Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Žiga Kokelj

Primerjava metod sledenja označenih kovancev v omrežju Bitcoin

magistrsko delo

magistrski študijski program druge stopnje
računalništvo in informatika

Mentor: doc. dr. Lovro Šubelj
Somentor: dr. Matej Trampuš

Ljubljana, 2021

Zahvala

Zahvaljujem se mentorju doc. dr. Lovru Šublju in somentorju dr. Mateju Trampušu za vso pomoč in koristne nasvete ob izdelavi magistrske naloge. Velika zahvala gre tudi mojim najbližjim, ki so mi skozi leta študija stali ob strani in me podpirali.

Žiga Kokelj, 2021

"Vires in Numeris."

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
POW	proof of work	dokaz o delu
POS	proof of stake	dokaz o imetju
POT	proof of time	dokaz o času
P2P	peer to peer	vsak z vsakim
FIFO	first in first out	prvi noter, prvi ven
LIFO	last in first out	zadnji noter, prvi ven
SHA	secure hash algorithm	kriptografska zgoščevalna funkcija SHA
ECDSA	elliptic curve digital signature algorithm	algoritem digitalnega podpisa, ki uporablja eliptične krivulje
UTXO	unspent transaction output	neporabljen izhod transakcije
P2PK	pay to public key	plačaj-na-javni-naslov
P2PKH	pay to public key hash	plačaj-na-zgostitev-javnega-ključa
P2MS	pay to multisig	večpodpisno plačilo
P2SH	pay to script hash	plačaj-na-zgostitev-naslova-skripte
ASIC	application specific integrated circuit	integrirano vezje za specifično aplikacijo
AMLD	anti money laundering directive	direktiva o preprečevanju pranja denarja
KYC	know your customer	poznavanje strank
RAM	random access memory	bralno pisalni pomnilnik
SSD	solid state drive	fiksni disk, ki temelji na tehnologiji hitrega pomnilnika
API	application programming interface	aplikacijski programski vmesnik

Povzetek

Naslov: Primerjava metod sledenja označenih kovancev v omrežju Bitcoin

Bitcoin s svojo odprtostjo in psevdonimnostjo nudi mnoge priložnosti in izzive. Eden od izzivov je sledenje označenim kovancem skozi omrežje Bitcoin transakcij z namenom opozarjanja na izhode transakcij, ki izvirajo iz kriminalnih dejanj. Zaradi velikega števila vozlišč in kompleksnosti grafa transakcij smo razvili metode za preiskovanje tega omrežja. V magistrski nalogi smo implementirali znane metode in jim dodali novo metodo, imenovano COMB. Pripravili in optimizirali smo podatkovno bazo, ki omogoča tako preiskovanje ter pridobili vzorca sumljivih in naključnih transakcij. Na njih smo pognali metode in analizirali dobljene rezultate. Ugotovili smo, da imajo vse metode določene prednosti in slabosti. Analizirali smo preseke grafov, nastalih z različnimi metodami, saj imajo te transakcije višjo verjetnost za povezavo z izvorno transakcijo. Pripravili smo tudi podatkovno bazo, ki vključuje dodatne podatke, ki jih metode pri svojem odločanju lahko uporabijo. Analiza je pokazala velik potencial tega pristopa, saj smo že na razmeroma majhni bazi v več primerih prišli do znanih transakcij.

Ključne besede

Bitcoin, veriženje blokov, analiza omrežij

Abstract

Title: Comparison of tainting analysis methods in Bitcoin network

Bitcoin offers many new opportunities and challenges with its pseudonymity and open source nature. One of the challenges is performing taint analysis in order to follow coins that originated from criminal activities. Due to a large number of nodes and the complexity of the Bitcoin transaction graph, methods for the performance of taint analysis have been developed. In this master's thesis, existing methods were implemented and furthermore a new method called COMB was proposed. A database that supports running these methods was put together. For the testing purpose, two data sets of starting transaction outputs were prepared. After executing all methods on the data sets and analysis of the results, it was concluded that all methods have pros and cons. The intersections of graphs produced by different algorithms from the same starting inputs were analyzed, because they contain transactions with a higher probability of being connected to the starting transaction output. Another database with off-chain data that can be used in implemented methods was developed. Even with a relatively small database, we were able to reach some known transactions with implemented methods, showing the big potential of this technique.

Keywords

Bitcoin, blockchain, network analysis

Poglavje 1

Uvod

Leta 2008 je Satoshi Nakamoto, katerega identiteta ostaja neznana, v svojem članku opisal delovanje omrežja Bitcoin in na ta način rešil problem, ki je v računalništvu znan kot problem bizantinskih generalov. Bitcoin predstavlja prvo sredstvo za prenašanje vrednosti med oddaljenimi posamezniki, ne da bi se pri tem zanašal na zaupanja vrednega posrednika. Kljub temu, da je v preteklem desetletju nastalo mnogo novih kriptovalut, pa je Bitcoin še vedno najbolj znana in (glede na tržno kapitalizacijo) najbolj vredna kriptovaluta. Transakcije, s katerimi uporabniki prenašajo svoje kovance med sabo se shranijo v podatkovno strukturo imenovano veriga blokov. Vsak uporabnik omrežja lahko vidi celotno zgodovino transakcij, ki so vključene v bloke. Kljub dostopu do teh podatkov, pa je povezovanje transakcij in naslovov z identiteto njihovih lastnikov težka naloga, saj vsi udeleženci sodelujejo s svojim psevdonimom. Skrivna identiteta je privlačila kriminalce, ki uporabljajo bitcoin za plačevanje ilegalnih poslov, pranje denarja in zahtevajo nakazila za odkupnine v bitcoinu. Uporabniki običajno vstopajo in izstopajo iz trga kriptovalut preko reguliranih kriptomenjalnic, ki pred nakupom in prodajo od uporabnika zahtevajo določene identifikacijske podatke. Zato je pomembno, da znamo transakcijam slediti skozi graf transakcij. S tem se

ukvarjajo določena specializirana podjetja[1][2] in pristojne agencije posameznih držav[3]. S sledenjem sumljivim kovancem skozi omrežje lahko posameznike in podjetja obvestimo o sumljivem izvoru in tako kriminalcem preprečimo ali pa vsaj otežimo unovčitev nezakonito pridobljenih kripto kovancev. Sledenje kovancem ni trivialno, saj je v množici transakcij običajno premalo podatkov, da bi se lahko algoritem nedvoumno odločal, katerim izhodom glede na označene vhode slediti. Zato uporabljamo različne metode, ki s svojimi hevristikami določajo izhode, ki so glede na označene vhode najbolj sumljivi in jim tako sledimo naprej. V nadaljevanju smo opisali znane metode, predlagali novo metodo in vse skupaj implementirali ter testirali na podatkih iz Bitcoin verige blokov. Metode smo med seboj primerjali po različnih metrikah in jih kritično ovrednotili. Preprečevanje in odkrivanje kriminalcev, ki pri poslovanju uporabljajo Bitcoin, se nam zdi pomembno ob vse večji uporabnosti kriptovalut. Upamo, da smo s tem delom prispevali delček k napredku na tem področju.

Sorodna dela

Raziskovanje in analiziranje grafa transakcij Bitcoin omrežja seveda ni nova tema. Večina člankov se osredotoča na razvrščanje posameznih Bitcoin naslovov v gruče [32, 31, 43] (angl. clustering) ter s tem povezanim nivojem anonimnosti [44, 45]. Za analizo blockchaina je bilo razvitih nekaj ogrodij [46, 47], ki omogočajo različne analize. Ogrodje BiVA omogoča vizualizacijo in raziskovanje podgrafa omrežja Bitcoin in analizo s pomočjo nekaterih algoritmov. Analizo celotnega blockchaina nam omogoča ogrodje [48], ki se v veliki meri osredotoča na hitrost, saj je ob naraščujoči velikosti blockchaina to ena ključnih lastnosti takih ogrodij. Za povečanje anonimnosti so bile raz-

[1] https://www.chainalysis.com/chainalysis-kyt/

[2] https://www.elliptic.co/blog/elliptic-follows-bitcoin-ransoms-paid-by-darkside-ransomware-victims

[3] https://www.justice.gov/opa/pr/department-justice-seizes-23-million-cryptocurrency-paid-ransomware-extortionists-darkside

vite različne storitve, ki izhode transakcij različnih uporabnikov med seboj premešajo (angl. mixing services) in tako otežijo njihovo sledenje [52]. Tudi ti pristopi pa niso brez pomanjkljivosti in vsaj delno še vedno omogočajo sledenje transakcijam [49, 50]. Mnogo člankov problematizira kriminal[28], povezan s kriptovalutami; podaja rešitve vezane na prepoznavanje takih aktivnosti [51] in njihovo sledenje. To potrjuje, da gre za zelo aktualen raziskovalni problem.

Poglavje 2

Bitcoin

2.1 Kriptovalute in veriženje blokov

Kriptovalute so digitalne valute, katerih delovanje temelji na kriptografskih konceptih. Praviloma so decentralizirane in za shranjevanje transakcij uporabljajo tehnologijo veriženja blokov. Načeloma jih ne izdaja in z njimi ne upravlja nobena posamezna entiteta, ampak vsi udeleženci sledijo pravilom določenega protokola, ki določa delovanje kriptovalute. Zaradi hitrega razvoja mnogih novih kriptovalut v zadnjem desetletju je vse težje določiti lastnosti, ki bi veljale za vse kriptovalute.

V nalogi se bomo osredotočili na kriptovaluto Bitcoin [1], saj je to prva in po tržni kapitalizaciji [2] še vedno največja kriptovaluta.

Ena glavnih lastnosti kriptovalut je možnost prenosa virtualnih kovancev med uporabniki, ne da bi za to potrebovali skrbnika, osrednji strežnik ali pa poznali in zaupali ostalim udeležencem v omrežju. Transakcije so zavarovane s pomočjo kriptografije javnih in zasebnih ključev. Iz veljavnih transakcij se ustvari enota, imenovana blok, ki se zapiše na konec verige blokov, katero pred spremembami ščitijo različni sistemi. Najpogosteje uporabljeni se imenuje dokaz dela (angl. proof of work, POW) in ga bomo podrobneje opisali v podpoglavju 2.3.6. Kot njegove alternative pa se pojavljajo še dokaz o imetju (angl. proof of stake, POS), dokaz o času (angl. proof of time, POT)

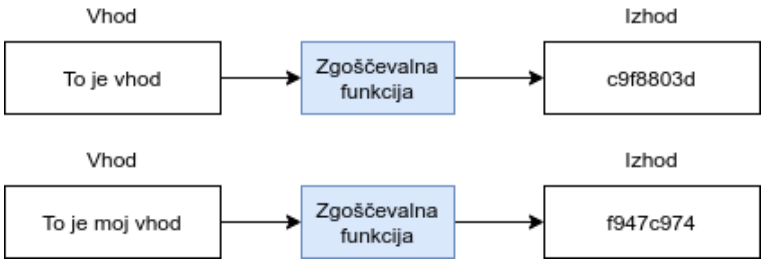
in drugi novejši in manj preverjeni sistemi.

2.2 Kriptografski koncepti uporabljeni v Bitcoinu

Kot omenjeno v prejšnjem poglavju 2.1, se kriptovalute pri svojem delovanju močno zanašajo na kriptografijo. V naslednjih podpoglavjih bomo opisali osnovne kriptografske koncepte uporabljene v protokolu Bitcoin.

2.2.1 Zgoščevalne funkcije

Zgoščevalna funkcija (angl. hash function) je funkcija, ki za vhod lahko sprejme poljubno dolgo sporočilo, kot izhod pa vrne vrednost fiksne dolžine imenovano zgoščena vrednost (angl. hash value). Kriptografske zgoščevalne funkcije so podmnožica v množici vseh zgoščevalnih funkcij. Zaradi svojih lastnosti so zelo uporabne v kriptografiji. Vhod (sporočilo) in izhod (zgoščena vrednost ali tudi izvleček, prstni odtis) sta običajno zapisana v binarni obliki, vendar se zaradi preglednosti pogosto pretvarjata v šestnajstiško obliko. Privlačna lastnost zgoščevalnih funkcij za uporabo v kriptografiji je enostaven izračun zgoščene vrednosti iz sporočila in hkrati izjemno visoka zahtevnost izračuna v obratni smeri.



Slika 2.1: Primer zgoščevalne funkcije

Pri kriptografskih zgoščevalnih funkcijah si želimo naslednjih lastnosti [4]:

- ob enaki vhodni vrednosti je rezultat (zgoščena vrednost) vedno enak,
- zgoščena vrednost ne razkrije nobenih informacij o vhodnem sporočilu (primer na sliki 2.1),
- majhna sprememba vhodnega sporočila povzroči velike spremembe v zgoščeni vrednosti (angl. avalanche effect),
- izračun zgoščene vrednosti je računsko in pomnilniško nezahteven.

Med zgoščevalne funkcije z dobrim nivojem varnosti uvrščamo zgoščevalne funkcije, ki imajo odpornosti na sledeče napade [3]:

- odpornost na napad s prasliko (angl. preimage resistance) pomeni, da je računsko praktično neizvedljivo v doglednem času na podlagi zgoščene vrednosti najti vhodno sporočilo. Za dano vrednost y mora biti izjemno težko najti vrednost x, pri kateri velja $f(x) = y$,
- odpornost na napad z drugo prasliko (angl. second preimage resistance) pomeni, da je računsko praktično neizvedljivo v doglednem času najti vhod, za katerega funkcija vrne enako zgoščeno vrednost, kot za podani vhod. (Za dano vrednost x je izjemno težko najti x', da velja $f(x) = f(x')$, pri čemer x in x' nista enaka.),
- odpornost na trke (angl. collision resistance) pomeni, da je računsko praktično neizvedljivo v doglednem času najti dve različni vrednosti, katerih zgoščena vrednost je enaka. (Izjemno težko je najti par x, x' za katera velja $f(x) = f(x')$.)

V protokolu Bitcoin se uporablja algoritem iz družine zgoščevalnih funkcij SHA-2 [5], ki vrne 256-bitno zgoščeno vrednost in se pogosto imenuje kar SHA-256. Spada med bločne šifre in deluje na blokih dolžine 512 bitov.

Na določenih delih pa je uporabljena tudi zgoščevalna funkcija RIPEMD-160 [6], ki vrne zgoščeno vrednost v dolžini 160 bitov in je tako prostorsko učinkovitejša za shranjevanje. Zaradi povečane varnosti pa se običajno

uporablja v kombinaciji z zgoraj omenjeno funkcijo SHA-256 na način, da zgoščeno vrednost pridobljeno s funkcijo SHA-256 še dodatno zgostimo s funkcijo RIPEM-160.

