $< \mathbf{V}, \mathbf{Q} >$ je *životaschopná*, práve vtedy, keď tento Stellar systém bude životaschopný aj *napriek množine* $V \setminus A$.

Vieme si zadefinovať aj *životaschopnosť* uzlu v v Stellar systéme voči množine B. A to tak, že bude musieť mať mať aspoň jeden kvórový rez neobsahujúci uzol z množiny B (teda bude schopný byť súčasťou niektorého kvóra bez zlyhaných uzlov).

Táto definícia dáva zmysel, keďže ak uzol v nebude životaschopný voči množine B, tak už žiadna množina uzlov obsahujúcich v neobsahujúca žiadny uzol z B nemôže byť životaschopná. Keďže na uzavretie kvóra potrebujeme aj kvórový rez uzlu v.

2.3 Priebeh schvaľovania bloku

Schvaľovanie nového bloku prebieha vo viacerých fázach. Počas tohto procesu sa z pohľadu každého uzlu môže blok nachádzať v 3 rôznych stavoch - *neznámy*, *akceptovaný*, *potvrdený*.

Na začiatku schvaľovacieho procesu je pre každý uzol blok *neznámy*. Počas celého priebehu schvaľovania si uzly vymieňajú správy, na základe ktorých si môže blok zvýšiť stav bloku (pre daný uzol môže ísť blok len do vyššieho stavu, teda ak už pre neho bol blok *akceptovaný*, nemôže ho blok prehlásiť spätne za *neznámy*). Správy si môžeme predstavovať ako hlasovanie uzlov, či má alebo nemá byť blok pridaný do blokovej reťaze. Keď pre každý korektný uzol bude blok *potvrdený*, môžu si ho pridať do svojej blokovej reťaze.

Uzly si budú meniť stavy bloku na základe správ od uzlov zo svojich kvórových rezov spomínaných vyššie, avšak detajly presných pravidiel na základe ktorých sa tieto stavy budú meniť využívať nebudeme. Preto pre potreby našej práce nepotrebujeme vedieť ako a aké správy si medzi sebou uzly budú vymieňať. Plnohodnotne nám bude stačiť intuícia zavedenia kvórových rezov a kvór.

Presnejšie detajly Stellar konsenzus protokolu, procesu schvaľovania bloku, samotné dôkazy, že takéto schvaľovanie vedie ku v praxi plne funkčnej sieti a napokon aj samotnú implementáciu môže čitateľ nájsť v [11].

Na záver kapitoly si zhrňme, že naozaj tento protokol spĺňa očakávania predstavené v podkapitole 1.2.

- **Jedna globálna** – naša sieť sa naozaj nedelí na žiadne podsiete a každý uzol môže komunikovať s každým
- **Decentralizovaná** – každý uzol si drží svoju kópiu blokovej reťaze a žiadna centrálna kópia neexistuje
- **Nízka latencia** – uzly dokážu schváliť nový blok len po vymenení si niekoľkých správ, uzly nepotrebujú robiť žiadnu výpočtovo náročnú prácu
- **Otvorené členstvo v sieti** – pridať sa do siete vie ktokoľvek, stačí sa pripojiť a nastaviť kvórové rezy
- **Otvorené členstvo medzi schvaľovateľmi** – „schvaľovateľmi“ sú vlastne uzly, ktoré sa nachádzajú v niekoho kvórovom reze. Stačí si teda vybudovať dôveru medzi ostatnými členmi siete, samotné pravidlá siete nebránia byť schvaľovateľom
- **Flexibilná dôvera** – každý uzol si určuje sám svoje kvórové rezy a teda sám určuje komu bude dôverovať
- **Odolnosť voči zlyhaniam** – ako sme si ukázali toto záleží už na samotnom systéme, ale pokiaľ si budú uzly rozumne voliť kvórové rezy, bude mať sieť aj túto vlastnosť. Hodnoteniu odolnosti jednotlivých sietí sa budeme venovať vo zvyšku našej práce

Kapitola 3

Metriky odolnosti siete

V tejto kapitole si predstavíme našu metriku podľa ktorej budeme vyhodnocovať, ktorá sieť je bezpečnejšia a životaschopnejšia na základe toho ako si uzly zvolili kvórové rezy. Ukážeme však, že spočítať tieto metriky na všeobecných sieťach sú ťažké problémy a spočítať ich rýchlo na veľkých sieťach je nemožné. Preto si predstavíme aj jeden špecifickejší typ siete, ktorý budeme v ďalších kapitolách používať.

Bezpečnostný koeficient Stellar systém $< \mathbf{V}, \mathbf{Q} >$ má *bezpečnostný koeficient* rovný k, kde k je najmenšie nezáporné číslo také, že existuje k prvková množina uzlov $B \in \mathbf{V}$ pre ktorú platí, že $< \mathbf{V}, \mathbf{Q} >^B$ nie je *bezpečný*.
Ak také k neexistuje (systém má prienik kvór nech odoberieme ľubovoľnú množinu uzlov), tak si tento koeficient dodefinuje ako $|\mathbf{V}|$.

Inými slovami: v sieti musí zlyhať aspoň k uzlov aby sieť prestala byť *bezpečná*. Teda existuje k prvková množina uzlov, ktorá nie je *nepotrebná*, konkrétne porušuje prvú podmienku *nepotrebnej množiny*. Naopak sieť dokáže prežiť aj napriek zlyhaniu $k - 1$ ľubovoľných uzlov a ani jeden uzol dodržujúci protokol nebude divergovať. Samozrejme máme tu špeciálny prípad keď $k = 0$, vtedy už samotná sieť nie je *bezpečná*. V prípadoch keď je sieť dokonale bezpečná, teda odolná voči zlyhaniu ľubovoľnej množiny uzlov si dodefinujeme koeficient na počet uzlov v systéme, keďže ak už zlyhajú všetky uzly nemá žiadny ďalší zmysel merať bezpečnosť siete.

Podobne si vieme definovať koeficient pre životaschopnosť.

Koeficient životaschopnosti Stellar systém $< \mathbf{V}, \mathbf{Q} >$, má *koeficient životaschopnosti* rovný k, kde k je najmenšie prirodzené číslo také, že existuje k prvková množina uzlov $B \in \mathbf{V}$ pre ktorú platí, že množina uzlov $\mathbf{V} \setminus B$ v tomto Stellar systéme nie je *životaschopná*.

V sieti musí zlyhať aspoň k uzlov aby sieť prestala byť *životaschopná*. Teda existuje

k prvková množina uzlov, ktorá nie je *nepotrebná*, konkrétne porušuje druhú podmienku *nepotrebnej množiny*. Naopak sieť dokáže prežiť aj napriek zlyhaniu $k-1$ ľubovoľných uzlov a všetky uzly sa budú schopné ďalej podieľať na prijímaní konsenzu. Špeciálny prípad $k = 0$ zrejme nemôže nastať, keďže $\mathbf{V} \setminus \emptyset = \mathbf{V}$ je kvórum v $< \mathbf{V}, \mathbf{Q} >$.

Na záver si zadefinujeme koeficient, ktorý nám bude zaručovať obe podmienky spomenuté vyššie. To čo si totiž môžeme uvedomiť je, že pokiaľ náš *bezpečnostný koeficient* je k, tak síce žiadne uzly dodržujúce protokol sme „nestratili“ v zmysle, že neschvaľujú protichodné bloky aj keď teda $k-1$ uzlov zlyhalo. Napriek tomu táto množina $k-1$ uzlov nemusí byť nepotrebná, keďže môže porušovať druhú podmienku a teda niektoré uzly mohli prestať byť schopné podieľať sa na schvaľovaní blokov. Podobne to funguje pri *koeficiente životaschopnosti*.

Koeficient odolnosti Stellar systém $< \mathbf{V}, \mathbf{Q} >$, má koeficient odolnosti rovný minimu z jeho *bezpečnostného koeficientu* a *koeficientu životaschopnosti*.

Všimnime si, že pokiaľ sieť obsahuje menej zlyhaných uzlov ako je jej *koeficient odolnosti*, tak to nijak neovplyňuje chod samotnej siete a ani jedného uzla dodržujúceho protokol. Keďže každá takáto množina je *nepotrebná*. Po zlyhaní ďalšieho uzla už ale žiadne garancie nie sú, systém môže byť narušený aj nemusí, keďže tiež záleží, ktoré konkrétne uzly zlyhajú.

3.1 Životaschopnosť

Skúsme sa najprv zamerať na vyhodnocovanie metriky životaschopnosti siete. V tomto prípade hľadáme najmenšiu (počtom prvkov) množinu uzlov B takú, že systém už nebude životaschopný aj *napriek množine* B. Teda, že bez uzlov z B už ostatné uzly nebudú tvoriť kvórum v pôvodnom systéme. Z definície kvóra vieme ale túto úlohu preformulovať. Kvórum musí obsahovať aspoň jeden kvórový rez každého jeho prvku. Je to teda ekvivalentné nájdeniu takého B, že niektorý uzol dodržujúci protokol bude mať v každom *vlastnom kvórovom reze* aspoň jeden prvok B. Vtedy budeme hovoriť, že tento uzol prestal byť *životaschopný*. Nájsť takéto B môžme ale pre každý vrchol zvlášť a potom pre sieť stačí vybrať minimálne B. Veľkosť takéhoto minimálneho B bude potom náš *koeficient životaschopnosti*.