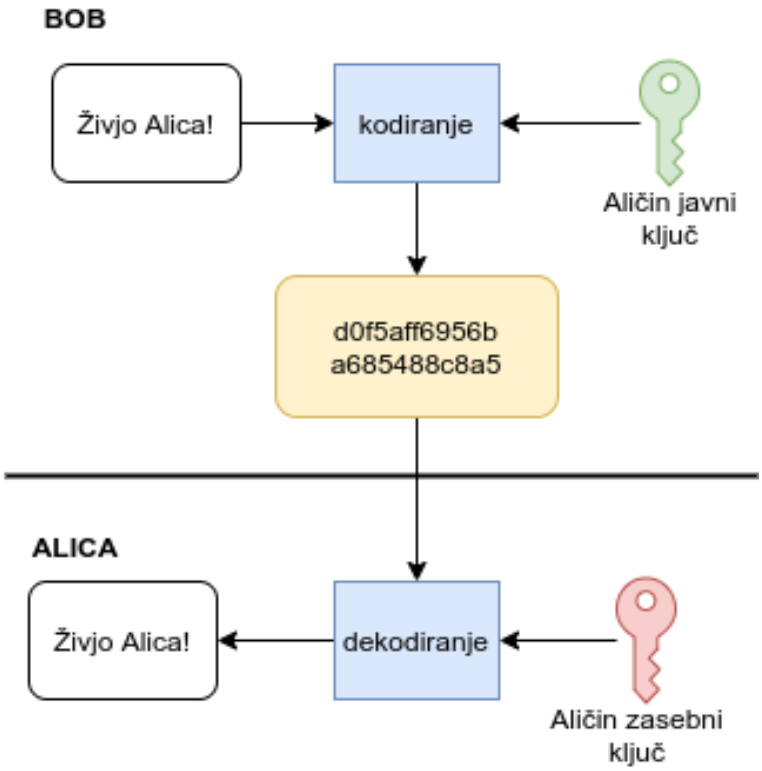
2.2.2 Kriptografija javnih in zasebnih ključev

Kriptografija javnih in zasebnih ključev je pomemben del Bitcoina in ostalih kriptovalut. Spada v področje asimetrične kriptografije, saj za šifriranje in dešifriranje potrebujemo dva različna ključa. Poznamo javni in zasebni ključ. Vrednost prvega običajno delimo z ostalimi udeleženci v komunikaciji, vrednost drugega pa moramo za varno delovanje poznati le mi. Na sliki 2.2 lahko vidimo prikaz uporabe javnega in zasebnega ključa.

Generiranje javnega in zasebnega ključa poteka s funkcijo, ki iz naključno izbrane številke generira zasebni ključ in iz njega izračuna javni ključ. Ker gre pri izračunu javnega ključa iz zasebnega za enosmerno funkcijo, je računsko praktično nemogoče izračunati zasebni ključ, kljub poznavanju javnega ključa.

V kriptovalutah nam javni ključ (ali pa naslov, ki je skrajšana verzija javnega ključa) predstavlja lokacijo, kamor lahko nekomu nakažemo kovance. Zasebni ključ pa nam omogoča uporabo kovancev, ki si jih lastimo, v nadaljnjih transakcijah.

V Bitcoinu se za operacije asimetrične kriptografije uporablja algoritem ECDSA (angl. Eliptic Curve Digital Signature Algorithm), ki uporablja eliptične krivulje. V prihodnosti pa bo v Bitcoinu poleg algoritma ECDSA na voljo tudi algoritem Schnorr [7], ki uporablja enake pare javnih in zasebnih ključev, a ponuja izboljšave pri podpisovanju in preverjanju podpisov.



Slika 2.2: Prikaz delovanje digitalnega podpisa

2.2.3 Digitalni podpisi

Digitalni podpisi [8] uporabljajo kriptografijo javno-zasebnih ključev in zgoščevalne funkcije, kar smo opisali v prejšnjih dveh podpoglavjih. Digitalni podpisi se uporabljajo za podpis sporočila. Za podpisovanje in preverjanje podpisa potrebujemo dve ločeni metodi, kar je razvidno tudi iz slike 2.3.

Postopek podpisovanja

1. Izračunamo zgoščeno vrednost dokumenta/sporočila, ki ga želimo podpisati.

2. Dobljeno zgoščeno vrednost šifriramo s svojim zasebnim ključem ter tako dobimo digitalni podpis.

3. Digitalni podpis dodamo k dokumentu in ga pošljemo prejemniku (prejemniku pošljemo tudi svoj javni ključ, v kolikor ga ta še nima).

Preverjanje podpisa

1. Sprejmemo dokument z digitalnim podpisom.

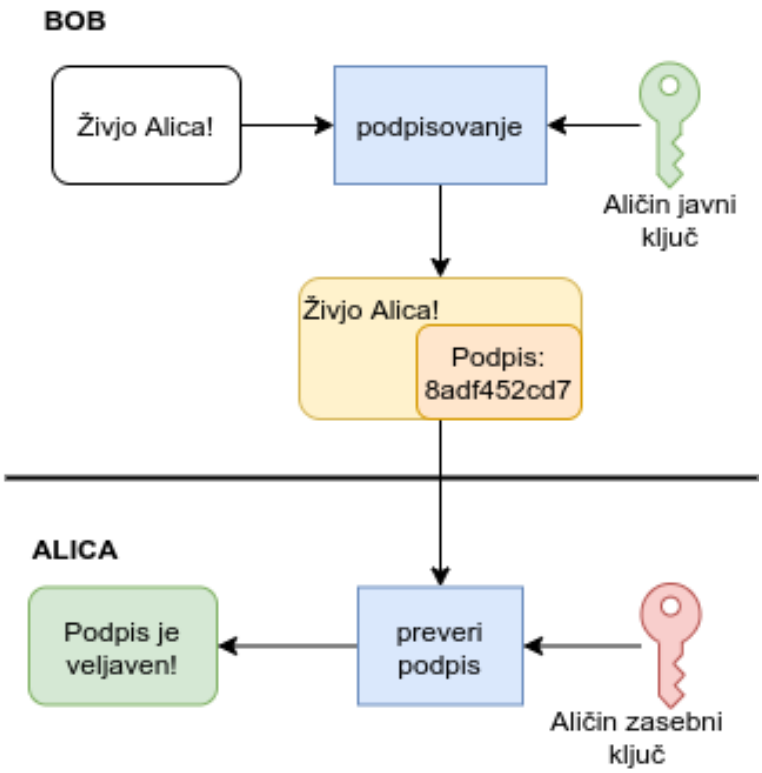
2. Izračunamo zgoščeno vrednost sporočila (brez podpisa).

3. Vzamemo digitalni podpis in ga dešifriramo s pomočjo javnega ključa pošiljatelja.

4. Primerjamo dešifriran digitalni podpis in izračunano zgoščeno vrednost. V primeru, da sta enaka, je podpis veljaven.

Z digitalnim podpisom tako dokažemo, da smo mi (z našim zasebnim ključem) podpisali točno to sporočilo (ujemajoče zgoščene vrednosti). Pri tem je pomembno, da pošiljatelj in prejemnik uporabljata enak algoritem za podpisovanje in preverjanje podpisa.

V Bitcoinu se za ta namen uporablja algoritem ECDSA, ki kot svojo eliptično krivuljo uporablja secp256k1 in dvakrat zgoščeno vrednost z algoritmom SHA-256 [9].



Slika 2.3: Prikaz delovanje digitalnega podpisa

2.3 Podrobnejši opis kriptovalute bitcoin

2.3.1 Uvod

Prva kriptovaluta bitcoin je svet ugledala leta 2009, ko je njen anonimni avtor Satoshi Nakamoto začel poganjati omrežje Bitcoin in to delil na spletnem forumu za kriptografske navdušence. Že leto prej pa je objavil članek z naslovom "Bitcoin: A Peer-to-Peer Electronic Cash System"[1], kjer opisuje njegovo delovanje. Gre za prvo decentralizirano digitalno kriptovaluto, ki je ustvarjena in shranjena v elektronski obliki in deluje brez osrednjega nadzora. Delovanje celotnega omrežja, ki vključuje izmenjavo transakcij, validacijo transakcij, ustvarjanje blokov, validacijo blokov, pošiljanje blokov in

doseganje soglasja glede stanja blokov temelji na odprtokodni programski opremi. Prvo verzijo klienta Bitcoin Core je v jeziku C++ razvil Satoshi Nakamoto, kasneje pa se je umaknil iz razvoja. Pozneje so nastale še druge implementacije Bitcoin klientov, ki sledijo istemu protokolu.

Bitcoin omrežje je decentralizirano omrežje uporabnikov (angl. peer to peer network, P2P), v katerem so vsi udeleženci obravnavani enakopravno. Udeleženci se med seboj ne poznajo in zato posameznemu udeležencu ne moremo zaupati. Kljub temu pa protokol poskrbi, da sistem deluje, dokler je računska moč poštenih udeležencev večja od 50 odstotkov.

Uporabniki kriptovalut običajno za upravljanje z javnimi in zasebnimi ključi ter za ustvarjanje transakcij uporabljajo kombinacijo programske in strojne opreme imenovano kripto denarnica.

Poznamo različne vrste denarnic:

- vroča denarnica (angl. hot wallet)

  Vroča denarnica je običajno programska oprema, ki jo uporabljamo na računalniku ali mobilnem telefonu. Njena prednost je priročnost uporabe, saj imamo sredstva ves čas na voljo. Glavna slabost pa se skriva v nevarnosti, da nam vdrejo v napravo in posledično ukradejo sredstva, saj so te naprave pogosto povezane s svetovnim spletom,

- hladna denarnica (angl. cold wallet)

  Za hladno denarnico običajno uporabljamo specifično strojno opremo, ki ni povezana s svetovnim spletom. Občutljive informacije (predvsem zasebni ključi) pa niso izpostavljeni niti pri povezavi z računalnikom.

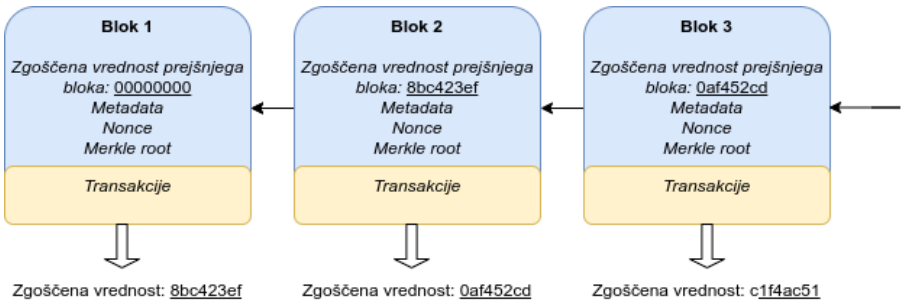
Nove enote valute bitcoin nastajajo v procesu, imenovanem rudarjenje, kar smo podrobneje opisali v poglavju 2.3.6. Večina lastnikov bitcoina pa svojih kovancev ni pridobila z rudarjenjem, temveč so jih kupili preko storitve, ki se imenuje kripto menjalnica. Storitev običajno poteka preko spletne strani, kamor se registriramo. V večini primerov moramo zaradi zakonodaje priložiti tudi sliko osebnega dokumenta. Nato pa preko bančnega nakazila ali plačila s kreditno kartico kupimo izbrano kriptovaluto (npr. bitcoin) po

takratni tržni ceni. Kupljeno kriptovaluto lahko hranimo v denarnici ponudnika kripto menjalnice ali pa si sredstva prenesemo v lastno denarnico.

V naslednjih podpoglavjih bomo predstavili ključne sestavne dele Bitcoina in opisali njihovo delovanje.

2.3.2 Veriga blokov

Veriga blokov je podatkovna struktura, pri kateri je možno le dodajanje novih elementov, imenovanih bloki. Vsak element (blok) vsebuje polje, ki nam pove, kateri blok je njegov predhodnik. V Bitcoinu se za vrednost polja, ki se sklicuje na prejšnji blok, uporablja zgoščena vrednost bloka, izračunana z algoritmom SHA-256. V verigi blokov je možno blok vedno dodati le za obstoječi blok. Dodajanje bloka med dva obstoječa bloka v verigi blokov ali spreminjanje katerekoli informacije v obstoječih blokih bi razveljavilo veljavnost vseh naslednjih blokov v verigi, saj vsaka sprememba povzroči spremembo zgoščene vrednosti bloka. To nam daje zagotovilo, da v tej podatkovni strukturi ne moremo spreminjati obstoječih zapisov, ampak le dodajamo nove na konec verige.



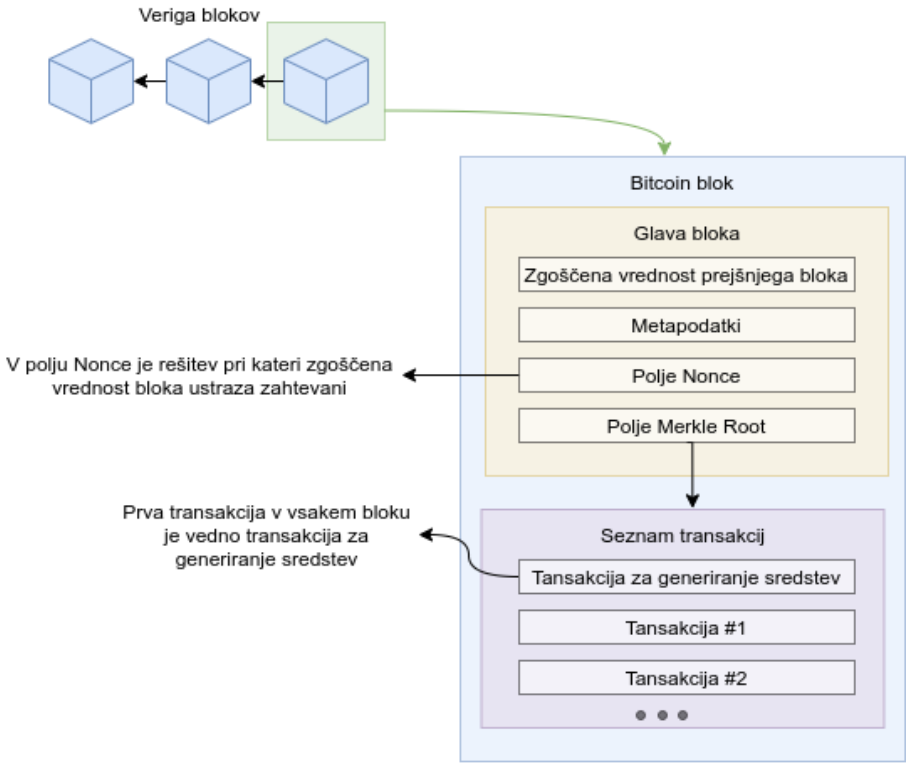
Slika 2.4: Veriga blokov

V praksi ne gre vedno le za verigo blokov, ampak nastane drevo, pri katerem vsaka veja drevesa predstavlja alternativno verzijo zgodovine. Tehnološko-ekonomski principi, opisani v poglavju 2.3.6 in 2.3.7 pa poskrbijo, da se udeleženci čim prej poenotijo glede zgodovine.

Veriga blokov predstavlja celotno zgodovino transakcij na Bitcoinu, in je distribuirano shranjena na vseh polnih vozliščih (angl. full node).

2.3.3 Blok

Kot omenjeno že v prejšnjem podpoglavju 2.3.2, je blok osnovni gradnik verige blokov. Sestavljen je iz glave bloka (angl. block header) in seznama transakcij. Podrobnejša zgradba je prikazana na sliki 2.5.



Slika 2.5: Struktura Bitcoin bloka.