Chceme teda nájsť takú najmenšiu množinu uzlov B, že pre uzol u a pre každý jeho *vlastný kvórový rez* platí, že aspoň jeden jeho prvok je z B. Toto je však známy problém pokrytia množín (angl. *set cover problem*) [6], kde prvky sú všetky uzly nachádzajúce sa v ľubovoľnom vlastnom kvórovom reze u. Množiny ktoré chceme pokryť sú vlastné kvórové rezy a teda hľadáme najmenšiu množinu, ktorá tieto množiny pokryje. O tomto probléme je známe, že je NP-úplný, čo nám teda hovorí, že tento koeficient

vo všeobecnosti nebudeme vedieť zistiť v polynomiálnom čase (ak $P \neq NP$), aj keď na tento problém existujú rôzne rýchle a rôzne dobré aproximačné algoritmy, ktoré by nám pomohli aspoň približne určiť koľko zlyhaným uzlov naša sieť prežije.

Ďalšou veľkou nevýhodou úplne ľubovoľných kvórových rezov je, že už len popísanie týchto kvórových rezov môže byť exponenciálne veľké (od počtu uzlov), čo nám výrazne ovplyvňuje výpočtovú silu. Preto mierne obmedzíme výber kvórových rezov aby ich popis dokázal byť dostatočne krátky a zároveň dosť voľný (nenútil uzlov veriť uzlom, ktorým nechcú). Na výmenu budeme o týchto sieťach vedieť lepšie a rýchlejšie uvažovať.

Poďme sa teda radšej pozrieť na siete s konkrétnejšími kvórovými rezmi, pri ktorých to budeme vedieť určiť.

NQK-siete budeme nazývať siete s N uzlami, kde každý uzol u si zvolí kvórové rezy tak, že si zvolí Q_u uzlov ktorým verí a jeho vlastné kvórové rezy budú všetky K_u prvkové podmnožiny týchto Q_u uzlov[1]. Samozrejme máme prirodzené predpoklady na sieť, teda $Q_u < N$ a $1 \leq K_u \leq Q_u$

V takýchto sieťach je ale jednoduché zistiť *koeficient životaschopnosti*.

Veta Koeficient životaschopnosti *NQK-siete* s množinou uzlov V je $\min_{u \in V}(Q_u - K_u + 1)$

Dôkaz Z úvodu podkapitoly vieme, že stačí nájsť pre každý uzol najmenšiu množinu, ktorá má aspoň jeden prvok v každom *vlastnom kvórovom reze* tohto uzla. Zoberme si teda uzol u. A označme si množinu uzlov nachádzajúcich sa v aspoň jednom jeho *vlastnom kvórovom reze* ako Q ($|Q| = Q_u$). Pokiaľ zrejme zlyhá ľubovoľných $Q_u - K_u$ uzlov z Q, tak stále máme aspoň jeden nepokrytý *vlastný kvórový rez*, keďže v *NQK-sieti* každá K_u prvková podmnožina Q je *vlastný kvórový rez*.

Naopak ak zlyhá $Q_u - K_u + 1$ uzlov už neostane žiadna K prvková podmnožina Q (vlastný kvórový rez), ktorá by neobsahovala zlyhaný uzol. Podľa úvodu podkapitoly teda máme, že koeficient životaschopnosti siete bude minimum zo všetkých týchto výsledkov pre všetky uzly u. ■

3.2 Bezpečnosť

Tentokrát sa pozrieme na metriku bezpečnosti siete. V tomto prípade sa snažíme nájsť najmenšiu množinu, ktorá poruší prienik kvór v systéme. Čo sa dá preformulovať na problém hľadania najmenšieho prieniku dvoch ľubovoľných kvór. Veľkosť tohoto prieniku bude náš *bezpečnostný koeficient* (pokiaľ v sieti existujú aspoň 2 kvóra, ináč na-

[1] Kvórové rezy vyrobíme z vlastných kvórových rezov tak, že ku každému z nich pridáme uzol, ktorému tieto kvórové rezy patria

stane špeciálny prípad z definície a koeficient bude počet uzlov v sieti). Totiž stačí uvažovať práve množinu uzlov obsahujúcu práve tento prienik a tieto dve kvóra po odstránení tejto množiny už prienik mať nebudú a teda celá sieť o túto vlastnosť príde. Keďže je to ale najmenší prienik, tak uvažovaním množiny menšieho počtu prvkov nikdy neobsiahne celý prienik a teda aj po ich odstránení všetky prieniky ostanú.

Dokonca platí trošku príjemnejšie pravidlo, ak si zadefinujeme nasledovný pojem.

Minimálne kvórum je také kvórum, že žiadna jeho vlastná podmnožina už nie je kvórum.

Bezpečnostný koeficient je dokonca veľkosť najmenšieho prieniku dvoch ľubovoľných minimálnych kvór, keďže ak máme kvóra $Q_3 \subseteq Q_1$ a $Q_4 \subseteq Q_2$, tak zjavne $|Q_3 \cap Q_4| \leq |Q_1 \cap Q_2|$.

Najprv si ukážeme, že pokiaľ nedáme žiadne obmedzenia na výber kvórových rezov, tak zistenie nášho bezpečnostného koeficientu bude NP-ťažký problém, ba čo viac dokonca len zistenie či náš koeficient je aspoň 1 bude NP-ťažké. Dokážeme to redukciou nášho problému na známy NP problém, ktorým bude SAT [2].

SAT je rozhodovací problém. Na vstupe má booleovskú formulu v konjuktívnej normálnej forme a úlohou algoritmu je rozhodnúť, či existuje ohodnotenie premenných vo formule, pri ktorom je vstupná formula splnená.

Veta Ak by sme vedeli zistiť v polynomiálnom čase pre daný Stellar systém, či má bezpečnostný koeficient 0 (či existujú kvóra bez prieniku), tak by sme vedeli v polynomiálnom čase zistiť či má SAT riešenie.

Dôkaz Majme teda na vstupe booleovskú formulu $X_1 \wedge X_2 ... \wedge X_m$, kde každé $X_i = z_{i,1} \vee z_{i,2} ... \vee z_{i,p_i}$, kde $z_{i,j} \in \{x_1, ..., x_n, \neg x_1, ..., \neg x_n\}$, teda máme konjuktívnu normálnu formu z premenných x_i a $\neg x_i$.

Teraz zostrojíme Stellar systém $< \mathbf{V}, \mathbf{Q} >$, taký, že táto formula je splniteľná práve vtedy keď zostrojený systém bude mať bezpečnostný koeficient 0. Množina uzlov bude $V = \{v_1, ..., v_{2n+1}, v_{1,1}, ..., v_{2n+1,m}, w_{1,1}, ..., w_{2n+1,n}\}$.

Aby sme nemuseli rozlišovať prípady premenných v pozitívnom a negatívnom kontexte, tak $v_1, ..., v_n$ budú zodpovedať premenným v pozitívnom kontexte (x_i) a uzly $v_{n+1}, ..., v_{2n}$ premenným v negatívnom kontexte ($\neg x_{i-n}$) a napokon tu budeme mať špeciálny uzol v_{2n+1}.

Potom budeme mať uzly $v_{i,j}$, ktoré budú symbolizovať klauzulu X_j a pre každý uzol (v_k) budeme mať kópiu (teda dokopy $2n + 1$ kópií). Prečo potrebujeme pre každý uzol (v_k) vlastnú kópiu uzla pre každú klauzulu, uvidíme pri zostrojovaní kvórovej funkcie.

Samotnú redukciu by sme dokázali spraviť aj bez nich, no potrebovali by sme príliš veľa kvórových rezov a redukcia by už nebola polynomiálna.

Napokon nám ostali špeciálne uzly $w_{i,j}$, ktorými sa budeme snažiť zabezpečiť aby splniteľnosť formuly naozaj zodpovedala našej sieti a budeme ho používať na to aby sme zabezpečili nejaký vzťah medzi x_j a $\neg x_j$ a zas budeme mať pre každý uzol reprezentujúci pôvodnú premennú vlastnú kópiu.

Kvórovú funkciu vracajúcu *vlastné kvórové rezy* zostrojíme nasledovne:

$$Q(v_i) = \{\{v_{i,1}, ..., v_{i,m}, w_{i,1}, ..., w_{i,n}\}\}$$
$$Q(v_{i,j}) = \{\{v_k\} | x_k \in X_j\} \cup \{\{v_{k+n}\} | \neg x_k \in X_j\} \cup \{\{v_{2n+1}\}\}$$
$$Q(w_{i,j}) = \{\{v_j\}, \{v_{j+n}\}\}$$

To znamená, že všetky uzly v_i majú jediný vlastný kvórový rez a to množinu všetkých svojich zodpovedajúcich kópií všetkých klauzúl (plus špeciálne uzly $w_{i,j}$). Naopak všetky uzly zodpovedajúce niektorej inštancii klauzuly majú 1-prvkové vlastné kvórové rezy. V nich uzly, ktoré zodpovedajú premenným (výskyt premennej v klauzule sme zapísali symbolicky množinovým zápisom), ktoré sa v klauzule nachádzajú (a špeciálny vrchol v_{2n+1}). Na záver špeciálne uzly $w_{i,j}$ majú len 2 vlastné kvórové rezy a to uzly reprezentujúce premennú x_j a jej negáciu (zase ako pre klauzuly máme $2n + 1$ kópií pre každú premennú).

Sieť už máme skonštruovanú a zjavne táto redukcia sa dá spraviť polynomiálne, keďže uzlov máme $(2n + 1)(m + n + 1)$ a každý uzol má najviac $n + 1$ kvórových rezov (presnejšie počet premenných v klauzule $+1$) a každý kvórový rez má veľkosť najviac m. Teraz ostáva ukázať, že naozaj v tejto sieti existujú 2 kvóra bez prienikov práve vtedy keď je booleovská formula splniteľná.

Najprv ukážeme, že ak booleovská formula je splniteľná, tak existujú 2 kvóra bez prieniku. Uvažujme jedno vhodné ohodnotenie premenných. Vytvorme si množinu $A = \{i | x_i = 1\} \cup \{n+i | x_i = 0\}$. A je vlastne množina indexov uzlov, ktoré reprezentujú kladne ohodnotené premenné v našom ohodnotení. Označme A' ako doplnok A do množiny všetkých platných indexov, teda $A' = \{1, 2, ..., 2n + 1\} \setminus A$.

Potom

$$\mathbf{Q_1} = \{v_a | a \in A\} \cup \{v_{a,j} | a \in A\} \cup \{w_{a,j} | a \in A\}$$

je kvórum.

Zrejme v_a majú svoj jediný kvórový rez v Q_1 ($v_{a,j}, w_{a,j}$ pre všetky j). Podobne aj $w_{a,j}$ tu má kvórový rez, keďže Q_1 obsahuje práve jeden prvok z dvojice v_i a v_{n+i} (buď je x_i ohodnotené 0 alebo 1). A napokon aj kvórový rez $v_{a,j}$ je v Q_1, keďže ak je formula splnená aspoň jeden literál klauzuly X_j musí byť pravdivý a naša množina obsahuje všetky uzly zodpovedajúce pravdivým literálom (ak bolo x_i pravdivé obsahujeme v_i, ináč obsahujeme v_{n+i}). Teda Q_1 je naozaj kvórum.

Ukážeme, že aj

$$\mathbf{Q_2} = \{v_a | a \in A'\} \cup \{v_{a,j} | a \in A'\} \cup \{w_{a,j} | a \in A'\}$$

je kvórum.

Podobne ako v Q_1, množina obsahuje aspoň jeden kvórový rez uzlov v_a a $w_{a,j}$. Keďže $v_{2n+1} \in Q_2$ a takisto aj $v_{2n+1} \in Q(v_{a,j})$, tak aj Q_2 je kvórum. Tieto kvóra ale nemajú prienik, keďže A a A' nemajú.

Ostáva teda ukázať, že ak v našom systéme existujú 2 kvóra Q_1 a Q_2 bez prieniku, tak naša formula je splniteľná.

Poskupinkujme všetky uzly podľa prvého indexu, tieto skupinky označme

$$\mathbf{V_i} = \{v_i, v_{i,1}, ..., v_{i,m}, w_{i,1}, ..., w_{i,n}\}$$

.

Všimnime si, že ak je v nejakom kvóre v_i, tak potom aj celé V_i (v_i má jediný kvórový rez).

Naopak ak je v kvóre iný uzol tejto skupiny a v_i v kvóre nie je, tak aj po odobratí tohto uzlu z kvóra, sa kvórum zachová, keďže v_i je jediný uzol, ktorý má ostatné uzly v skupine vo svojom kvórovom reze.

Preto pre jednoduchosť uvažujme kvóra Q_1 a Q_2, ktoré obsahujú iba celé skupiny V_i. Toto môžeme spraviť keďže ako sme ukázali vyššie, stačí skúmať prieniky minimálnych kvór.

Bez ujmy na všeobecnosti nech Q_1 je kvórum neobsahujúce špeciálnu skupinu V_{2n+1} (keďže sú bez prieniku jedno z nich túto vlastnosť má).

Ohodnoťme premenné vo formule nasledovne:

$$x_i = \begin{cases} 1 & V_i \subseteq Q_1 \\ 0 & V_{i+n} \subseteq Q_1 \end{cases}$$

Všimnime si, že každá premenná je definovaná, keďže každé kvórum obsahuje skupinu V_i alebo V_{i+n}. Totiž, každá skupina (V_j) obsahuje uzol $w_{j,i}$, ktorej jediné kvórové rezy sú $\{v_i\}$ a $\{v_{i+n}\}$ a teda potom už nutne kvórum obsahuje celú skupinu V_i alebo celú skupinu V_{i+n}.

Avšak Q_1 a Q_2 nemajú prienik a keďže obe obsahujú aspoň jednu zo skupín V_i a V_{i+n}, tak každé z nich obsahuje dokonca práve jednu skupinu a teda je ohodnotenie premenných aj dobre definované.

Na záver ukážeme, že formula je splnená. Keďže $Q_1 \neq \emptyset$, tak obsahuje skupinu V_i pre nejaké i a teda aj uzly $v_{i,1}, ..., v_{i,m}$. Tieto uzly ale majú také kvórové rezy, že naše kvórum musí obsahovať aj uzly z každej klauzuly $X_1, ..., X_m$. Teda naozaj ak sú v našom systéme 2 kvóra bez prieniku pôvodná formula je splniteľná. ■

Počas písania našej práce bol do predtlače publikovaný článok[8], ktorý dokázal rovnaký výsledok. Dokonca ho na záver jednoduchou úvahou zosilnil na tvrdenie, že už len zistiť či bezpečnostný koeficient siete je 0 v sieťach s najviac 2 kvórovými rezmi a 2 prvkami v kvórovom reze je NP-ťažké.

Môžeme si všimnúť, že naša redukcia dokonca používala *NQK-sieť* definovanú v predošlej podkapitole. Preto nebudeme vedieť rýchlo zisťovať tento koeficient ani v týchto sieťach aj keď koeficient životaschopnosti zistiť vieme. Budeme sa teda snažiť ešte viac spresniť pravidlá na výber kvórových rezov, aby sme tento koeficient vedeli presne vyrátať a tak vedeli aj niečo o danej sieti povedať.

Kapitola 4

Niektoré rýchlo merateľné siete

Vzhľadom na to, že sme si v minulej kapitole dokázali, že vo všeobecnosti je naše metriky ťažké vypočítať, v tejto kapitole sa skúsime pozrieť na niektoré konkrétne typy sietí. Pozrieme sa na siete za predpokladu, že by si uzly volili kvórové rezy určitým spôsobom a skúsime sa zamyslieť ako veľmi by boli takéto siete bezpečné. Po analýze niektorých typov dostaneme lepší náhľad do tvorenia kvór a kvórových rezov a budeme vedieť popísať niektoré osvedčené postupy na tvorenie kvórových rezov, tak aby bola odolnosť siete vysoká.

4.1 Reťazové siete

Najprv si ešte upravíme a zjednodušíme definíciu sietí, ktorými sa budeme zaoberať.

Zjednodušené NQK-siete sú *NQK-siete*, pre ktoré platí $\forall v : Q_v = Q \wedge K_v = K$.

Reťazové siete budú zjednodušené NQK-siete s nasledujúcou voľbou vlastných kvórových rezov:

$$Q(v_i) = \begin{cases} [\{v_0, ..., v_Q\} \setminus \{v_i\}]^K & i < Q \\ [\{v_{i-Q}, ..., v_{i-1}\}]^K & inak \end{cases}$$

Pričom uzly sme si očíslovali: $v_0, ..., v_{N-1}$. Teda vlastné kvórové rezy uzlov sú ľubovoľné K-prvkové podmnožiny Q uzlov pred ním (v zoradení podľa indexov) až na uzly, ktoré nemajú Q uzlov pred sebou, tie použijú prvých Q uzlov v zoradení (okrem seba, aby to bol vlastný kvórový rez).

Všimnime si, že tým, že všetky uzly „závisia“ na uzloch pred nimi, tak každé kvórum v sieti bude musieť obsahovať uzly s nízkym indexom. Formálnejšie: každý vrchol v kvóre musí mať v kvóre aj jeden zo svojich kvórových rezov a teda ak v každom svojom kvórovom reze má uzol s menším indexom, tak tento vrchol nemôže byť v žiadom kvóre uzol s najmenším indexom.

V našej sieti, ale jediné uzly s kvórovým rezom neobsahujúcim menšie indexy sú uzly $v_0, ..., v_Q$, keď sa lepšie zamyslíme dokonca to budú len uzly $v_0, ..., v_{Q-K}$, keďže každý kvórový rez obsahuje $K+1$ uzlov. Teda ak si označíme uzol s najmenším indexom v kvóre ako v_i, tak platí $i \leq Q-K$.