Glava bloka vsebuje naslednja polja:

- zgoščeno vrednost prejšnjega bloka,
- metapodatke (verzija, časovni žig, zahtevana težavnost),

- polje Nonce,
- polje Merkle Root, ki predstavlja povzetek vseh transakcij.

V seznamu transakcij, ki sledi glavi bloka, so zapisane vse transakcije, vključene v blok.

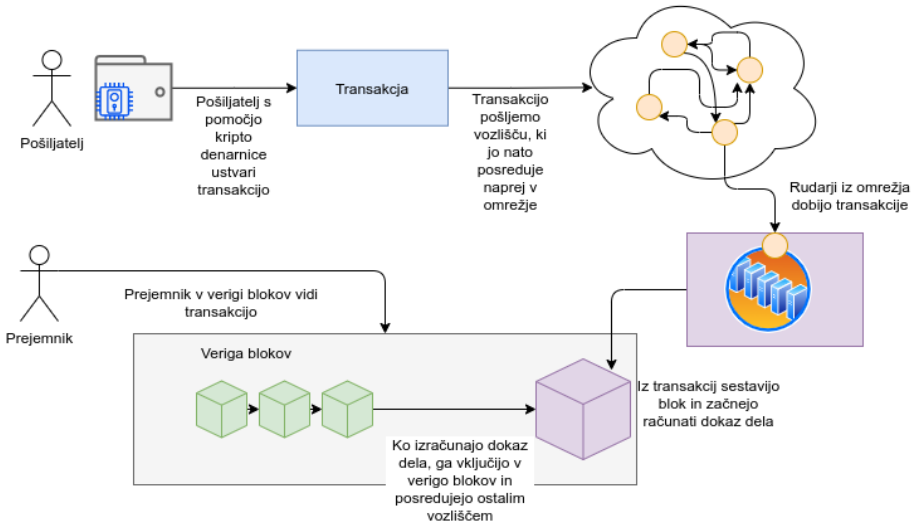
V verigo blokov se nov blok doda v povprečju na vsakih 10 minut. Velikost bloka pa je v bitcoinu (BTC) omejena na 1 MB. Ta omejitev je bila v preteklosti tudi glavni razlog za močna nesoglasja v skupnosti, ki so privedla do razcepa protokola in nastanka kriptovalute Bitcoin Cash ter njenega kasnejšega razcepa s katerim je nastala kriptovaluta Bitcoin SV.

Omejitev velikosti bloka je poleg povprečnega časa ustvarjanja novega bloka razlog, da je omrežje Bitcoin sposobno v povprečju izvesti le okoli 7 transakcij na sekundo, kar je v primerjavi s sodobnimi plačilnimi sistemi, kot sta Visa ali PayPal, izredno nizko. Prednost majhne velikosti blokov je v večji stopnji decentralizacije, saj ima tako več uporabnikov na voljo strojno opremo, s katero lahko sodeluje pri validaciji transakcij.

2.3.4 Transakcija

Transakcija je osnovna enota prenosa sredstev med uporabniki Bitcoina. V osnovi so sestavljene iz vhodov in izhodov (podrobnejši opis je na voljo spodaj). Bitcoin za beleženje stanja uporablja model UTXO (angl. unspent transaction output), ki uporablja neporabljene izhode transakcij za beleženje stanja in kreiranje novih transakcij. Model se precej razlikuje od klasičnega modela beleženja stanja, kjer imamo seznam uporabnikov in pri vsakem uporabniku določeno stanje na računu. Posledica tega je, da lahko v bitcoinu izhod transakcije porabimo le enkrat v celoti. Za lažje razumevanje si lahko predstavljamo analogijo iz fizičnega sveta, kjer lahko evrski bankovec unovčimo le v celoti, morebitno razliko med ceno in vrednostjo kovanca pa dobimo vrnjeno. Na podoben način deluje Bitcoin, saj si v primeru, da vhodi presežejo vrednost, ki jo želimo nakazati v transakciji, lahko enostavno

ustvarimo še dodaten izhod in na ta način preostanek nakažemo nazaj na svoj naslov.



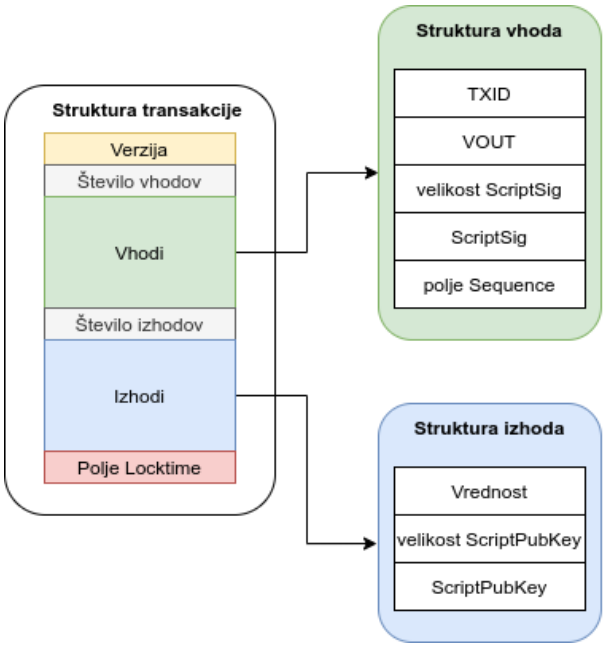
Slika 2.6: Potek bitcoin transakcije

Transakcije imajo točno določeno strukturo, ki vsebuje:

- verzijo strukture transakcije,
- število vhodov transakcije,
- vhode transakcije,
- število izhodov transakcije,
- izhode transakcije,
- polje Locktime, ki nam pove minimalno višino/čas bloka, v katerega naj bi bila transakcija vključena.

Po mehkem razcepu protokola leta 2017, imenovanem SegWit obstaja tudi drugačna struktura transakcije. Digitalni podpis je od takrat mogoče shraniti tudi izven bloka. Na ta način se zmanjša velikost transakcije in zaradi

konstantne velikosti bloka lahko v posamezen blok shranimo več transakcij. Digitalni podpisi se v tem primeru shranijo v ločeno podatkovno strukturo.



Slika 2.7: Originalna struktura Bitcoin transakcije

Bitcoin transakcije se identificirajo z identifikacijskim nizom transakcije imenovanim TXID (angl. transaction id), ki ga izračunamo iz podatkov transakcije s pomočjo zgoščevalne funkcije.

Ob navajanju vhodov transakcije moramo navesti identifikacijski niz transakcije v kateri je nastal izhod, ki bo sedaj porabljen in njegovo zaporedno številko. Poleg tega pa navedemo še velikost skripte in skripto za odklepanje (angl. ScriptSig), ki odklene kovance, zaklenjene v navedenem izhodu.

Pri izhodih pa moramo navesti vrednost, ki jo želimo v določenem izhodu. Bitcoin je deljiv na osem decimalnih mest. Njegova najmanjša enota je imenovana Satoshi. Po vrednosti podamo tudi velikost skripte ter skripto, ki ta izhod zaklene (angl. ScriptPubKey). Poleg zaklepa izhoda ima Bitcoin še nekaj drugih možnosti uporabe, ki pa so bolj podrobno opisane v podpoglavju 2.3.5.

Pri ustvarjanju transakcij moramo paziti, da je vsota vrednosti vhodov višja ali enaka vrednosti izhodov transakcije. Razlika med vhodi in izhodi pa je nagrada za rudarja, ki transakcijo vključi v blok in prvi prikaže dokaz dela. Na ta način je rudarjem v interesu v bloke vključevati transakcije s čim višjo nagrado (angl. miner fee).

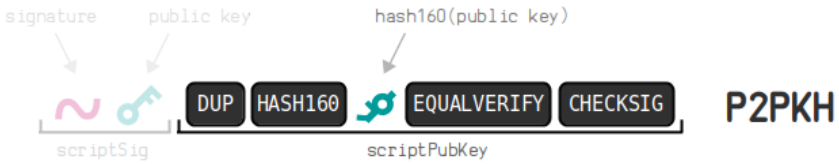
Edina izjema pri pravilu o vsoti vhodov in izhodov je transakcija za generiranje sredstev (angl. coinbase transaction). Ta se vedno nahaja na prvem mestu bloku in z njo rudar dobi nagrado za opravljeno delo in vse nagrade iz transakcij, vključenih v blok. Transakcija tako nima nobenega vhoda, vendar ima izhode, kar je edini način s katerim se ustvarjajo novi Bitcoin kovanci.

Če je katerokoli pravilo za veljavnost prekršeno (npr. vhod v transakcijo je bil že porabljen, skripta za odklepanje ne odklene sredstev, vsota izhodov je večja od vsote vhodov, itd.), se transakcija zavrne. Na ta način se ne širi med uporabniki v omrežju in ni vključena v blok. Če bi jo zlonameren rudar vseeno vključil v blok in ta blok dodal v verigo blokov, pa bi ta blok in vse bloke, ki gradijo na tem bloku, zavrnili vsi ostali pošteni udeleženci.

2.3.5 Bitcoin Script

Kot smo omenili v prejšnjem poglavju, v izhode bitcoin transakcij običajno damo skripto, ki jih zaklene. Ob njihovi uporabi pa uporabimo skripto, ki odklene zaklenjene izhode. V bitcoinu se za ta namen uporablja skriptni jezik imenovan Script. Gre za jezik, ki je zelo podoben jeziku Forth [10]. Jezik namerno nima zank, izvajanje skript pa je časovno in prostorsko omejeno. Jezik podpira kriptografske operacije, ki so potrebne za preverjanje podpisov in se izvaja s pomočjo sklada (angl. stack-based).

Ob preverjanju transakcij združimo in poženemo skripto iz izhoda, ki ga zaklene ter skripto iz vhoda, ki ga odklene. Če se združena skripta izvede brez napak in je na skladu le element z logično vrednostjo resnično (angl. true), to pomeni, da je bila skripta za odklepanje pravilna.



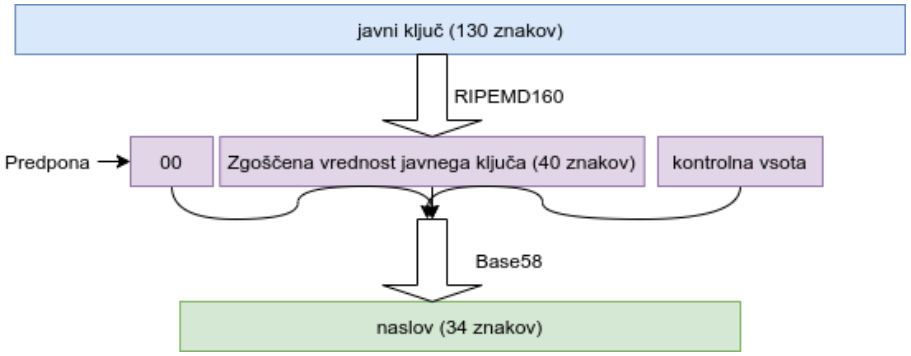
Slika 2.8: "Plačaj-na-zgostitev-javnega-ključa".[39]

Kljub možnostim ustvarjanja poljubnih skript, pa se v veliki večini uporabljajo tako imenovane standardne skripte, ki jih delimo na naslednje tipe:

- "Plačaj-na-javni-naslov" (angl. Pay To Public Key, P2PK) – s to skripto zaklenemo izhod na določen javni ključ.
- "Plačaj-na-zgostitev-javnega-ključa" (angl. Pay To Public Key Hash, P2PKH) – s to skripto zaklenemo izhod na zgoščeno vrednost javnega ključa. Struktura te skripte je razvidna iz slike 2.8. Gre za najpogosteje uporabljen tip skripte.
- Večpodpisna (angl. Pay To Multisig, P2MS) – ta skripta nam omogoča, da izhod lahko porabimo, če skripta za odklep vsebuje vsaj N od M potrebnih digitalnih podpisov. Vrednosti N im M definiramo v skripti.
- "Plačaj-na-zgostitev-naslova-skripte" (angl. Pay To Script Hash, P2SH) – ta skripta nam omogoča, da zaklenemo izhod z zgoščeno vrednostjo skripte. Ob odklepanju moramo poleg podpisov podati tudi celotno skripto, katere zgoščena vrednost se ujema s tisto, podani v skripti P2SH.
- skripta NULL DATA – ta skripta nam omogoča shranjevanje dodatnih podatkov v verigo blokov. Izhodi s to skripto nikoli ne morejo biti porabljeni kot vhodi v nove transakcije.

Med uporabo bitcoina se pogosto srečamo z Bitcoin naslovom. Gre za niz dolg 34 znakov. Prepoznamo jih po tem, da se začenjajo z znakoma 1 ali 3.

Njihovo generiranje je možno na podlagi javnega ključa, kar je tudi prikazano na sliki 2.9. V primerjavi z javnim ključem je naslov krajši, kodiranje v Base58 [12] pomaga pri preprečevanju napak zaradi zamenjav podobnih črk, poleg tega pa je naslovu dodana tudi kontrolna vsota.



Slika 2.9: Postopek izračuna naslova iz javnega ključa.

2.3.6 Rudarjenje in dokaz dela

Rudarji (angl. miners) skrbijo za ustvarjanje novih blokov, ki jih dodajajo na konec verige blokov. Za dodajanje novega bloka morajo najti dokaz o delu (angl. proof of work). Rudarji v Bitcoin omrežju med seboj tekmujejo za to, kdo bo prvi rešil računsko težko nalogo in dodal nov blok, s tem pa tudi pridobil nagrado v transakciji za generiranje sredstev. Težavnost naloge se dinamično prilagaja tako, da je v povprečju nov blok dodan vsakih 10 minut. Za dodajanje novega bloka mora rudar izračunati zgoščeno vrednost bloka, ki mora biti manjša od zahtevane vrednosti, podane v glavi prejšnjega bloka. To lahko doseže s spreminjanjem vrednosti nonce v glavi trenutnega bloka. Pri tem se uporablja zgoščevalni algoritem SHA-256 in zaradi njegovih lastnosti pri izračunu zgoščene vrednosti bloka ni bližnjic, temveč morajo rudarji z grobo silo poizkušati najti pravo polja nonce, da je za predlagan blok njegova zgoščena vrednost manjša od zahtevane. Težavnost se zviša v primeru, da je bil blok v verigo blokov dodan pred predvidenim časom in se po drugi strani zmanjša, če bloka v predvidenem času ni bilo. Ko rudarji

najdejo tak blok, jim je v interesu, da to čim prej sporočijo vsem ostalim udeležencem v omrežju, saj vsi želijo, da se veriga blokov nadaljuje na njihovem bloku. Le na ta način bodo upravičeni do nagrade v transakciji za generiranje sredstev, saj jo lahko uporabijo v transakciji šele 100 blokov po tem, ko je bil blok dodan v verigo blokov. Rudarjenje se je v začetku izvajalo na povsem običajnih osebnih računalnikih, dandanes pa to običajno počnejo podjetja in posamezniki s posebej prilagojeno strojno opremo ASIC (angl. Application Specific Integrated Circuits). Potek sodobnega rudarjenja kriptovalut je prikazan na sliki 2.10. Zaradi visoke porabe električne energije se industrija rudarjenja kriptovalut seli na področja s cenejšo električno energijo. Prav visoka poraba energije in njen ogljični odtis pa postajata vedno bolj okoljsko sporna [13, 14].