Všimnime si ale, že všetky tieto uzly majú kvórové rezy v množine $[\{v_0, ..., v_Q\}]^{K+1}$. Čo teda znamená, že každé kvórum reťazovej siete obsahuje ako podmnožinu niektorú $(K+1)$-prvkovú podmnožinu $\{v_0, ..., v_Q\}$. Naopak zoberme si jednu $(K+1)$-prvkovú podmnožina $\{v_0, ..., v_Q\}$ a označme ju A. Všetky prvky A majú z definície kvórové rezy presne tieto $(K+1)$-prvkové podmnožiny, a teda jeden kvórový rez pre každý prvok je samotná množina A. Teda dokonca A je kvórum.

Ukázali sme teda, že všetky minimálne kvóra v reťazovej siete sú $[\{v_0, ..., v_Q\}]^{K+1}$. Ako sme už dokázali v kapitole 3.2 bezpečnostný koeficient je veľkosť ich najmenšieho prieniku.

Máme teda $(K+1)$-prvkové podmnožiny $(Q+1)$-prvkovej množiny a hľadáme ich najmenší prienik z čoho dostaneme, že bezpečnostný koeficient reťazovej siete je $max(0, 2K-Q+1)$. Keďže reťazová sieť je špeciálnym typom NQK-siete, tak vieme aj koeficient životaschopnosti a to je $Q-K+1$.

Teraz vieme povedať sieti ako zvoliť K aby jej koeficient odolnosti bol čo najväčší, teda hľadáme $\max\limits_{\forall K} \min(2K-Q+1, Q-K+1)$. [1]

Táto funkcia nadobúda maximum pre $K = \frac{2}{3}Q$. A teda najodolnejšie reťazové siete majú koeficient odolnosti rovný $\frac{Q}{3}+1$. Teda optimum je zvoliť počet uzlov vo vlastných kvórových rezov rovný dvoch tretín počtu uzlom ktorým vrchol dôveruje.

4.2 Cyklické siete

Ako bežne, uzly si očíslujme $v_0, ..., v_{N-1}$. Keďže siete budú cyklické, často budeme používať pojem po sebe idúce uzly, a takými budeme uvažovať aj uzly v_0 a v_{N-1}. Ďalej si zadefinujeme aj intervaly uzlov, ktoré budeme označovať ako

$$< v_a, v_b >= \begin{cases} \{v_a, ..., v_b\} & a \leq b \\ \{v_a, ..., v_{N-1}, v_0, ...v_b\} & inak \end{cases}$$

Podobne označíme aj (polo)otvorené intervaly, kde len nebudeme uvažovať hraničné uzly (tieto môžu byť potencionálne aj prázdne).

Cyklické siete budeme označovať *zjednodušené NQK-siete*, ktorých vlastné kvórové rezy budú definované nasledovne:

[1] bezpečnostný koeficient rovný 0 sme nemuseli zbytočne uvažovať, keďže vieme zvoliť K aby $2K-Q+1$ bolo kladné

$$Q(v_i) = [< v_{(i+1)modN}, v_{(i+Q)modN} >]^K$$

Vlastné kvórové rezy sú teda ľubovoľné K-prvkové podmnožiny nasledujúcich Q uzlov za daným uzlom (v zoradení podľa indexov), pričom indexy uzlov rátame cyklicky, teda modulo počet uzlov v sieti.

Tieto siete sú veľmi podobné už spomínaným *reťazovým* sieťam, avšak v nich sme využili, že všetky uzly „záviseli“ len na malej množine uzloch a tam našli tie podstatné ktoré boli v každom kvóre. V tomto type sietí je väčšia decentralizácia, keďže aj uzly s veľkým indexom závisia zas na uzloch s malým indexom, takže tu budeme musieť analýzu siete urobiť trikovejšie.

Veta Uvažujme Q po sebe idúcich uzlov v cyklickej sieti $A =< v_i, v_{(i+Q)modN})$. Potom v každom jej kvóre je aspoň K uzlov z A.

Dôkaz Zoberme si ľubovoľné kvórum Q_1 a pre spor predpokladajme, že menej ako K uzlov z tohto kvóra je v množine A. Potom zrejme $v_{(i-1)modN} \notin Q_1$, keďže všetky jeho vlastné kvórové rezy sú K-prvkovou podmnožinou A, čo by bol spor. Keďže každý kvórový rez má aspoň $K + 1$ prvkov, tak existuje uzol v_j taký že

$$v_j \in Q_1 \wedge v_j \notin A \wedge \forall u \in < v_{(j+1)modN}, v_i) \ : u \notin Q_1$$

Teda „posledný“ uzol z kvóra, ktorý je „pred“ množinou A. V Q_1 musí ale byť aj niektorý vlastný kvórový rez v_j, teda K vrcholov z $< v_{(j+1)modN}, v_{(j+Q)modN} >$.
Rozlíšime 2 prípady:

1. $v_i \in < v_{(j+1)modN}, v_{(j+Q)modN} >$

   V takomto prípade platí

   $< v_{(j+1)modN}, v_{(j+Q)modN} > = < v_{(j+1)modN}, v_i) \cup < v_i, v_{(j+Q)modN} >$.

   Pričom z definície v_j žiadny uzol z $< v_{(j+1)modN}, v_i)$ nie je v Q_1. Teda všetkých K uzlov musí byť z $< v_i, v_{(j+Q)modN} > \subseteq < v_i, v_{(i+Q)modN})$, čo je presne čo sme chceli dokázať.

2. $v_i \notin < v_{(j+1)modN}, v_{(j+Q)modN} >$

   V takomto prípade Q_1 neobsahuje žiadny kvórový rez v_j, keďže
   $< v_{(j+1)modN}, v_{(j+Q)modN} > \subseteq < v_{(j+1)modN}, v_i)$. Čo je spor s tým, že Q_1 obsahuje kvórový rez v_j a teda je kvórum.

■

Teraz už ľahko vieme odhandúť najmenší množný počet uzlov v kvóre. Totiž v každej Q-tici (tých je N) je aspoň K uzlov a zároveň každý uzol z kvóra sme zarátali presne Q-krát, teda uzlov v kvóre je aspoň $\lceil \frac{NK}{Q} \rceil$. Keď už vieme minimálnu veľkosť kvóra vieme odhadnúť aj minimálny prienik dvoch kvór. Keďže všetkých uzlov je N, tak každý prienik bude mať aspoň $2\lceil \frac{NK}{Q} \rceil - N$ uzlov.

Aby sme sa nemuseli zaoberať prípadmi kde sa budeme trápiť s hornými celými časťami a technicky ťažšími konštrukciami, ďalej v analýze pokračujme len s prípadom keď N je násobkom Q.
V tomto prípade ukážeme, že tento náš odhad je aj tesný a je to náš bezpečnostný koeficient. Uvažujme dve kvóra:

$$\{v_i \mid i \ mod \ Q < k\} \ a \ \{v_i \mid i \ mod \ Q \geq Q - k\}$$

Je priamočiare overiť, že naozaj obe množiny sú kvóra a ich prienik je rovný $\max(\frac{N}{Q}(2K - Q), 0)$, čo teda naozaj sedí s našim odhadom po úprave a teda naozaj toto je najmenší možný prienik kvór. Zas sa teda ukazuje, že voliť $K \leq \frac{Q}{2}$ nevedie k bezpečným sieťam. Všimnime si, že bezpečnostný koeficient vychádza podobne ako pri *reťazových* sieťach, avšak je prenásobený $\frac{N}{Q}$, keďže uzly sú medzi sebou viac zviazané a neexistujú žiadne „centrálne“ uzly na ktorých všetko závisí.

Podobne ako pri *reťazových* sieťach sa zamyslime ako voliť K aby bol koeficient odolnosti, čo najvyšší. Bezpečnostný koeficient ako aj koeficient životaschopnosti je lineárna funkcia v K s opačným vedúcim znamienkom (jedna klesá, druhá stúpa) a preto keď chceme maximalizovať minimum z týchto dvoch funkcií, stačí ich položiť do rovnosti ($K \leq \frac{Q}{2}$ už neuvažujeme).

Teda maximálny koeficient odolnosti získame ak platí:

$$\frac{N}{Q}(2K - Q) = Q - K + 1 \qquad \text{teda} \qquad K = \frac{Q+N+1}{2N+Q}Q$$

Pri zvolení takto optimálneho K (je jasné, že nie vždy nám vyjde celočíselne a budeme musieť číslo zaokrúhliť pre dosiahnutie maxima) dostávame *koeficient odolnosti*:

$$Q - K + 1 = Q - \frac{Q+N+1}{2N+Q}Q + 1 = Q(\frac{N-1}{2N+Q}) + 1 = \frac{Q - \frac{Q}{N}}{2 + \frac{Q}{N}} + 1$$

Keďže $\frac{Q}{N} \leq 1$, tak sa nám ukazuje, že naša sieť sa vie dostať blízko ku odolnosti $\frac{Q}{2}$ (alebo aj kúsok ďalej), teda polovici počtu uzlov, ktorým jeden dôveruje. Vysvetlenie je jednoduché. Pri $K \leq \frac{Q}{2}$ sme mali nulový prienik, teda siete neboli použiteľné. Naopak v našom bepečnostnom koeficente nám vystupovalo N, ktorým sme vedeli ľubovoľne zväčšiť bezpečnostný koeficient, zatiaľ čo koeficient životaschopnosti závisel len od rozdielu Q a K. Preto pre dostatočne veľké siete sme vedeli pri vhodnom výbere K zvýšiť koeficient odolnosti výrazne.

Ako príklad si môžeme zobrať sieť s parametrami $N = 6, Q = 3, K = 2$, kde nám koeficient odolnosti vyjde 2.

Kapitola 5

Náhodné siete

Keď jednotlivým uzlom v sieti necháme voľnosť pri výbere kvórových rezov výsledná sieť môže vyzerať rôzne. Keď necháme sieť organicky sa vyvíjať v reálnom svete ukazuje sa, že sieť skončí viac či menej centralizovaná alebo naopak náhodná, v ktorej ťažko hľadať pravidlá a zákonitosti podľa ktorých sa bude sieť vyvíjať ďalej, poprípade ktoré uzly majú väčšiu váhu pri rozhodovaní siete o novom bloku.

Reťazové siete z minulej kapitoly boli príkladom siete, ktorá skončila centralizovaná, presnejšie závislá len na niekoľkých uzloch. Pokiaľ sieť naozaj vznikala organicky bez vonkajších zásahov (každý uzol si zvolil kvórové rezy podľa seba), tak tieto uzly na ktorých závisí sieť majú na jednej strane zdravo vybudovanú autoritu a teda predpoklady byť naozaj dôveryhodné, no pokiaľ ich je málo znižujú bezpečnostný koeficient.

V tejto kapitole sa preto budeme venovať druhému zaujímavejšiemu prípadu. A to sieťam, kde kvórové rezy budú generované náhodne. Pokúsime sa najskôr nagenerovať niekoľko takýchto sietí a následne zrátať ich koeficienty odolnosti. Aby sme vedeli ale robiť niektoré porovnania medzi jednotlivými sieťami a zamedziť veľa prípadom kde sieť má bezpečnostný koeficient 0 (tieto prípady sú nezaujímavé, keďže v praxi sú tieto siete nepoužiteľné) budeme uvažovať len *zjednodušené NQK-siete*.

5.1 Generovanie sietí

Na generovanie sietí, konkrétne kvórovej funkcie, sme vytvorili program, ktorý ako parameter berie typ siete ktorý má vytvoriť. Podporuje typy: „chain“ (reťazová sieť), „cyclic“ (cyklická sieť) a „random“ (náhodná sieť). Aby bolo jednoduché doplniť prípadné generovanie aj iných typov sietí, na samotné naprogramovanie sme použili návrhový vzor „Stratégia“[5].

Následne po spustení programu je možné na vstupe zadať N, Q, K – parametre zjednodušenej NQK-siete. Pri generovaní náhodnej siete sme na generovanie náhodných kvórových rezov použili Fisher-Yatesov algoritmus[4], ktorým sme pre každý uzol

vytvorili náhodnú permutáciu ostatných uzlov a všetky K-prvkové podmnožiny prvých Q uzlov sme označili za jeho vlastné kvórové rezy.
Samotný program na generovanie sietí je priložený v prílohe.

5.2 Hľadanie bezpečnostného koeficientu

Na nájdenie tohto koeficientu potrebujeme nájsť veľkosť najmenšieho prieniku kvór. Ako sme ukázali tento problém je NP-ťažký, preto podobne ako pri generovaní sietí sme naprogramovali pomocou *Stratégie* viac možných postupov ako tento prienik nájsť. Algoritmus na hľadanie sa skladá z dvoch častí, z hľadania kvór a následne najmenšieho prieniku medzi ľubovoľnými dvoma z nich.

Na hľadanie kvór sme naprogramovali tri stratégie:

- *all* – pozrie sa na každú podmnožinu a overí, či je to kvórum
- *minimal* – rovnako ako stratégia *all*, ale ak už je niektorá podmnožina kvórum, tak jej nadmnožiny už neoveruje, teda nájde len minimálne kvóra
- *random* – predchádzajúce stretégie generujú všetky podmnožiny a preto sú nepoužiteľné na väčších sieťach, preto táto stratégia skúsi nájsť aspoň niekoľko náhodných kvór

Použitie stratégie si môžeme zvoliť parametrom pri spúšťaní programu. Overiť či je podmnožina kvórum je jednoduché, stačí overiť, či pre každý vrchol existuje kvórový rez taký, že aj on je celý v danej podmnožine.

Stratégia *random* postupuje tak, že sa spustí niekoľko krát. Na začiatku každého spustenia si náhodne určí štartovací vrchol a udržuje si aktuálnu množinu uzlov Q, v ktorej máme na začiatku iba štartovací vrchol. Následne sa snaží množinu Q „rozširovať“ pokým netvorí kvórum.

Rozširovať sa ju snaží nasledovne. Udržuje si množinu A ešte nespracovaných uzlov (takých čo ešte nemajú celý kvórový rez v Q) v ktorej je na začiatku len štartovací vrchol. Pokým je A neprázdna vyberieme z nej ľubovoľný prvok a a snažíme sa do Q pridať niektoré uzly, tak aby niektorý kvórový rez a bol obsiahnutý v Q. Keďže naša sieť je *NQK-sieť*, tak len vyberieme našich Q_a dôveryhodných uzlov a stačí nám do Q pridať K_a uzlov odtiaľto. Algoritmus vždy vyberie všetky uzly, ktoré už v Q sú (nebudeme zbytočne zväčšovať kvórum). A pokiaľ ich ešte nie je dosť, náhodne vyberie ďalšie a tie pridá do A aj do Q. Teraz už uzol a má kvórový rez v Q a môžeme ho vymazať z A. Takto keď vyprázdnime A, tak z Q už budeme mať určite kvórum a náš program môžeme zopakovať a hľadať iné kvórum. Zjavne každý uzol bude v A najviac raz, takže algoritmus sa nezacyklí.

Pre detajly uvádzame aj pseudokód 1 na nájdenie jedného kvóra, keď ich chceme nájsť viac spustíme tento algoritmus viac krát.

Algoritmus 1 Hľadanie jedného kvóra stratégiou random

$v \leftarrow$ náhodný začiatočný uzol
$Q \leftarrow \{v\}$
$A \leftarrow \{v\}$
while A je neprázdne **do**
    $a \leftarrow$ ľubovoľný uzol z A
    $kandidati \leftarrow$ všetky dôveryhodné uzly uzla a nenachádzajúce sa v Q
    if $K_a > Q_a - |kandidati|$ **then**
        $noveUzly \leftarrow$ náhodných $K_a - Q_a + |kandidati|$ uzlov z množiny $kandidati$
        $Q \leftarrow Q \cup noveUzly$
        $A \leftarrow A \cup noveUzly$
    $A \leftarrow A \setminus \{a\}$
return Q

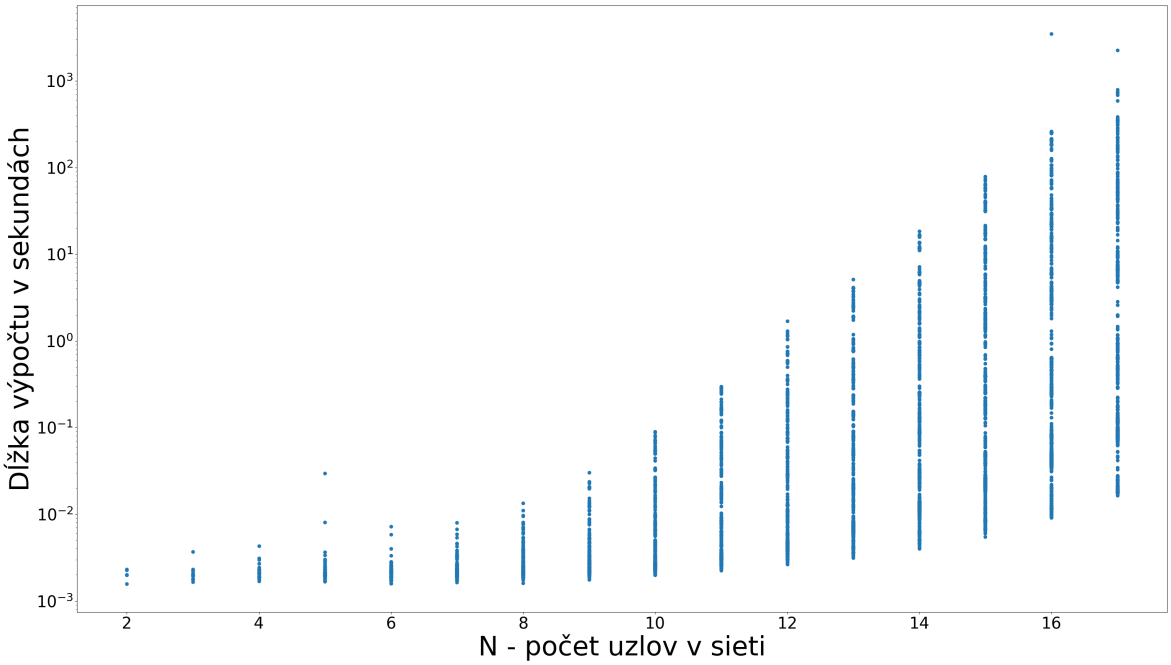
Keď už máme nájdený nejaký zoznam kvór (či už všetky alebo aspoň niekoľko náhodných), tak ešte musíme zistiť ich najmenší prienik. To budeme robiť jednoducho a teda tak, že porovnáme každé kvórum s každým, zrátame počet spoločných uzlov a zistíme najmenší výsledok.
Samotný kód v C++ sa nachádza v prílohe.