Slika 2.10: Slika iz podjetja, ki se ukvarja z rudarjenjem kriptovalut[40]

2.3.7 Doseganje soglasja

Ker je Bitcoin porazdeljen sistem, v katerega lahko vstopi vsakdo brez predhodne privolitve obstoječih članov, obstaja problem zaupanja med posame-

znimi vozlišči v omrežju. Kot smo že omenili v poglavju 2.3.2 , se v verigi (oz. drevesu) blokov občasno pojavijo posamezne veje, ki predstavljajo alternativno verzijo zgodovine. Vozlišča pa se morajo glede celotne zgodovine poenotiti, kljub temu, da drug za drugega ne vedo ali so poštena ali zlonamerna.

Za dosego porazdeljenega soglasja v omrežju moramo zadostiti naslednjim zahtevam [15]:

- strinjanje (vsa poštena vozlišča se strinjajo z neko vrednostjo),
- končni izid izvajanja (vsa poštena vozlišča bodo sprejela odločitev v doglednem času),
- veljavnost (transakcija, na podlagi katere je bilo doseženo soglasje, je bila predlagana s strani poštenega vozlišča),
- odpornost na napake (algoritem mora biti sposoben učinkovitega delovanja tudi v prisotnosti zlonamernih vozlišč),
- doslednost (vsako vozlišče sprejme odločitev le enkrat v ciklu za doseganje soglasja in je ne spreminja).

Problem doseganja soglasja v porazdeljenem sistemu pa ni nov, ampak že več desetletij star problem, imenovan problem bizantinskih generalov (angl. Byzantine generals problem) [16].

Problem bizantinskih generalov

Pri problemu bizantinskih generalov gre za skupino generalov. Vsak izmed njih vodi svoj del vojske in namerava osvojiti mesto. Na izbiro imajo dve možnosti: napad ali umik. Generali se morajo med seboj poenotiti in sprejeti odločitev o usklajenem napadu, saj bi napad le dela generalov prinesel njihov poraz. Problem nastane, ker se generali med seboj ne vidijo, saj so mesto obkolili vsak iz svoje strani. Komunicirajo lahko le prek nezanesljivih kurirjev,

ki prenašajo sporočila med njimi. Dodatno nevarnost predstavlja dejstvo, da so med generali lahko tudi izdajalci, ki želijo le povzročati zmedo z lažnimi sporočili. Sprva preprost problem se tako spremeni v zelo kompleksnega.

Problem lahko preslikamo v Bitcoin tako, da generali predstavljajo vozlišča v omrežju, kurirji predstavljajo komunikacijo med njimi, namesto napada pa se morajo poenotiti glede veljavnega stanja transakcij v omrežju. Problem se v omrežju Bitcoin reši z uporabo dokaza dela, kar smo podrobneje opisali v podpoglavju 2.3.6. Vsako vozlišče mora za dodajanje informacij v obliki bloka opraviti zahtevano delo, za kar porablja energijo. V kolikor bi zlonamerno vozlišče v blok vključilo neveljavne transakcije in blok (skupaj z dokazom dela) poslalo ostalim udeležencem v omrežju, bi vsa poštena vozlišča ta blok zavrnila in ga ne bi dodala v verigo blokov. Na ta način zlonamerna vozlišča nimajo ekonomskega razloga za ustvarjanje neveljavnih blokov.

Spreminjanje pravil protokola

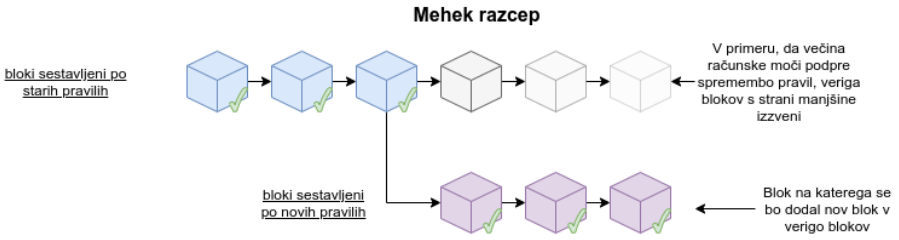
Kot omenjeno zgoraj, morajo vsa poštena vozlišča slediti pravilom istega protokola. Zaradi nadgradenj, popravkov napak ali drugih optimizacij si želimo pravila protokola občasno tudi spremeniti in prilagoditi. Ker gre za decentraliziran sistem, v katerem nimamo osrednje entitete, ki bi določala pravila in njihove popravke, gre pri tem za bolj zapleten proces. Pravila za veljavnost transakcij in blokov se lahko spremenijo na način, da ta postanejo bolj ali manj stroga [20].

- **mehki razcep** (angl. soft fork)

  Pri mehkem razcepu se pravila protokola spremenijo na način, da bi bili nekateri - pred spremembo veljavni bloki po spremembi neveljavni. Če se le del vozlišč odloči za nadgradnjo, bodo vozlišča, ki niso sprejela novih pravil še vedno lahko sprejemala bloke, ustvarjene z novimi pravili. Vendar pa bloki, ustvarjeni po starih pravilih, ne bodo označeni kot veljavni in pri vozliščih, ki upoštevajo novo verzijo protokola, ne bodo vključeni v verigo podatkov. Zato ta del vejitve za uveljavitev

novih pravil potrebuje podporo več kot polovice računske moči, ki jo prispevajo rudarji. V kolikor imajo spremenjena pravila večinsko podporo v računski moči, čez čas tudi rudarji, ki sprva niso podpirali spremembe, nadgradijo svojo programsko opremo, saj njihovi bloki niso več vključeni v verigo blokov in na ta način izgubijo svoj vir zaslužka, kar je razvidno tudi iz slike 2.11. Pri mehkem razcepu so vse spremembe združljive za nazaj (angl. backwards compatible) in vsa vozlišča lahko po potrebi lahko preidejo na verzijo protokola pred spremembo. Običajno se skupnost pred spremembo pravil poskuša uskladiti, in če predlog nima širše podpore, ne pride do njegove implementacije.

Najbolj znani mehki razcepi v protokolu Bitcoin so: SegWit (izveden leta 2017), SegWit2x (neuspešen), Taproot (napovedan za november 2021).



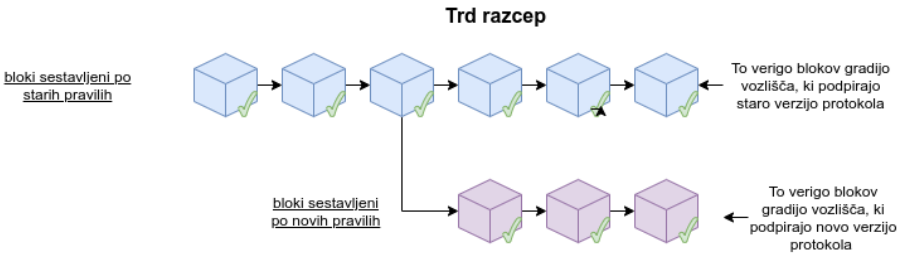
Slika 2.11: Prikaz mehkega razcepa

- **trdi razcep** (angl. hard fork)

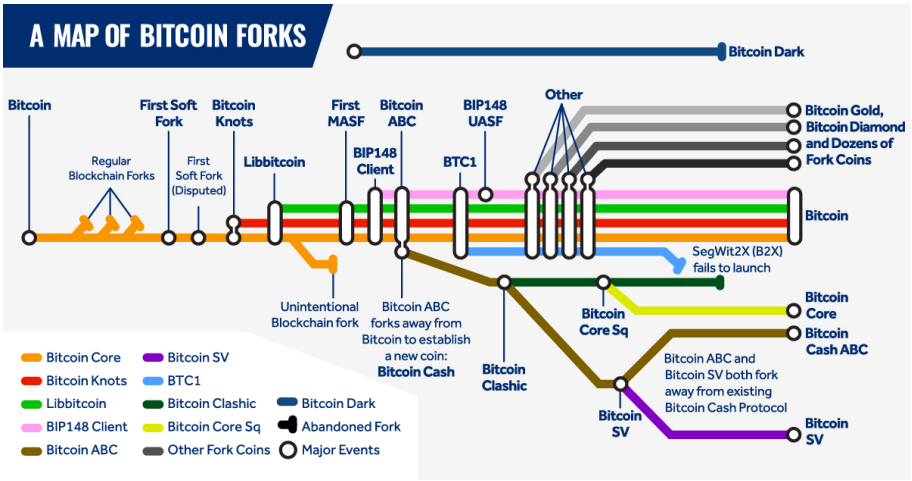
  Pri trdem razcepu se pravila protokola spremenijo tako, da so bloki ustvarjeni po spremembi protokola za vozlišča, ki protokola niso spremenila, neveljavni. Za to, da vozlišča v omrežju še naprej lahko gradijo verigo blokov in dosegajo soglasje, morajo vsa vozlišča v omrežju nadgraditi svojo programsko opremo. V primeru, da del vozlišč ostane na starem protokolu se ustvarita dve različni veji (angl. fork) verige blokov, kar je razvidno iz slike 2.12. Tako vsak del omrežja gradi in

dosega konsenz po svojih pravilih. Taka ločitev je nepovratna, saj verig blokov ni mogoče ponovno združiti. Na ta način lahko iz ene kriptovalute z novimi pravili nastane nova ločena kriptovaluta, obe pa imata do točke razcepa skupno zgodovino in tako imajo imetniki neporabljenih kovancev pred delitvijo svoje kovance obeh kriptovalut. Nove kriptovalute na ta način ni težko ustvariti, vendar je njena vrednost močno odvisna od števila uporabnikov in računske moči rudarjev. Sprememba pravil, ki ustvari nepovratne spremembe, pa ne vodi nujno v vejitev in novo kriptovaluto, saj se lahko vozlišča poenitijo in vsi sprejmejo nova pravila protokola.

Najbolj znan primer trdega razcepa je nastanek kriptovalute Bitcoin Cash v letu 2017, ko je prišlo do nestrinjanja v skupnosti glede povečanja velikosti blokov. Iz podobnih razlogov je kasneje iz kriptovalute Bitcoin Cash nastala tudi kriptovaluta Bitcoin SV. Na sliki 2.13 je prikazana celotna zgodovina razcepov protokola Bitcoin.



Slika 2.12: Prikaz trdega razcepa



Slika 2.13: Prikaz razcepov v protokolu Bitcoin.[41]

2.3.8 Decentralizacija

Decentralizacija pomeni odpravo centralne avtoritete, ki bi imela nadzor nad sistemom. Gre za eno ključnih prednosti kriptovalut, saj močno zmanjša tveganje za zlom sistema. Zaradi razpršenosti podatkov med vozlišči v omrežju tako posamezna točka okvare (angl. single point of failure) me obstaja več, ob enem pa tudi močno oteži kontrolo nad podatki in njihovo manipulacijo. Vsa vozlišča lahko preverjajo vse transakcije in se o stanju sistema odločajo z mehanizmi za doseganje soglasja, opisanimi v prejšnjem poglavju.

K višji stopnji decentralizacije prispevajo naslednji kriteriji [17, 18]:

- število vozlišč v omrežju (višje število vozlišč pomeni, da več udeležencev v sistemu preverja veljavnost transakcij in blokov ter večjo redundanco shranjenih podatkov),
- fizična razpršenost vozlišč v omrežju (poleg števila vozlišč je pomembna tudi široka fizična razpršitev vozlišč, saj tako npr. naravne nesreče na enem delu sveta ne morejo povzročiti izpada sistema),
- različne implementacije klientov, ki poganjajo vozlišča (uporaba več različnih implementacij klientov, ki sledijo istemu protokolu, je koristna

zaradi izogibanja morebitnim napakam v programski kodi. Če bi vsa vozlišča poganjala isto verzijo programske opreme z usodno napako, bi ta lahko povzročila izpad sistema),

- razpršenost računske moči, ki opravlja dokaz dela [19]:

  - **Ginijev koeficient**,

    Čeprav se običajno uporablja za izračun ekonomske neenakosti in razporeditve bogastva, ga lahko v svetu kriptovalut uporabimo za izračun neenakosti in razporeditve računske moči.

    Formula za izračun je: $G = \frac{\sum A_i, A_j \in A | NB_{A_i} - NB_{A_j} |}{2|A| \sum_{NB_{A_j} \in NB} NB_{A_j}}$. V formuli A predstavlja množico rudarjev, NB_{A_i} pa število blokov, ustvarjenih s strani posameznega rudarja.

    Nižji Ginijev koeficient pomeni, da mora več rudarjev sodelovati, da bi s tem povzročili težave v delovanju sistema in s tem nakazuje na višjo stopnjo decentralizacije.

  - **Informacijska entropija**,

    Višja vrednost entropije pomeni višjo stopnjo naključnosti v razporeditvi računske moči in s tem nakazje na višjo stopnjo decentralizacije. Izračunamo jo s pomočjo sledečih formul:

    $p_i = \frac{b_i}{-\sum_{i=1}^{n} b_i}$,
    $E = -\sum_{i=1}^{n} p_i log_2 p_i$

    V formulah b_i predstavlja število blokov, ki jih ustvari i-ti rudar, n pa predstavlja število rudarjev, za katere računamo entropijo.

  - **Nakamotov koeficient**.

    Nakamotov koeficient je definiran kot minimalno število entitet (oz. rudarjev) potrebnih za dosego 51 % računske moči v celotnem sistemu. Višji kot je koeficient, manjša je možnost za sodelovanje med potencialno škodljivimi rudarji, kar posledično pomeni višjo decentralizacijo in varnost sistema.

Izračunamo ga po formuli: $N = min\{k \in [1, ..., K] : \sum_{i=1}^{k} p_i \geq 0.51\}$. V formuli p_i predstavlja delež blokov ustvarjenih s strani posameznega rudarja.

Decentralizacija je poleg hitrosti in varnosti ena ključnih dimenzij trileme veriženja blokov (angl. blockchain trilemma) [21]. Gre za tri zelo zaželjene dimenzije v verigah blokov, žal pa velja, da do sedaj ni še nikomur uspelo maksimizirati dveh, ne da bi zmanjšal tretjo.

Poglavje 3

Kraje v omrežju Bitcoin

Psevdonimnost

Nakazila v omrežju Bitcoin se precej razlikujejo od digitalnih nakazil FIAT valut, saj pri slednjih državni regulatorji zahtevajo, da ima ponudnik določene identifikacijske podatke o plačniku ter prejemniku. Podobne zahteve so v svetu kriptovalut nepraktične, saj uporabniki lahko izvajajo nakazila neposredno med seboj, brez posredovanja institucij. Prav tako identiteti pošiljatelja in prejemnika nista nujno znani. Transakcije pa kljub vsemu niso anonimne, saj so vsem vozliščem v omrežju znani približen čas bloka, v katerega je transakcija vključena; naslovi med katerimi so bila prenešena sredstva in vrednost teh sredstev. Ker v verigi blokov ni zapisana identiteta uporabnikov, temveč le njihov javni naslov, Bitcoin transakcije veljajo za psevdonimne.