5.3 Exaktné metriky malých sietí

Najprv sa skúsime pozrieť na siete, ktorým vieme zrátať bezpečnostný koeficient exaktne. Aj keď stratégia *minimal* je rýchlejšia ako *all*, keďže nekontroluje úplne všetky podmnožiny, tak nie je použiteľná na siete s vysokým počtom uzlov. Môže sa totiž stať že bude musieť kontrolovať takmer každú podmnožinu.

Vygenerovali sme si teda *zjednodušené NQK-siete* s parametrami $2 \leq N \leq 17$, $1 \leq Q < N, 1 \leq K \leq Q$, pričom z každej trojice parametrov spĺňajúcich tieto obmedzenia sme vygenerovali 5 náhodných sietí (teda pre každý uzol náhodné kvórové rezy).

Ako si môžeme všimnúť na grafe 5.1 čas výpočtu nášho algoritmu v najhoršom prípade exponenciálne rastie s počtom uzlov v našej sieti. Pre 17 uzlov v sieti algoritmus niekedy potrebuje 10^3 sekúnd, čo je zhruba 15 minút. Preto sme už pre väčšie siete nedokázali analyzovať exaktné výsledky. Všimnime si, že naopak niektoré siete s 17 uzlami sme boli schopný analyzovať celkom rýchlo pod sekundu. Je to spôsobené našou stratégiou, ktorá závisí od veľkosti minimálnych kvór. Totiž ak má sieť

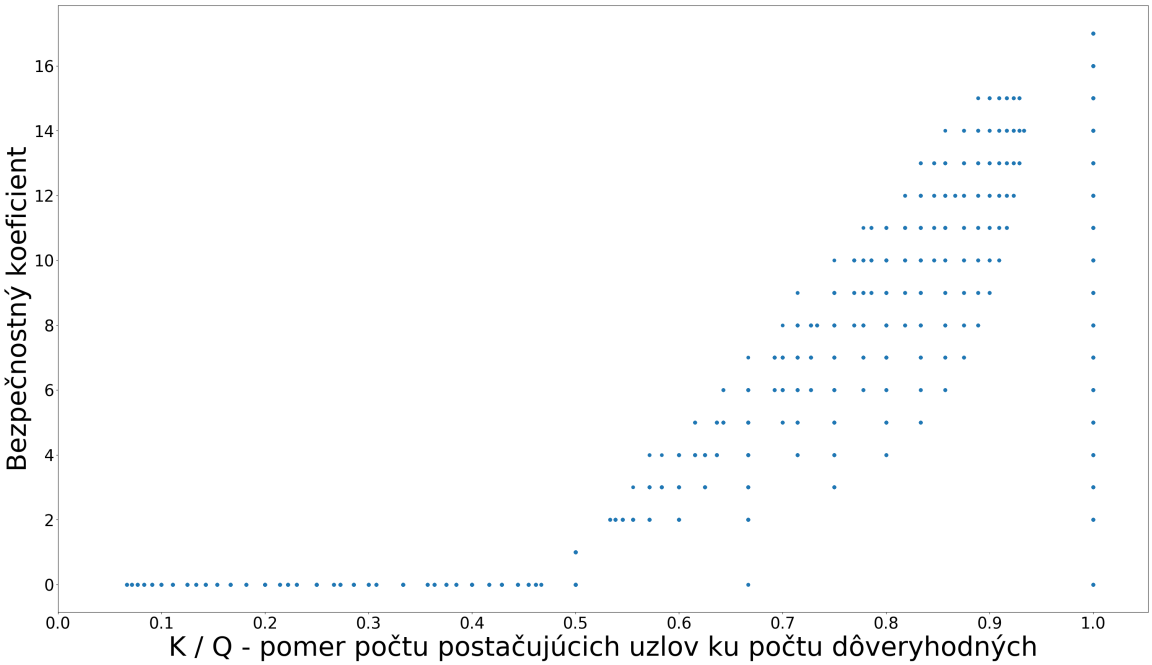


Obr. 5.1: Graf rastu dĺžky výpočtu algoritmu od veľkosti siete

malé kvóra nemusíme kontrolovať väčšie nadmnožiny a algoritmus je rýchlejší. Avšak vidno, že aj rýchlosť algoritmu v najlepšom prípade tiež rapídne rastie (y-ová os rastie logaritmicky).

Analýzu týchto malých sietí, ktoré sme boli schopný exaktne spočítať začneme grafom 5.2, ktorý ukazuje aký bezpečnostný koeficient majú jednotlivé siete vzhľadom na pomer parametrov K a Q. Môžme si všimnúť, že graf má akokeby 3 časti. Prvá časť sú siete v ktorých je K príliš malé vzhľadom na Q. Presnejšie ak $K < Q/2$, tak medzi nami vygenerovanými sieťami žiadna nemala prienik kvór. Z čoho dostávame, že nie sú bezpečné a teda sú v praxi nepoužiteľné a pre nás nezaujímavé. Naše dáta sú síce veľmi malé siete, no môžeme vysloviť hypotézu, že siete s $K < Q/2$ nebudú nikdy bezpečné a teda môžeme neodporučiť uzlom voliť takéto kvórové rezy. Na našu hypotézu sa ešte raz pozrieme v nasledujúcej podkapitole kde sa budeme pozerať aj na väčšie siete hoci nebudeme vedieť určiť bezpečnostný koeficient úplne presne, budeme vedieť spraviť aspoň horný odhad, ktorý pokiaľ bude nulový bude aj presný.

Posledná časť grafu sú siete kde $Q = K$. Pokiaľ sa pozrieme na kvóra takýchto vygenerovaných sietí, zistíme, že majú jediné kvórum a to všetky uzly, čo im teda dá bezpečnostný koeficient rovný N, avšak koeficient životaschopnosti bude 1, čo výrazne obmedzuje odolnosť siete. Určite jednoducho vieme zostrojiť aj siete tohto typu s viacerými kvórami. Napriek tomu sme vygenerovali množstvo sietí (uvažujúc každý uzol



Obr. 5.2: Graf vzťahu pomeru K ku Q a bezpečnostného koeficientu

za jedinečný, každú sieť s rovnakou pravdepodobnosťou pri zadaných parametroch) a všetky boli tohto typu, z čoho môžme usudzovať, že drvivá väčšina takto postavených sietí skončí s jediným minimálnym kvórom.

Ostali nám len siete s $Q > K \geq Q/2$, ktoré na grafe vyzerajú už zaujímavejšie. Môžme si všimnúť, že s rastúcim pomerom stúpa aj maximálny aj minimálny bezpečnostný koeficient nájdený pre daný pomer (nemusí nutne stále rásť, ale „zhruba“ rastie). Pre takto vysoké pomery dokonca platí zaujímavá vlasnosť.

Veta V *NQK-sieťach* s množinou uzlov V, ktoré majú prienik kvór je bezpečnostný koeficient aspoň $\min_{v \in V} 2K_v - Q_v + 1$.

Dôkaz Pre spor predpokladajme, že existuje menší prienik dvoch kvór. Keďže takáto sieť má prienik kvór, tak existuje uzol u, ktorý sa nachádza v tomto prieniku. Keďže u patrí obom kvóram, tak obom kvóram patria aj po jednom z jeho vlastných kvórových rezov. A to sú K–prvkové podmnožiny Q–prvkovej množiny uzlov vybranej uzlom u. Všimnime si ale, že každé dve takéto podmnožiny majú prienik aspoň $2K_u - Q_u$ uzlov, ktoré teda patria tiež obom kvóram. Teda náš uvažovaný prienik má aspoň $2K_u - Q_u + 1 \geq \min_{v \in V} 2K_v - Q_v + 1$ uzlov, čo je spor.■

Pri štúdiu našich *zjednodušených NQK-sietí*, teda máme že tento bezpečnostný

Obr. 5.3: Graf odchýlky koeficientu odolnosti ku optimu

koeficient je aspoň $2K - Q + 1$ (ak je nenulový). Týmto môžeme získať intuíciu rastu minima bezpečnostného koeficientu medzi vygenerovanými sieťami s daným pomerom K/Q. Keďže jednak pre zväčšujúci sa pomer, sa bude funkcia $2K - Q + 1$ tiež zväčšovať a tiež fixný menovateľ zlomku má len obmedzenú možnosť priblížiť sa 1 (najbližšie sa dostane pri $\frac{Q-1}{Q}$), teda aj samotné Q bude čoraz väčšie. Mohlo by sa teda zdať, že najlepšie pre uzly je voliť K blízke Q. Avšak v takýchto prípadoch klesá *koeficient životaschopnosti*.