Zasebnost Bitcoin transakcij

Stopnja zasebnosti transakcij v omrežju je močno odvisna od upoštevanja priporočil za otežitev sledljivosti [22, 23]. Spodaj so našteta najpogosteje omenjena priporočila.

- **izogibanje ponovne uporabe naslovov**,

S ponovno uporabo Bitcoin naslovov tvegamo, da bo nekdo to ugotovil in posledično sklepal, da vse transakcije iz istega naslova pripadajo enemu uporabniku.

- **izogibanje pošiljanja zaokrožene vsote**,

  V transakcijah, s katerimi nekomu plačujemo zaokroženo vsoto kripto kovancev lahko običajno sklepamo, da je izhod z zaokroženo vsoto plačilo, izhod s preostankom kripto kovancev pa nakažemo nazaj sami sebi. S to hevristiko lahko povežemo izvoren naslov z naslovom, kamor je bil nakazan preostanek sredstev.

- **izogibanje vključevanja več naslovov v eno transakcijo**,

  V transakcijah so običajno vsi vhodi last istega uporabnika, saj ima zasebni ključ, s katerimi jih lahko odklene. Če je v transakciji veliko število vhodov, lahko s tega sklepamo, da vsi vhodni naslovi pripadajo isti entiteti.

- **uporaba specializiranih storitev za povečanje anonimnosti (angl. bitcoin mixers)**. [24]

  Za povečanje zasebnosti so uporabniki razvili namenske storitve. Med seboj se razlikujejo v načinih delovanja, njihov namen pa je to, da vhode različnih uporabnikov uporabijo v skupnih transakcijah in na ta način deloma zabrišejo sledi oz. jih speljejo na napačno pot. Ta tehnika povečevanja zasebnosti pa ni zastonj, saj običajno za storitev zahteva plačilo določenega dela kovancev, poleg tega pa morajo uporabniki plačati tudi strošek transakcije (angl. transaction fee), ki ga prejme rudar, ki transakcijo vključi v blok. Zaradi povečanja zasebnosti jih pogosto uporabljajo kriminalci v namene pranja denarja, kar je povzročilo njihovo omejevanje s strani nekaterih držav.

Stopnja kriminalnih aktivnost v omrežju Bitcoin

Psevdonimnost in nereguliranost trga je - poleg kriptografskih navdušencev - precej zgodaj pritegnila tudi kriminalce. Bitcoin uporabljajo za pranje denarja [25], plačevanje ilegalnih storitev in izdelkov [26] ter plačilo odkupnin [27].

Med najbolj znane primere kraj in uporabe bitcoina za ilegalne dejavnosti spadajo:

- **vdor v kripto menjalnico Mt. Gox**

  Vdor se je zgodil leta 2014 in je sprožil stečaj tedaj največje kripto menjalnice na svetu. Po nekaterih ocenah je bilo ukradenih okoli 740.000 bitcoinov, takrat vrednih približno 460 milijonov ameriških dolarjev.

- **illegalna spletna trgovina Silk Road**

  Ilegalna spletna trgovina Silk Road, dostopna preko globokega spleta (angl. deep web), na kateri je bilo mogoče do njenega zaprtja naročiti droge, orožje in celo umore je za plačilo storitev in izdelkov uporabljala kriptovaluto Bitcoin.

- **plačilo petih milijonov dolarjev za odkupnino podjetja Colonial Pipelines**

  Maja 2021 je ameriško podjetje Colonial Pipelines po vdoru v njihov informacijski sistem plačalo okrog petih milijonov dolarjev odkupnine.

- **vdor v slovensko podjetje NiceHash**

  Slovensko podjetje NiceHash, ki se ukvarja s posredovanjem računske moči pri rudarjenju kriptovalut, je bilo decembra 2017 žrtev ene najbolj odmevnih kraj v svetu kriptovalut. Ob vdoru so jim ukradli kriptovalute v vrednosti okrog 60 milijonov dolarjev. Po nekaterih podatkih gre za eno največjih kraj v zgodovini Slovenije.

Kljub temu, da je je bil del Bitcoin zgodovine močno povezan z ilegalnimi dejavnostmi in odmevnimi primeri kraj, pa je po novejših raziskavah del teh dejavnosti sedaj močno upadel. Po nekaterih ocenah [28] je bilo pred letom 2016 do 46 % transakcij povezanih z nezakonitimi posli. Delež je nato začel naglo upadati, saj se je uporaba kriptovalut bolj razširila med splošno populacijo. Poleg tega je zaradi boljšega dela organov pregona na tem področju postal vedno manj zanimiv za uporabo v ilegalnih dejavnostih. V letošnjem poročilu podjetja Chainalysis [29] je delež transakcij povezanih z ilegalnimi posli letu 2020 znašal 0,39 %. Kljub občutnemu zmanjšanju odstotka, pa se je povečalo število zahtev za odkupnine izsiljevalskih virusov, tako da tematika ostaja zelo pomembna.

Ilegalne dejavnosti se niso začele in se ne bodo končale zaradi kriptovalut. Njihova uporaba za tovrstna financiranja pa meče slabo luč na celotno industrijo. Zaradi globalizacije in uporabe interneta je tarča tovrstnih napadov lahko vsakdo, tako da je ključno, da se napadalce čim bolj omeji in se jim oteži izvajanje njihovih dejavnosti.

Omeniti je treba, da Bitcoin ni kriptovaluta s posebnim poudarkom na anonimnosti in zasebnosti. Na tem področju obstajajo druge kriptovalute (predvsem Monero in ZCash), ki s svojimi protokoli še bolj varujejo zasebnost uporabnikov. Sledenje transakcijam in odkrivanje ilegalnih dejavnosti je tam še bistveno težje, kot v omrežju Bitcoin.

Poglavje 4

Sledenje Bitcoin kovancem

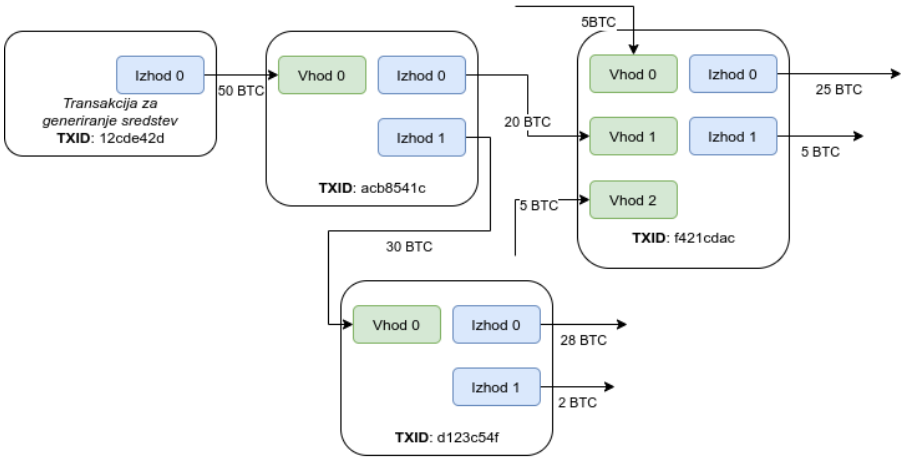
V tem poglavju bomo opisali sledenje označenim Bitcoin kovancem s pomočjo grafa transakcij in metode, ki smo jih pri tem uporabili.

Kot opisano v poglavju 2.3, so podatki o transakcijah v Bitcoin omrežju shranjeni v verigi blokov. Kljub temu, da so podatki o transakcijah prosto dostopni, pa pri njihovi analizi naletimo na več zahtevnih problemov.

Iz podatkov o transakcijah lahko sestavimo več različnih grafov [30]:

- **graf transakcij**,

  Graf transakcij predstavlja pretok kripto kovancev med transakcijami skozi čas. Vsako vozlišče predstavlja transakcijo, usmerjene povezave med vozlišči pa nam povedo smer in količino prenesenih kovancev.



Slika 4.1: Primer enostavnega grafa transakcij

- **graf naslovov**,

  Graf naslovov predstavlja pretok kripto kovancev med Bitcoin naslovi. Vsako vozlišče predstavlja svoj naslov, usmerjene povezave med njimi pa predstavljajo smer in količino prenesenih kovancev.

- **graf skupkov naslovov**.

  Graf skupkov naslovov (angl. address cluster graph) je zelo podoben grafu naslovov. Razlika med njima je, da pri tem grafu naslove združimo skupaj s pomočjo hevristike. Običajno je cilj združevanja naslovov, da združimo naslove, ki pripadajo isti entiteti. [31, 32].

Bitcoin za beleženje stanja uporablja model UTXO, kjer se v transakciji izhodi na novo ustvarijo, porabijo pa se izhodi (vhodi v to transakcijo), ki so bili ustvarjeni v predhodnih transakcijah. Zaradi tega, ker posameznim izhodom vhod, iz katerega prejmejo sredstva, ni točno določen, temveč mora biti le količina kovancev na vhodih višja ali enaka, kot na izhodih, v veliko primerih ne moremo enoznačno določiti, iz katerih vhodov izvira določen izhod. V opisanih metodah si zato pomagamo z različnimi hevristikami.

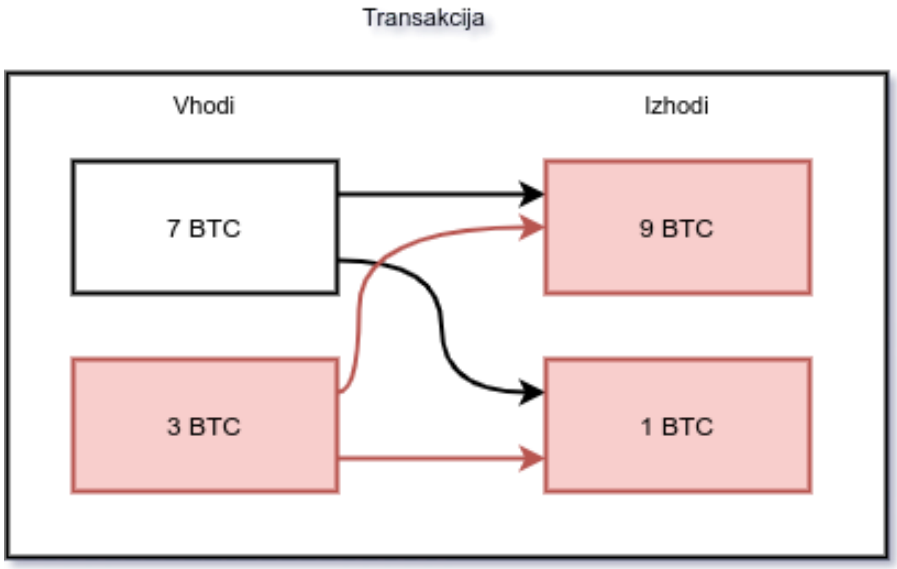
4.1 Metode za sledenje transakcijam

V prihodnjih podpoglavjih bomo opisali že predlagane metode iz članka [35], njihove nadgradnje in nove metode, ki smo jih razvili v okviru magistrske naloge.

4.1.1 Metoda Poison

Metoda Poison za sledenje in označevanje sumljivih kovancev uporablja strategijo, po kateri kot sumljive označimo vse izhode iz transakcije, kjer je bil en od vhodov predhodno označen kot sumljiv. Metoda pri označevanju zagotovo ne zgreši nobenega izhoda, saj je edini način, da označeni kovanci vplivajo na neoznačene ta, da so vhodi hkrati v isti transakciji. Kot je razvidno tudi iz slike 4.2, je težava te metode, da se število označenih kovancev med izvajanjem algoritma eksponentno povečuje. V praksi to predstavlja velik problem, saj grafi transakcij označenih kovancev eksponentno naraščajo, zato metoda v praksi ni zelo uporabna.

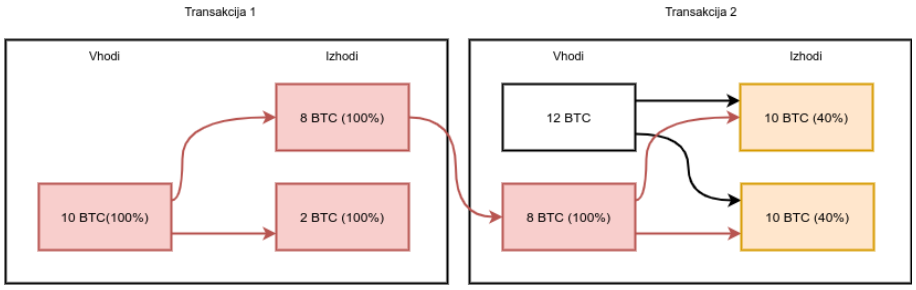
Poleg časovne in prostorske zahtevnosti je slabost metode tudi v tem, da potencialni napadalci lahko majhen del označenih kovancev pošljejo na različne naslove. V kolikor kdo od prejemnikov porabi to majhno prejeto vsoto kovancev, metoda sledi tudi vsem njihovim transakcijam, čeprav si običajno tega ne bi želeli.



Slika 4.2: Metoda Poison

4.1.2 Metoda Haircut

Metoda Haircut za označevanje potencialnih sumljivih izhodov transakcije uporablja strategijo, označitve izhodov kot delno sumljivih, glede na to kolikšen delež vhodov je bil označen kot sumljiv, kar je razvidno tudi iz slike 4.3. Metodo, opisano v članku [35], smo nadgradili z dodatnim parametrom, s katerim nastavljamo delež označenih kovancev, pri katerih izhode še označimo kot sumljive. Na ta način se izognemo potencialnemu napadu, kjer bi majhen delež označenih kovancev vplival na sledenje vseh naslednjih transakcij. Posledica tega je tudi, da se graf označenih kovancev ne širi več eksponentno in tako metoda postane bolj obvladljiva ter uporabna v praksi. Rezultate metode Haircut lahko uporabimo kot neko merilo, s katerimi lahko primerjamo rezultate drugih metod.

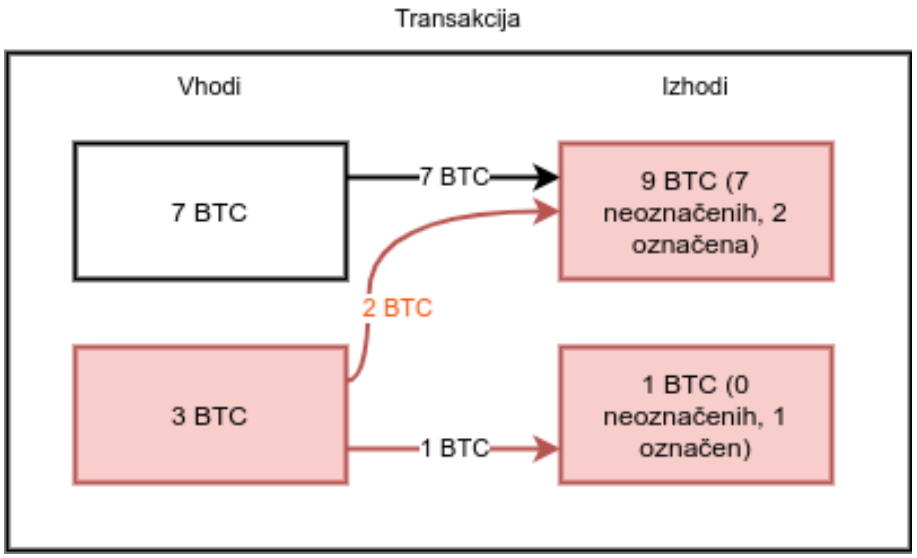


Slika 4.3: Metoda Haircut

4.1.3 Metoda FIFO

Metoda FIFO (angl. First In, First Out) je osnovana na konceptu upravljanja zalog in je ena od predstavnic metod, ki se pri distribuciji označenosti zanašajo na vrstni red vhodov in izhodov iz transakcij. Označbe bodo iz vhodov prenesene na izhode od prvega proti zadnjemu. Za razliko od zgoraj opisanih metod, lahko pri metodi FIFO v transakciji z označenim vhodom določene izhode označimo, druge pa pustimo neoznačene in jim ne sledimo več v grafu transakcij. Princip prvi noter, prvi ven se poleg te metode uporablja pogosto tudi v računovodstvu in pravu. Znan pravni primer, pri katerem je bila uporabljena ta metoda, je iz leta 1816 [37].

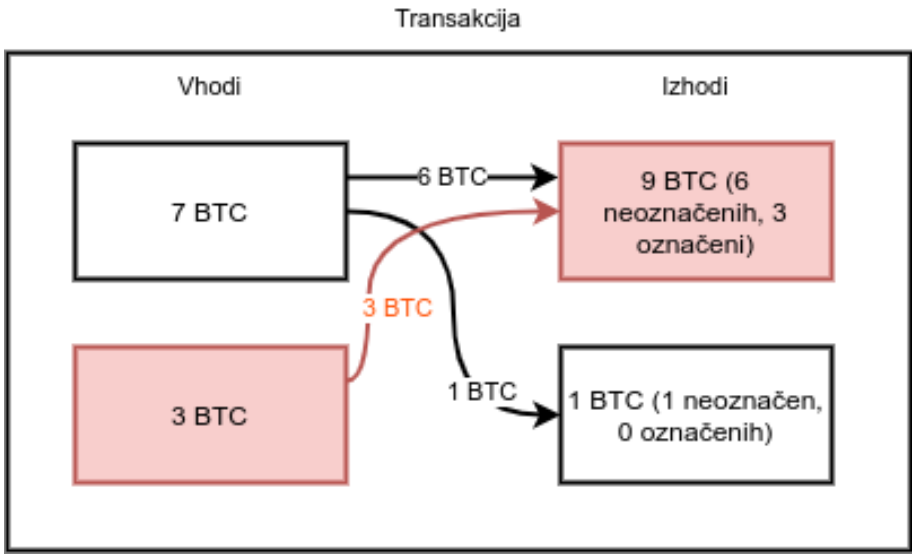
Kljub svojim številnim dobrim lastnostim in uporabi v pravu, pa ima uporaba metode FIFO v svetu Bitcoina določene pomanjkljivosti. Vrstni red vhodov in izhodov je ob kreiranju transakcije poljubno določen in je odvisen od programske opreme (običajno denarnice), kjer kreiramo transakcijo. Vrstni red pa nikakor ne vpliva na veljavnost transakcije in vključitev v blok.



Slika 4.4: Primer metode FIFO, kjer se vhodi v izhode preslikajo v vrstnem redu, kot so vstopili v transakcijo

4.1.4 Metoda LIFO

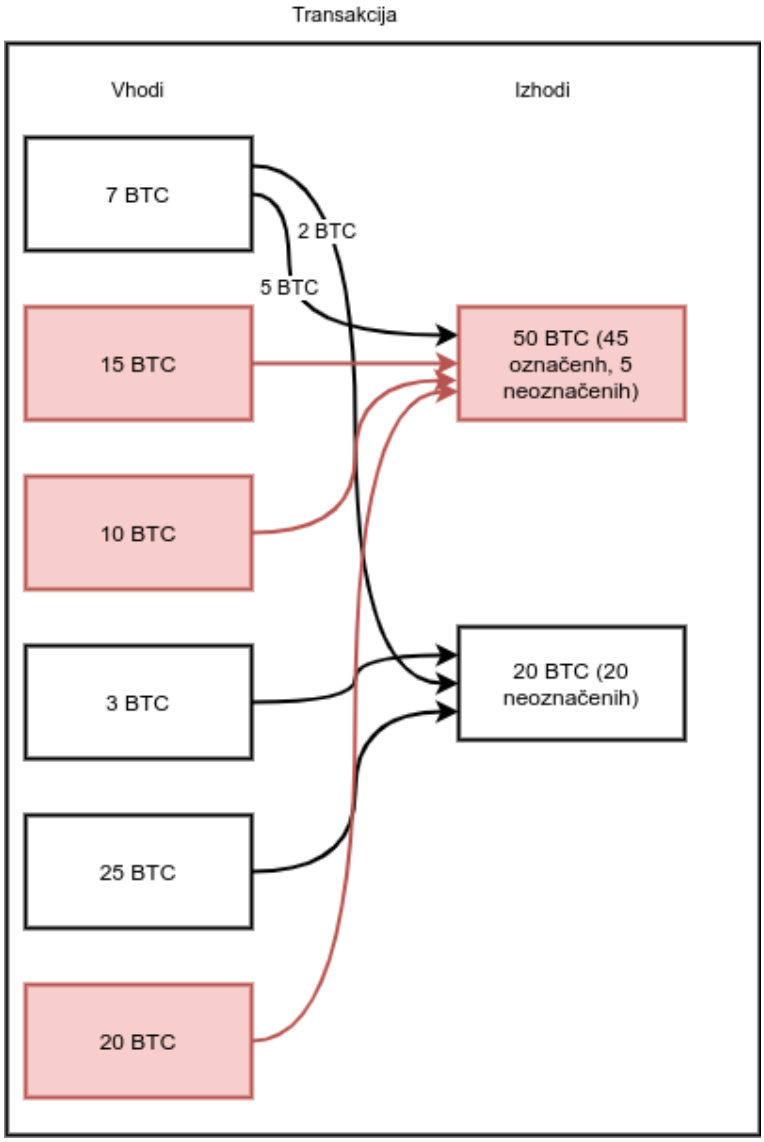
Metoda LIFO (angl. Last In, First Out) je zelo podobna zgoraj opisani metodi FIFO. Edina razlika med metodama je, da označene kovance iz vhodov pretvori v izhode v obratnem vrstnem redu. Velika podobnost obeh metod nam omogoča enostavno medsebojno primerjavo.



Slika 4.5: Primer metode LIFO, kjer se vhodi v izhode preslikajo v obratnem vrstnem redu, kot so vstopili v transakcijo

4.1.5 Metoda TIHO

Metoda TIHO (angl. Taint In, Highest Out) favorizira distribucijo vhodnih označenih kovancev v izhode z najvišjo vrednostjo. Ker upošteva vrednosti izhodov, in ne recimo vrstnega reda, kot metodi FIFO in LIFO, jo uvrščamo med metode, osnovane na vrednosti izhodov (angl. value based method). Osnovana je na predpostavki, da je izhod z večjo vrednostjo običajno glavni namen transakcije, ostali izhodi pa pogosto služijo kot vračilo ostanka denarja plačniku. Pomembno je omeniti, da ta predpostavka ne drži vedno, saj obstajajo tudi primeri, ko ima oseba v neporabljenem izhodu transakcije veliko kovancev in z njimi želi opraviti transakcijo, kjer mora plačati bistveno manj. V tem primeru bo najvišji izhod iz transakcije ravno vračilo na račun plačnika.



Slika 4.6: Primer metode TIHO, kjer označimo izhod z najvišjo vrednostjo

4.1.6 Metoda COMB

Implementirali smo tudi novo metodo, imenovano COMB. Metoda iz vrednosti vhodov in izhodov poizkuša s pomočjo kombinatorike poiskati podmnožice vhodov in izhodov znotraj transakcije, kjer se podmnožica vhodov lahko pre-

slika v podmnožico izhodov. Vedno obstaja interpretacija, kjer se vsi vhodi preslikajo v vse izhode. Z našo metodo pa poizkušamo poleg omenjene trivialne rešitve najti tudi alternativne interpretacije.

Glede na vrednosti vhodov in izhodov transakcije iz slike 4.7 so možne naslednje interpretacije:

- (Vhod1) → (Izhod1, Izhod2), (Vhod2) → (Izhod3, Izhod4)
- (Vhod1) → (Izhod3, Izhod2), (Vhod2) → (Izhod1, Izhod4)
- (Vhod1, Vhod2) → (Izhod1, Izhod2, Izhod3, Izhod4)

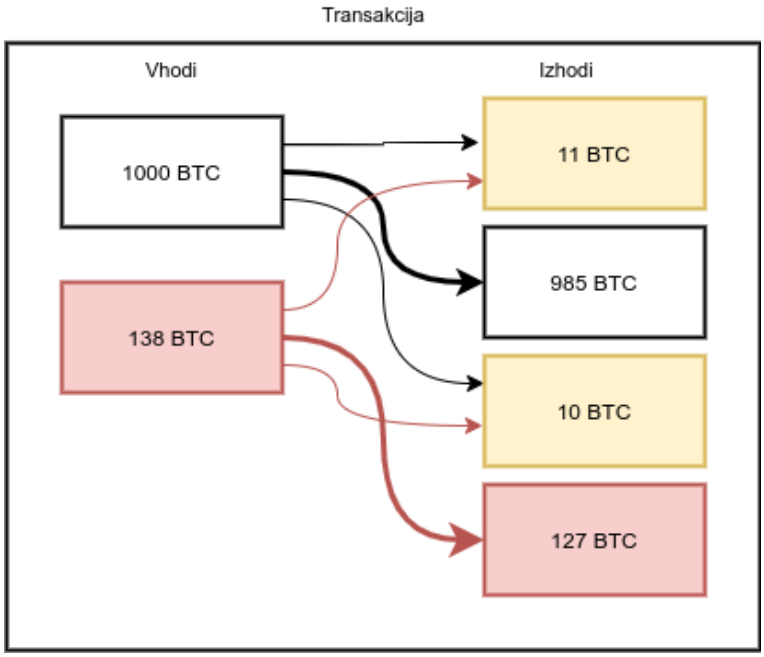
Iz naštetih interpretacij preslikav vhodov v izhode lahko sklepamo, da sta preslikavi Vhod1 → Izhod2 in Vhod2 → Izhod4 prisotni v vseh možnih kombinacijah preslikav podmnožic vhodov v podmnožice izhodov.

Metoda je računsko zelo zahtevna, saj mora tako za vhode kot izhode rešiti problem pokritja (angl. exact cover). Množica S predstavlja podmnožice X, za katero iščemo vsa možna pokritja. Z algoritmom poizkušamo najti take podmnožice S^*, da je vsak element iz množice X v natanko eni podmnožici S^*. Gre za NP-poln (angl. NP-complete) problem, in ga do sedaj nikomur ni uspelo rešiti v polinomskem času. Zaradi težavnosti smo se odločili, da opisani algoritem uporabimo le v transakcijah, ki nimajo več kot sedem vhodov ali izhodov. V transakcijah z več vhodi ali izhodi smo se odločili za enako označevanje izhodov, kot v metodi Haircut. Algoritem smo še dodatno omejili tako, da označenh izhodov z vrednostmi pod 10000 Satoshijev ne pregledujemo naprej, saj je njihova vrednost glede na trenutno ceno Bitcoina in povprečno provizijo za transakcijo relativno majhna.

Po izračunu vseh možnih točnih pokritij izhodov moramo za vsako tako pokritje preveriti katera točna pokrija izhodov ustrezajo naslednjim zahtevam:

- vsak vhod je porabljen točno enkrat,
- vsak izhod je vsebovan točno enkrat,

- vsota vhodov posamezne podmnožice je manjša ali enaka vsoti izhodov podmnožice v katero se preslika.



Slika 4.7: Primer metode COMB, kjer označimo izhod z najvišjo vrednostjo

4.2 Metrike za ocenjevanje metod

Za primerjavo metod bomo uporabili metrike iz grafa transakcij, ki predstavlja vse transakcije, ki izhajajo iz izbranega naključnega izhoda ob uporabi izbrane metode ter metrike, izračunane ob izvajanju.
Metrike, izračunane na podlagi grafa, ki predstavlja elemente grafa transakcij skozi katerega je šla določena metoda:

- **število vozlišč grafa transakcij**,
- **število povezav v grafu transakcij**,
- **gostota grafa**,

Gostota grafa je izračunana po formuli: $d = \frac{m}{n(n-1)}$, kjer n predstavlja število vozlišč, m pa število povezav med njimi.

- **povprečna stopnja vozlišča**,

  Povprečno stopnjo vozlišča izračunamo po formuli: $\overline{deg} = \frac{2m}{n}$, kjer m predstavlja število povezav, n pa število vozlišč.

- **koeficient nakopičenosti**,

  Koeficient nakopičenosti grafa je izračunan po formuli: $C = \frac{1}{n}\sum_{v \in G} c_v$, kjer n predstavlja število vozlišč, G graf in c_v delež možnih trikotnikov, ki vključujejo to vozlišče. Formula za njihov izračun je sledeča: $c_u = \frac{2T(u)}{deg(u)(deg(u)-1)}$. $T(u)$ predstavlja število trikotnikov, ki vključujejo vozlišče u, $deg(u)$ pa stopnjo vozlišča u.

- **povprečna razdalja**,

  Ker grafi transakcij lahko vsebujejo zelo veliko vozlišč n in povezav m, je zahtevnost izračuna enaka $\mathcal{O}(n * m)$. Standardna praksa v takih primerih je, da izberemo določeno število vozlišč (v našem primeru 100) in izračunamo povprečno razdaljo med vsemi vozlišči. Za izračun najkrajše poti smo uporabili Dijkstrov algoritem.

- **premer**,

  Tudi pri premeru grafa se srečamo z enako časovno zahtevnostjo kot pri izračunu povprečne razdalje. Zato smo tako kot pri izračunu povprečne razdalje uporabili 100 naključno izbranih vozlišč in izbrali najdaljšo med najkrajšimi potmi med izbranimi vozlišči in vsemi preostalimi vozlišči v grafu.

- **modularnost**,

  Modularnost smo izračunali po formuli $Q = \sum_{c=1}^{n} \left[\frac{L_c}{m} - \gamma \left(\frac{k_c}{2m} \right)^2 \right]$. V formuli c predstavlja skupnosti, m predstavlja število povezav, L_c število povezav znotraj skupnosti c, k_c vsoto stopenj vozlišč v skupnosti c, γ

pa resolucijski parameter. Za izračun metrike moramo najprej odkriti skupnosti v grafu, kar smo storili z uporabo Louvainove metode [33].

- **najvišja središčnost vozlišča**,

  Središčnost stopnje vozlišča (angl. degree centrality) je izračunana kot delež vozlišč, s katerimi je trenutno vozlišče povezano. Med vrednostmi vseh vozlišč nas najbolj zanima najvišja vrednost.