Posledný odstavec boli len úvahy na vysvetlenie správania sa grafu 5.2. To čo je ale podstatné pre život siete je koeficient odolnosti. Ten vieme zrátať jednoducho keď už máme bezpečnostný koeficient. Koeficient životaschopnosti pre *zjednodušené NQK-siete* sa ráta ľahko ako sme si už ukázali. Keďže siete s $K < Q/2$ neboli bezpečné a použiteľné, tak ich v tejto úvahe používať nebudeme. Chceme zistiť ako voliť K aby sieť dosiahla čo najväčší koeficient odolnosti. Pomocou grafu 5.3 si ukážeme, že ako najrozumnejšia voľba sa ukazuje $K = \frac{2}{3}Q$.

Pre každú vygenerovanú sieť sme si vygenerovali jej „optimálny“ náprotivok, teda sieť s rovnakými parametrami s tým, že $K = \frac{2}{3}Q$ ($K = \lfloor \frac{2}{3}Q \rfloor + 1$ ak by K nevychádzalo celočíselné). Vypočítali sme koeficienty odolnosti oboch sietí a do grafu zaznačili rozdiel koeficientu pôvodnej siete od „optimálnej“.

Ako si môžeme na grafe všimnúť naozaj všetky tieto rozdiely vyšli nezáporné a teda

sa nám podarilo potvrdiť na našich dátach optimálnosť voľby K. Ďalšia zaujímavá vec je, že, keď vypočítame koeficient odolnosti na sieťach s $K = \frac{2}{3}Q$, dostávame výsledky okolo $\frac{Q}{3}$, čo je veľmi podobné s výsledkami v *reťazových* sieťach. Dokonca pokiaľ si zadefinujeme obdobné pojmy ako naše koeficienty v centralizovanom Byzantínskom protokole[9], získame presne rovnaký pomer, ktorý musíme zvoliť aby sme získali rovnovážny stav medzi obdobnými pojmami koeficientu životaschopnosti a bezpečnostného koeficientu v tomto protokole. A ako výsledok dostaneme približne rovnakú odolnosť siete (počet uzlov, ktorých môžu zlyhať a neovplyvní to chod siete).

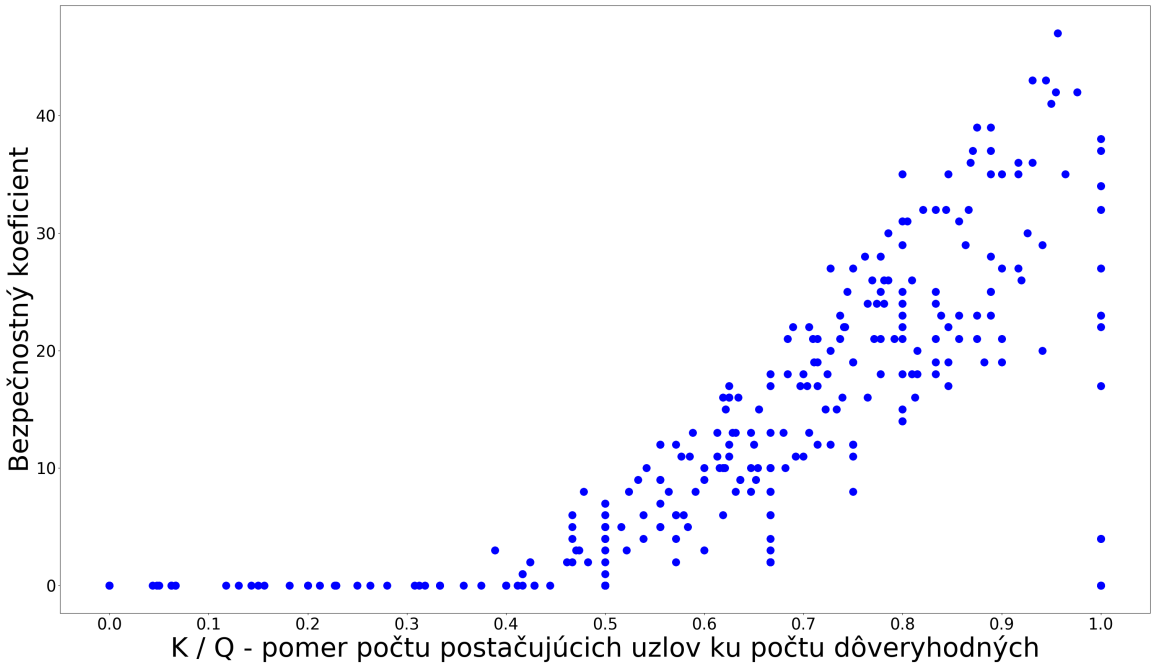
5.4 Horné odhady na koeficient odolnosti väčších sietí

Teraz sa skúsime pozrieť aj na siete s väčším počtom uzlov, avšak už na hľadanie bezpečnostného koeficientu budeme schopný požiť len stratégiu *random*. Tá však nemusí nájsť všetky kvóra a teda nebudeme schopný určiť tento koeficient presne, ale budeme ho vedieť zhora ohraničiť, keďže ak už nejaký prienik kvór nájdeme, tak buď sme ešte nejaký menší nenašli alebo už menší neexistuje.

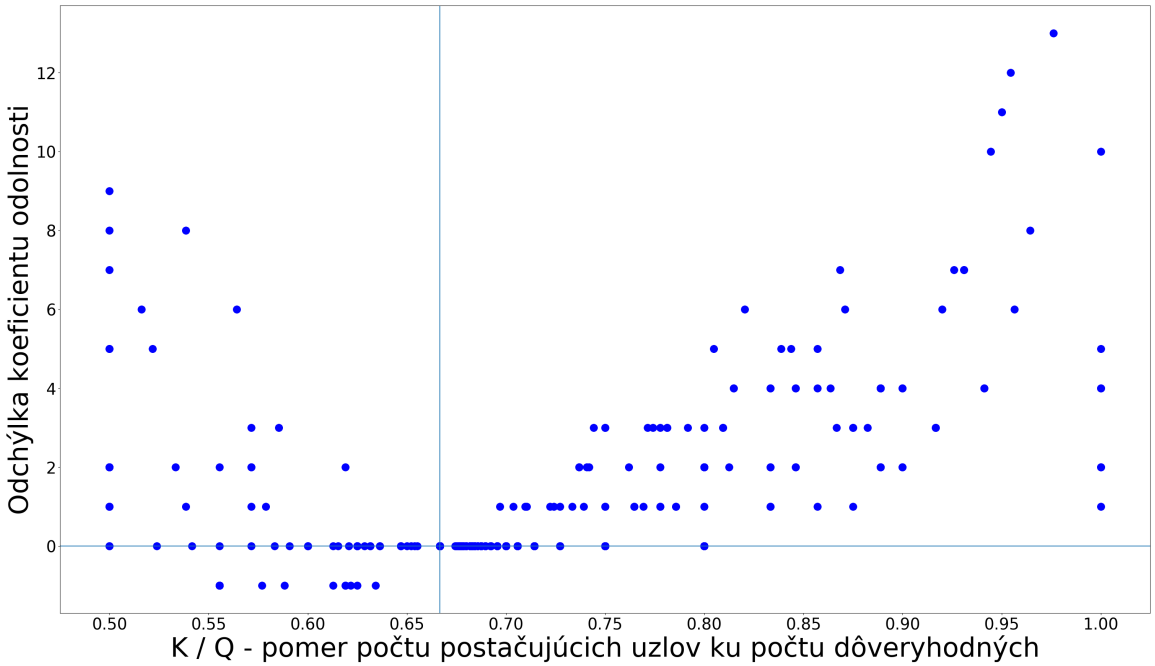
Vygenerovali sme si teda *zjednodušené NQK-siete* s parametrami $21 \leq N \leq 50$, $1 \leq Q < N, 1 \leq K \leq Q$, pričom parametre sme generovali tiež náhodne a dokopy sme vygenerovali 200 sietí. Ako hornú hranicu sme zvolili 50 aby sme jednak mali náš odhad celkom presný (generovali sme 2000 náhodných kvór pre jednu sieť) a zároveň dĺžka výpočtu bola únosná (väčšina výpočtov dokázala bežať do 15 sekúnd).

Najprv si skúsime vykresliť obdobný graf 5.4 ako pri malých sieťach a to vzťah medzi pomerom K/Q a bezpečnostným koeficnetom. Podobne ako v grafe 5.2 tiež vidíme, že siete s $K < \frac{Q}{2}$ majú poväčšine bezpečnostný koeficient nulový. Ešte by sme si mali uvedomiť, že v grafe máme len horné odhady na bezpečnostné koeficienty, teda reálne môžu byť aj nižšie. V každom prípade to potvrdzuje našu hypotézu o tom, že voliť kvórové rezy tak, že $K < \frac{Q}{2}$ vedie k veľmi nízkemu až nulovému koeficientu bezpečnosti a teda je dobré používať väčšie K. Zvyšok grafu vyzerá veľmi podobne ako 5.2. Môžme preto ľahko našu hypotézu rozšíriť aj na väčšie siete a predpokladať, že bezpečnostný koeficient bude rásť podobne. Čo nám dáva lepší odhad pri budovaní siete o jej bezpečnosti.

Na záver sa však pozrime na to najdôležitejšie a teda koeficient odolnosti. Tu už takisto nebudeme uvažovať siete $K < \frac{Q}{2}$. Podobne ako pri malých sieťach si vykreslíme rovnako stavaný graf 5.5, na ktorého vybudovanie sme však použili vygenerované väčšie siete a iba horný odhad bezpečnostného koeficientu a teda aj koeficientu odolnosti. Podobne ako pri malých sieťach sa nám znova potrvdzuje hypotéza o ideálnom volení parametra K v sieti, tak aby sme maximalizovali koeficient odolnosti. Ukazuje sa nám,



Obr. 5.4: Graf vzťahu pomeru K ku Q a bezpečnostného koeficientu veľkých sietí



Obr. 5.5: Graf odchýlky koeficientu odolnosti ku optimu vo väčších sieťach

že pri mierne nižšom zvolení pomeru vieme získať o kúsok väčší koeficient odolnosti (v našom prípade o 1), avšak bezpečnosť siete je oveľa dôležitejšia ako samotná životaschopnosť a tak podobne ako v [9], môžeme tvrdiť že predsalen je lepšie zvoliť pomer medzi K a Q ako $\frac{2}{3}$. Pretože aj keď celková odolnosť môže vyjsť o kúsok horšie, keďže bezpečnostný koeficient vyjde lepšie (graf 5.4), tak nepriateľ prinajhoršom iba zablokuje sieť ale dáta v nej ostanú správne. A z tohto stavu je jednoduchšie sieť opraviť a obnoviť jej prevádzku.

Čo však na tomto grafe vidno omnoho lepšie je, že čím viac sa vzďaľujeme od tohto magického pomeru, tým viac sa naša odchýlka zväčšuje. A teda najmä pri sieťach kde ťažko hovoriť o dôveryhodnosti konkrétnych uzlov a akejkoľve dôležitosti uzla na protokol odporúčame uzlom voliť kvórové rezy tak, aby počet postačujúcich uzlov (K) bol aspoň približne rovný $\frac{2}{3}$ počtu dôveryhodných uzlov (Q). Keďže ako sme ukázali to štatisticky vedie ku sieti, ktorá je schopná prežiť najviac zlyhaných uzlov. Tak ako sme už uviedli niekoľko krát vyššie, *koeficient odolnosti* bude vychádzať okolo $\frac{Q}{3}$.

Záver

V tejto práci sme si v úvode predstavili problematiku dosahovania konsenzu, ukázali sme si ako využiť riešenie tohto problému vo finančnej sieti, kde často musíme riešiť podobné problémy a rátať s nedôveryhodnými entitami v sieti. Tiež sme si zhrnuli nedostatky existujúcej finančnej siete a takisto doteraz známych protokolov, ktoré využívali na vykonávanie transakcií konsenzus alebo aj iné metódy, ktoré sú medzi dnešnými kryptomenami populárnejšie.

Následne sme si predstavili samotný Stellar konsenzus protokol a porovnali sme ho s už predstavenými inými metódami a ukázali si, ktoré problémy iných metód rieši. Pri vylepšení samotného problému konsenzu išlo o zbavenie sa centrálnej autority pri vstupe novej entity do siete. Zoznámili sme sa so samotným fungovaním Stellar konsenzus protokolu a zistili sme podmienky pri ktorých sieť dokáže odolávať útočníkom.

Sformulovali sme si presnejšie metriky na hodnotenie siete podľa toho koľko zlyhaní uzlov ju nedokáže ovplyvniť. Dokázali sme si, že pre všeobecné siete je zisťovanie týchto metrík NP-ťažký problém a teda sme si spresnili model siete aby sme vedeli o sieti a jej odolnosti povedať aspoň niečo.

Vybrali sme si konkrétne modely *reťazových* a *cyklických* sietí, zrátali ich metriky a dostali lepší vhľad do tvorenia kvór v takýchto sieťach. Pri *reťazových sieťach* sme zistili, že celá sieť je závislá na menšej množine uzlov. Táto centralizácia však nebola až taká nepríjemná, keďže sieť dokázala odolať aj zlyhaniu $\frac{1}{3}$ z týchto uzlov. Pri *cyklických sieťach* už sme na žiadnu centralizáciu nenarazili, dokonca sme si ukázali, že sieť by bola v istých parametroch schopná prežiť napadnutie až $\frac{1}{2}$ dôveryhodných uzlov, ktoré si konkrétny uzol vybral. Táto sieť je však prudko symetrická a tesne zviazaná, čo sa nám v reálnej sieti pravdepodobne nestane.

Na záver sme sa pozreli na to ako by dopadla odolnosť siete ak by boli siete vytvorené náhodne. Najprv pri malých sieťach, kde sme metriky vedeli vypočítať presne a potom aj na väčších kde sme však dokázali tieto metriky vypočítať len približne.

Štatistiky z vybraných typov sietí a neskôr aj z náhodných sietí ukazujú, že dokážeme získať odolnosť voči zlyhaniu až $\frac{1}{3}$ entít, čím sa dokážeme porovnávať aj s centralizovanou verziou protokolov na konsenzus. Zistili sme, že takúto vysokú odolnosť dosahujeme keď konkrétna entita je ochotná nechať sa presvedčiť okolo $\frac{2}{3}$ entitami, ktorým dôveruje (voľba K v našich *NQK-sieťach*). Naopak voľba $K < \frac{Q}{2}$ sa ukázala ako

nevhodná, keďže väčšina takto vygenerovaných sietí bola nepoužiteľná s bezpečnostným koeficientom 0 a teda sieť nedávala žiadne garancie aj keď sa všetky uzly správali podľa protokolu.

Napriek tomu sme dokázali analyzovať len siete s počtom entít do 50, čo nám dáva veľký priestor na zlepšenie. Takisto naša metóda pri väčších sieťach nemusela byť veľmi presná a už len lepšie zrátať jej presnosť a tak zlepšiť odhady našich metrík môže dať oveľa lepší pohľad do bezpečnosti a životaschopnosti sietí založených na Stellar konsenzus protokole.

Literatúra

[1] Miguel Castro, Barbara Liskov, et al. Practical byzantine fault tolerance. In *OSDI*, volume 99, pages 173–186, 1999.

[2] Stephen A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the third annual ACM symposium on Theory of computing*, pages 151–158. ACM, 1971.

[3] Cynthia Dwork and Moni Naor. Pricing via processing or combatting junk mail. In *Annual International Cryptology Conference*, pages 139–147, 1992.

[4] Ronald A Fisher and Frank Yates. *Statistical tables for biological, agricultural and medical research*. Oliver and Boyd, 1943.

[5] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software Addison-Wesley*, pages 315–325. 1995.

[6] Richard M. Karp. Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85–103. Springer, 1972.

[7] Sunny King and Scott Nada;. Ppcoin: Peer-to-peer crypto-currency with proof-of-stake. `https://pdfs.semanticscholar.org/0db3/8d32069f3341d34c35085dc009a85ba13c13.pdf`, 2012. [Citované 2019-01-24].

[8] Łukasz Lachowski. Complexity of the quorum intersection property of the federated byzantine agreement system. `https://arxiv.org/pdf/1902.06493.pdf`, 2019. [Citované 2019-04-10].

[9] Jinyuan Li and David Maziéres. Beyond one-third faulty replicas in byzantine fault tolerant systems. In *In Proceedings of the 4th Symposium on Networked Systems Design and Implementation*, pages 131–144, 2007.

[10] M. S. Paterson M. J. Fischer, N. A. Lynch. Impossibility of distributed consensus with one faulty process. Technical report, MIT, CAMBRIDGE LABORATORY FOR COMPUTER SCIENCE, 1982.

[11] David Mazieres. The stellar consensus protocol: A federated model for internet-level consensus. `https://www.stellar.org/papers/stellar-consensus-protocol.pdf`, 2015. [Citované 2019-01-24].

[12] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. `https://bitcoin.org/bitcoin.pdf`, 2008. [Citované 2018-01-24].

[13] Karl J. O'Dwyer and David Malone. Bitcoin mining and its energy footprint. In *Irish Signals and Systems Conference*, pages 280–285, 2014.

Príloha

CD

K dispozícií na priloženom CD sa nachádzajú 2 programy v C++. V priečinku „network_creator“ sa nachádza generátor *zjednodušených NQK-sietí* popísaný v práci.

V priečinku „safety_coefficient_finder“ zas program zisťujúci bezpečnostný koeficient siete zo vstupu. Sieť musí byť zadaná vo formáte výstupu programu „network_creator“.