- **koeficient mešanja**,

  Koeficient mešanja meri podobnost povezanih vozlišč v grafu z ozirom na njihovo stopnjo. Ob tem lahko upoštevamo le vhodni/izhodni stopnji začetnega in končnega vozlišča. S kombinacijo teh dveh paramtrov tako dobimo štiri različne metrike. Za izračun uporabimo naslednjo formulo: $r = \frac{\sum_{xy} xy(e_{xy} - a_x b_y)}{\sigma_a \sigma_b}$. a_x in b_y predstavljata delež povezav, ki se začnejo in končajo v vozliščih z vrednostima x in y. Matrika e predstavlja deleže povezav za posamezna vozlišča. Zanjo veljata naslednji enačbi: $\sum_y e_{xy} = a_x$ in $\sum_x e_{xy} = b_x$. Kot je razvidno iz zgornje formule, se za izračun uporablja Pearsonova korelacija.

- **potenčni zakon**.

  Pri izračunu potenčnega zakona izraunamo porazdelitev stopenj vozlišč po formuli: $p_k = \frac{\#k}{n}$. $\#k$ predstavlja število vozlišč, ki so stopnje k, n pa predstavlja število vseh vozlišč. V številnih omrežjih ta porazdelitev sledi potenčnemu zakonu (angl. power law) po formuli: $p_k \approx k^{\gamma}$. Z algoritmom ocene največje vernosti(angl. maximum likelihood estimation) ocenimo koeficient γ, ki običajno leži na intervalu $[2, 3]$. Omrežjem, kjer porazdelitev stopenj sledi potenčnemu zakonu, pravimo tudi brezlestvična omrežja (angl. scale free networks).

Poleg naštetih metrik, ki so med grafi pogoste, pa smo se odločili primerjati tudi nekatere druge metrike, povezane z metodami za sledenje Bitcoin kovancem:

- **čas izvajanja**,

  Zaradi velikega števila transakcij nas poleg drugih metrik zanima tudi čas, potreben za dokončanje izvajanja določenega algoritma.

- **razmerje med količino označenih kovancev pred začetkom in po koncu izvajanja metode**,

  Razmerje med količino označenih kovancev pred in po izvajanju algoritma je ena ključnih metrik za ocenjevanje uspešnosti. Razmerje želimo ohraniti čim bližje 1, saj bi to pomenilo, da smo označili vedno enako število kovancev. Ni pa to edina metrika, ki jo moramo upoštevati, saj lahko tudi enako število označenih kovancev prinese slab rezultat, če smo označili napačne.

- **število označenih neporabljenih izhodov transakcij**,

  V tej metriki nas zanima, koliko različnih neporabljenih izhodov vsebuje označene kovance, kar nam pove, kako široko se je razširil graf in kako zahtevno bo preiskovanje v prihodnosti, ko bodo ti neporabljeni izhodi uporabljeni v novih transakcijah.

- **vrednosti v neporabljenih izhodih transakcij**.

  Poleg metrike, ki nam pove število neporabljenih izhodov, pa nas zanima tudi razporeditev njihovih vrednosti.

Poglavje 5

Podatki

5.1 Generiranje podatkove baze

Kljub temu, da so podatki o Bitcoin transakcijah javno dostopni in jih vsakdo lahko prenese na svoj računalnik, pa ti podatki niso v obliki, ki bi bila primerna za izvajanje metod sledenja označenim kovancem, opisanih v poglavju 4.1. V naslednjih podpoglavjih bomo opisali postopek, s katerim smo podatke pretvorili v želeno obliko.

5.1.1 Pridobivanje podatkov

Pri pridobivanju podatkov smo se odločili, da podatke o Bitcoin transakcijah prenesemo z Bitcoin Core klientom[1]. S pomočjo klienta se povežemo z ostalimi vozlišči v omrežju in prenesemo celotno verigo blokov (ang. initial block download). Vse transakcije in bloke ob prejemu tudi validiramo. Preneseni podatki se shranijo v izbran direktorij v obliki binarnih blkXXXXX.dat datotek. Posamezne datoteke so manjše od 128 MB, skupna velikost prenesenih podatkov pa presega 350 GB. Podatki vključujejo vse bloke in transakcije od prvega bloka iz leta 2009, do blokov iz avgusta 2021.

[1] `https://bitcoin.org/en/bitcoin-core/`

5.1.2 Branje podatkov

Ko smo uspešno prenesli vse podatke, smo se lotili njihovega branja. Podatki so v datotekah shranjeni v binarni obliki z vnaprej določeno strukturo, razvidno tudi iz slike 2.5.

Za branje prenesenih datotek nismo našli odprtokodnega programa, ki bi ustrezal našim zahtevam, zato smo se odločili napisati lasten program. Izbrali smo jezik C++, saj je glede na velikost podatkovne baze zelo pomembna hitrost branja in obdelovanja podatkov.

Program prebere podatke jih shrani v razrede *Block, Tx, TxIn* in *TxOut*. Pri branju smo morali biti zelo pozorni, saj so nekateri podatki napisani po pravilu malega konca (angl. little endian), drugi pa po pravilu velikega konca (angl. big endian). Poleg tega je bilo treba izračunati tudi zgoščene vrednosti transakcij, saj se te uporabljajo za njihovo identifikacijo. To vrednost imenujemo TXID. Ker se jih da z zgoščevalno funkcijo izračunati iz transakcij samih, njihove vrednosti niso zapisane v verigi blokov. Za pridobitev TXID transakcije je treba njeno vsebino zgostiti z uporabo algoritma SHA256 in dobljeno vrednost še enkrat zgostiti z istim algoritmom. V programu smo si pomagali z uporabo knjižnice "picosha2.h"[2]

$TXID = SHA_{256}(SHA_{256}(podatki\_transakcije))$

Dvojna zgoščena vrednost je bila predlagana v knjigi "Cryptography Engineering" [34] in služi kot zaščita pred napadom podaljševanja dolžine (angl. length extension attack).

Dodaten problem pri vstavljanju podatkov v podatkovno bazo je predstavljalo dejstvo, da nam zapisi v datotekah blkXXXXX.dat ne zagotavljajo pravilnega vrstnega reda blokov. Do tega je prišlo zaradi optimizacije hitrosti prenosa verige podatkov s strani izbranega klienta, saj lahko dobljen blok zapišemo na disk, še preden do nas pride njegov predhodnik. Za vstavljanje v našo podatkovno bazo potrebujemo bloke v enakem vrstnem redu, kot so zapisani v verigi blokov, saj se transakcije kasnejših blokov sklicujejo na transakcije iz predhodnih blokov. Zaradi velike količine podatkov vseh seveda ne

[2] `https://github.com/okdshin/PicoSHA2`

moremo hraniti v bralno-pisalnem spominu. Pri rešitvi tega problema sta nam pomagali naslednji dve dejstvi. Bloki v prenesenih datotekah nikoli niso zamaknjeni s svojega mesta za več kot 1024 mest in velikost blokov je omejena na 1 MB. Algoritem začne z branjem blokov in ko naleti na blok, pri katerem je vrednost prejšnje zgoščene vrednosti bloka (angl. previous block hash) enaka zgoščeni vrednosti bloka, ki smo ga dodali nazadnje, ga zapišemo v podatkovno bazo. Če se vrednosti ne ujemata, pa blok ohranimo v bralno-pisalnem pomnilniku in po vsaki iteraciji preverimo, če kateri od teh blokov ustreza prej opisanemu pogoju. Ko blok zapišemo v podatkovno bazo, lahko sprostimo spomin na bralno-pisalnem pomnilniku, ki ga ta zaseda. To nam zagotavlja, da zaradi ohranjanja blokov, ki so na disk zapisani v napačnem vrstnem redu, v pomnilniku zasedenost tega ne preseže 1024 MB.

5.1.3 Ustvarjanje podatkovne baze

Transakcije so v Bitcoin verigi blokov shranjene na način, da se nove transakcije pri svojih vhodih sklicujejo na prejšnje transakcije. Tak način hranjenja je zelo smiseln za verigo blokov, saj se bloki vedno le dodajajo in za preverjanje vhodov novih transakcij lahko hitro dostopamo do prejšnjih izhodov. Pri sledenju kovancem v grafu transakcij pa moramo graf preiskovati v obratni smeri, za kar bi ob obstoječi podatkovni strukturi porabili ogromno časa in računske moči.

Za shranjevanje smo uporabili podatkovno bazo, ki hrani pare ključ-vrednost (angl. key-value database). Odločili smo se za implementacijo RocksDB[3]. Gre za odportokodno podatkovno bazo razvito s strani podjetja Facebook. RocksDB je naslednjica podatkovne baze LevelDB, katero uporablja tudi več implementacij Bitcoin klienta (med drugimi tudi najbolj priljubljena implementacija Bitcoin Core). Podatkovna baza hrani le ključe in vrednosti in tako ne potrebuje vnaprej definirane strukture. Za izboljšanje hitrosti izkorišča velike hitrosti branja in pisanja v bralno-pisalni pomnilnik in je prilagojena za uporabo negibljivih diskov (angl. solid state drive).

[3] `http://rocksdb.org/`

Kljub temu da sama baza ne potrebuje strukture (za razliko od večine SQL) podatkovnih baz pa je bilo treba določiti, katere podatke in pod katerimi ključi bomo shranili v bazo. Podrobnosti so razvidne iz slike 5.1 in opisane v naslednjem seznamu.

Key	Value	Name
"b"+block_hash	block_height, previous_block_hash	Block
TxID	TxC	TxIDTxC
TxC	TxID, output_count, block_hash, block_height, vector<inputs>	Transaction
"i"+TxC+"."+N	TxC	Input
"o"+TxC+"."+N	value, script_length	Output

Slika 5.1: Struktura podatkovne baze

- **Block**

  Uporabimo ključ *"b"+blockHash* in pod ta ključ za določen blok zapišemo višino bloka in zgoščeno vrednost prejšnjega bloka.

- **TxIDTxC**

  Ko smo vstavili blok, se lotimo vstavljanja transakcij. V metodah za sledenje označenim kovancem potrebujemo zaporedno številko transakcije, hkrati pa nam zaporedna številka transakcije lahko služi kot identifikator namesto TXID in tako prihranimo prostor v podatkovni bazi. Ker pa vseeno potrebujemo povezavo med TXID in zaporedno številko transakcije, pa to shranimo v podatkovno bazo. TXID predstavlja ključ, vrednost pa je zaporedna številka transakcije (TxC). Števec, ki beleži transakcije po vsakem vstavljanju povečamo in si tako zagotovimo, da so transakcije označene v pravilnem vrstnem redu.

- **Transaction**

  Do transakcije želimo v podatkovni bazi dostopati preko zaporedne številke transakcije, zato jo uporabimo kot ključ. Za vrednost pa želimo imeti na voljo vse relevantne podatke o transakciji, zato ta vsebuje njeno identifikacijsko vrednost (TXID), število vhodov, zgoščeno vrednost bloka v katerega je bila vključena, višino bloka v katerega je bila vključena, in seznam vhodov v obliki *Output*, ki nam povejo, kateri izhodi so bili v transakciji uporabljeni kot vhodi.

- **Input**

  Polje Input vsebuje ključ v obliki *"i"+TxC+"."+N*, kjer *TxC* predstavlja zaporedno številko transakcije, kjer je bil ustvarjen izhod, *N* pa zaporedno število izhoda. Vrednost polja vsebuje zaporedno številko transakcije, kjer je bil ta izhod porabljen.

- **Output**

  Polje Input vsebuje ključ v obliki *"o"+TxC+"."+N*, kjer *TxC* predstavlja zaporedno številko transakcije, *N* pa zaporedno število izhoda. Vrednost polja predstavlja vrednost, preneseno s tem izhodom in dolžino skripte za zaklepanje. Iz dolžine skripte za zaklepanje lahko sklepamo tip skripte, ki je bil uporabljen.

Podatke je bilo potrebno pred zapisom v bazo serializirati in jih ob branju tudi deserializirati. Pri tem smo si pomagali s knjižnico cereal[4].

Kljub vsem optimizacijam podatkov (od prve Bitcoin transakcije pa vse do 7. 7. 2021) je podatkovna baza zasedla 432,7 GB prostora na disku. Zaradi visoke porabe prostora in posledično višjih časih v poizvedbah ter izbiri testnih podatkov (več o tem v poglavju 5.2), smo se odločili, da se omejimo na krajše časovno obdobje. Izbrali smo obdobje med blokoma 550.000 (14. 11. 2018) in 690.000 (7. 7. 2021).

[4] `https://uscilab.github.io/cereal/`

Ker izbrano časovno obdobje ne sovpada z začetkom verige blokov, smo za optimizacijo baze morali prilagoditi vstavljanje z naslednjimi optimizacijami:

- v bazo ne vstavljamo blokov (in njihovih transakcij) po določeni višini,
- iz baze izbrišemo vse izhode, ki so bili porabljeni v transakcijah pred izbrano višino,
- v bazo ne vstavljamo podatkov o transakcijah pred izbrano višino bloka. (še vedno pa moramo vstaviti podatke o vseh izhodih, saj so ti lahko porabljeni v izbranem časovnem obdobju)

Na ta način smo velikost podatkovne baze zmanjšali na 195,2 GB.

Branje in vstavljanje v podatkovno bazo smo izvajali na računalniku Lenovo T580 s 16 GB bralno-pisalnega pomnilnika in procesorjem Intel® Core™ i5-8250U CPU @ 1.60GHz × 8. Za branje in pisanje pa sem uporabil disk Samsung SSD T5.

5.2 Izbira testnih podatkov

V nalogi poizkušamo slediti označenim kovancem v primerih različnih zlorab in kraj, ki so podrobneje opisane v poglavju 3. Za pridobitev podatkov o transakcijah, ki so bile vključene v različne zlorabe, smo si pomagali s podatki iz storitve BitcoinAbuse.com[5]. Gre za storitev, kamor oškodovanci prijavijo Bitcoin naslove, ki so bili udeleženi v potencialno kriminalnih dejavnostih (npr. naslov, na katerega so zahtevali nakazilo odkupnine). Ker podatki niso preverjeni s strani institucije, ki bi ji lahko zaupali, smo se odločili uporabiti le naslove, ki jih je prijavilo najmanj 20 uporabnikov.

S pomočjo programskega vmesnika (angl. application program interface, API) BlockCypher[6] smo iz naslovov pridobili transakcije, nakazane na izbrane račune v izbranem obdobju. Transakcije skupaj s številko vhoda predstavljajo vhod razvite metode za sledenje označenim kovancem.

[5] `https://www.bitcoinabuse.com/`

[6] `https://www.blockcypher.com/`

Kot kontrolne transakcije smo izbrali naključne transakcije iz istega časovnega obdobja. Glede na poročilo podjetja Chainalysis [29] imamo pri naključni izbiri okrog 1 % možnost, da smo pri izbiri naključne transakcije izbrali tako, ki je del kriminalne dejavnosti. V primeru izbire časovnega okna v zgodnejšem obdobju, pa bi bila ta verjetnost precej višja, kar bi lahko bistveno vplivalo na rezultate.

Za namen primerjave metod smo pripravili množico 400 izhodov transakcij, za katere sumimo, da so vključeni v ilegalne aktivnost. V kontrolni skupini pa smo izbrali 800 izhodov transakcij, ki so bili ustvarjeni in porabljeni v istem časovnem obdobju.

5.3 Drugi podatki

Podatki, opisani do sedaj, izhajajo izključno iz verige blokov. V metodah pa lahko uporabimo tudi informacije, ki niso del verige blokov(angl. off chain data). Taki podatki lahko vsebujejo informacije o specifičnih entitetah, povezanih z določeno transakcijo, o lastnikih določenih naslovov itd.

5.3.1 Zbiranje podatkov

Odločili smo se, da zberemo podatke o transakcijah, ki so povezane z določenimi kriptomenjalnicami. Na ta način lahko ob sledenju Bitcoin kovancem skozi omrežje zaznamo, kdaj pride del sredstev do kripto menjalnice. Zaradi zakonov o preprečevanju pranja denarja in financiranja terorizma [36] so kripto menjalnice obvezane k hranjenju informacij, ki povezujejo transakcije z identitetami imetnikov kriptokovancev in prepoznavaju sumljivih dejavnosti. V Evropski uniji je bila leta 2018 sprejeta direktiva AMLD5[7] (angl. Anti Money Laundering Directive 5). V primeru, da med preiskovanjem grafa transakcij naletimo na transakcijo, ki izvira iz kriptomenjalnice, lahko preiskovalni organi od kripto menjalnice zahtevajo razkritje identitete pošiljatelja oz. prejemnika transakcije.

[7] `https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex%3A32018L0843`

Za pridobivanje podatkov smo izbrali spletno stran WalletExplorer.com[8], kjer so navedene transakcije povezane s posameznimi kripto menjalnicami. Zaradi obsežnosti podatkov smo se odločili, da se omejimo na kripto menjalnice Huobi[9], Bittrex[10], Poloniex[11] in Kraken[12], saj imamo o njih dovolj podatkov, poleg tega pa vse naštete spadajo med 20 največjih kripto menjalnic na svetu. Za dodatno omejitev velikosti podatkov smo podatke o transakcijah tudi časovno omejili na enak okvir kot pri podatkovni bazi, kjer imamo vse podatke iz verige blokov.

5.3.2 Uporaba v metodah za sledenje kovancem

Prenešene podatke o transakcijah, povezanih s posamezno kripto menjalnico, smo vstavili v RocksDB podatkovno bazo. Vsem metodam smo dodali nov način izvajanja, kjer ob pregledovanju posamezne transakcije preverimo tudi, če je le-ta povezana s katero od zgoraj naštetih kripto menjalnic. V kolikor je to res, od te transakcije naprej ne preiskujemo več grafa transakcij, temveč to transakcijo skupaj z imenom kripto menjalnice zapišemo v ločeno datoteko. To nam omogoča, da preiskovanje ustavimo v vejah, ki dosežejo regulirane ponudnike kripto storitev in ne raziskujemo grafa transakcij po nepotrebnem.

Za potrebe testiranja smo pripravili podatkovno bazo z 2.602.055 transakcijami, povezanimi z zgoraj omenjenimi menjalnicami. Kljub temu, da je absolutno število zbranih transakcij dokaj veliko, pa te predstavljajo le manjši delež transakcij, za katere regulatorji od ponudnikov zahtevajo poznavanje svojih uporabnikov in jih zaradi tega lahko tudi identificirajo. V kolikor bi v podatkovni bazi imeli transakcije vseh kripto menjalnic in ostalih reguliranih subjektov na trgu kriptovalut, bi metode lahko delovale bolj učinkovito, saj bi jim s tem dodatno omejili preiskovalni prostor.

[8] `https://www.walletexplorer.com/`
[9] `https://www.huobi.com`
[10] `https://global.bittrex.com/`
[11] `https://poloniex.com/`
[12] `https://www.kraken.com/`

Poglavje 6

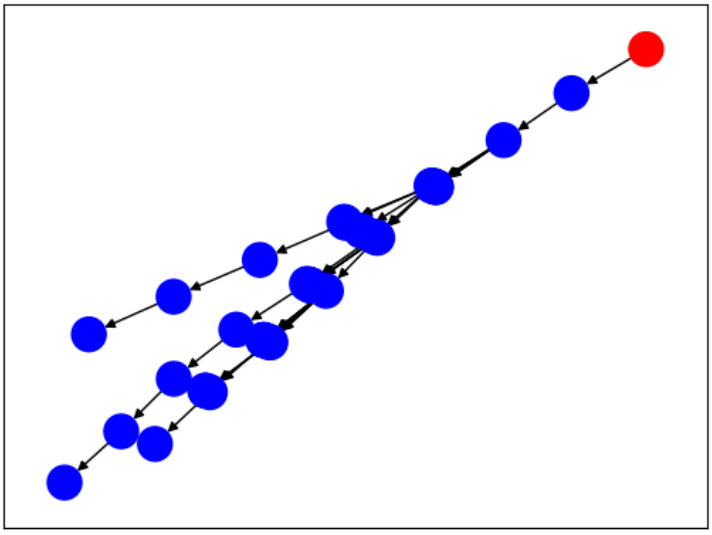
Rezultati

Glavni namen magistrske naloge je primerjava metod opisanih v podpoglavju 4.1. Vse opisane metode smo implementirali in na pripravljeni podatkovni bazi, z dvema testnima množicama, testirali njihovo delovanje. V tem poglavju smo metode primerjali glede na različne metrike. Iz analize smo izvzeli metodo Poison, ki zaradi eksponentne rasti števila vozlišč daje v mnogo primerih precej neuporabne rezultate, ki bi za svojo analizo potrebovali ogromno računske moči in časa. V analizi smo metodo Haircut, ki ji lahko prilagajamo vrednost, pod katero želimo prenehati slediti transakcijam skozi omrežje, uporabili z vrednostima $h = 0{,}1$ in $h = 0{,}05$. Vse metode smo pognali na 400 izbranih sumljivih izhodih transakcij, katerih pridobivanje smo podrobneje opisali v poglavju 5.2 in na 800 naključno izbranih izhodih transakcij iz istega obdobja.

V naslednjih podpoglavjih bomo predstavili rezultate. Osredotočili smo se predvsem na primerjavo med različnimi metodami in na razlike, ki so nastale med vzorcema sumljivih in naključno izbranih izhodov transakcij.

Primerjava s pomočjo grafov transakcij

Opisane metode s svojimi hevristikami iz označenih vhodov določijo označene izhode, ki jim nato sledimo v transakcije, kjer so porabljeni. Na ta način obiskujemo vozlišča v grafu transakcij, dokler obstajajo izhodi, ki jim še lahko sledimo (so porabljeni v blokih, ki jih imamo shranjene v podatkovni bazi in so glede na hevristiko izbrane metode zanimivi za nadaljnje sledenje). Vsem metodam smo dodali možnost, izvoza grafa, ki ga na ta način obiščejo (primer grafa je na sliki 6.1). Lastnosti teh grafov glede na uporabljeno metodo in na izbrani vzorec bomo predstavili v tem podpoglavju.



Slika 6.1: Primer grafa metode COMB z začetnim vozliščem v transakciji (rdeča točka v grafu) $40E51FB828C5A2E1327439F38767F90ECC2F62ECE8748699A164601C7C12C8D00$

Na vseh grafih smo s pomočjo knjižnice NetworkX[1] izračunali metrike, opisane v podpoglavju 4.2. Vse rezultate smo zbrali v tabeli in iz njih s pomočjo statističnih metod poskušali določiti metode, ki ustvarjajo grafe,

[1] `https://networkx.org/`

ki imajo statistično različne lastnosti med sumljivimi in naključno izbranimi transakcijami.

Ker standardne deviacije vzorcev niso enake in njihova porazdelitev ni normalna, smo se odločili za uporabo Mann-Whitney U testa (imenovanega tudi Mann-Whitney-Wilcoxonov test). Gre za neparametričen test z ničelno hipotezo, da je pri naključno izbranih vrednostih X in Y iz dveh populacij (vsaka vrednost iz svojega vzorca) verjetnost, da je vrednost X večja od vrednosti Y enaka verjetnosti, da je vrednost Y večja od vrednosti X.

Mann-Whitney U test je definiran s formulo:

$U = \sum_{i=1}^{n} \sum_{j=1}^{m} S(X_i, Y_j)$. Pri čemer izračunamo $S(X, Y)$ z uporabo formule:

$$S(X,Y) = \begin{cases} 1; X > Y, \\ 0.5; X = Y, \\ 0; X < Y \end{cases}$$

Za mejno vrednost p smo si izbrali vrednost 0,001. Pri tej vrednosti smo na podlagi tabele in izračunane vrednosti U dobili naslednje rezultate.

- Med vzorcema je statistično signifikantna razlika pri številu vozlišč za metodo Haircut (pri obeh vrednostih parametra) in pri metodi COMB. Grafi, ki izhajajo iz naključnega vzorca imajo v povprečju večje število vozlišč.

- Zelo podobna situacija je v primeru števila povezav, saj imajo grafi iz metode Haircut statistično signifikantno več povezav v vzorcu naključno izbranih izhodov transakcij. Grafi, ustvarjeni z metodo COMB, pa ravno obratno.

- Vzorci iz istih treh metod kažejo statistično signifikantno razliko tudi pri gostoti grafa. V povprečju imajo grafi sumljivih transakcij višjo gostoto.

- Pri povprečni stopnji vozlišča se povprečji vzorcev razlikujeta pri metodah Haircut (oba parametra), COMB in TIHO.

Pri metodi Haircut ($h = 0{,}1$; $h = 0{,}05$) imajo grafi iz vzorca naključnih izhodov v povprečju višjo stopnjo vozlišč. Pri metodah COMB in TIHO pa je situacija ravno obratna.

- Koeficient nakopičenosti se statistično razlikuje v metodi Haircut ($h = 0{,}1$; $h = 0{,}05$) in metodi TIHO. V obeh primerih je ta v povprečju večji pri vzorcih iz naključno izbranih izhodov transakcije.

- Povprečna razdalja med vozlišči je večja pri naključno izbranih vzorcih za metodi Haircut ($h = 0{,}1$; $h = 0{,}05$) in COMB.

- Premer je v povprečju statistično signifikantno višji pri naključno izbranih transakcijah za grafe, ki so rezultat metod Haircut ($h = 0{,}1$; $h = 0{,}05$) in COMB.

- Modularnost grafov, pridobljenih z metodo Haircut($h = 0{,}1$; $h = 0{,}05$), je v povprečju statistično signifikantno višja pri naključno izbranih vrednostih. Razporeditev za metodo Haircut s parametrom 0,1 je prikazana v obliki histograma na sliki 6.2.

- Najvišja središčnost vozlišča grafov, pridobljenih z metodo Haircut ($h = 0{,}1$; $h = 0{,}05$), je v povprečju višja pri vzorcih sumljivih transakcij.

- Pri koeficientu mešanja imamo s kombinacijami samo vhodnih in izhodnih povezav na voljo 4 različne metrike. Kot statistično signifikantna med obema vzorcema se je pokazala le tista, ki v obeh primerih upošteva izhodne stopnje vozlišč. To smo opazili le pri metodi Haircut ($h = 0{,}1$, $h = 0{,}05$).

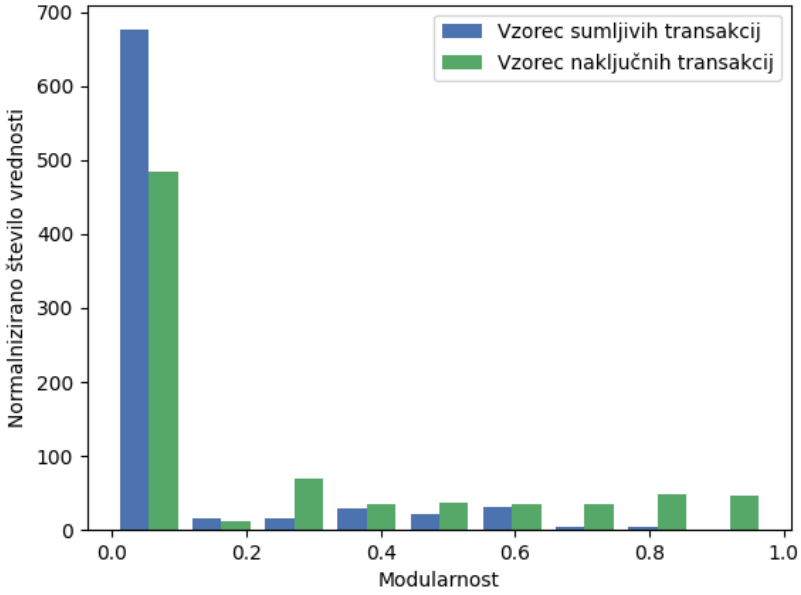
Zgornje rezultate smo grafično predstavili v spodnji tabeli. Z modro barvo so označena polja, kjer je vrednost metrike pri izbrani metodi statistično signifikantno večja pri naključno izbranem vzorcu, z rdečo barvo pa polja, kjer je vrednost metrike statistično signifikantno večja pri vzorcu sumljivih

transakcij. Polja, označena s sivo barvo, predstavljajo metrike, kjer grafi nastali s pripadajočo metodo v metriki, niso statistično bistveno drugačni.

Metrika - Metoda	Haircut$(0,1)$	Haircut$(0,05)$	TIHO	COMB
Število vozlišč				
Število povezav				
Gostota grafa				
Povprečna stopnja vozlišč				
Koeficient nakopičenosti				
Povprečna razdalja med vozlišči				
Premer				
Modularnost				
Najvišja središčnost vozlišča				
Koeficient mešanja (out, out)				

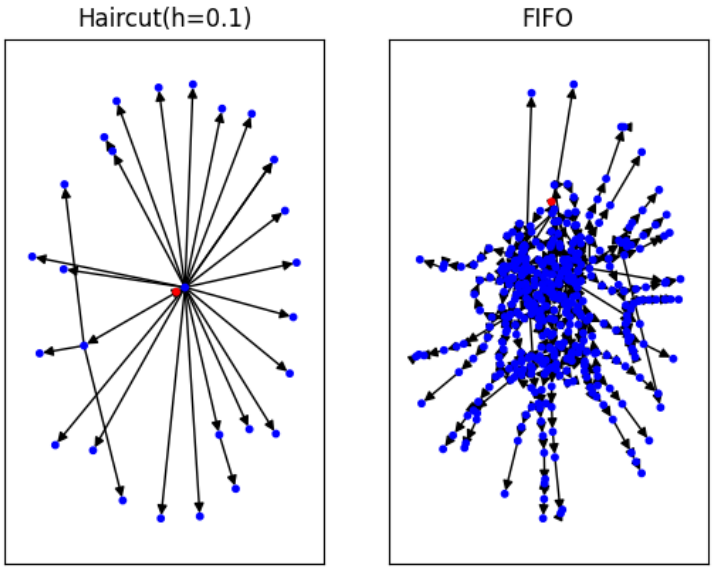
Iz zgornjega opisa in tabele lahko razberemo, da se povprečni rezultati večine metrik statistično signifikantno največkrat razlikujejo pri metodi Haircut. Pri tem vrednost parametra za testirani vrednosti ni imela bistvenega vpliva na rezultate. V primeru petih metrik so statistično signifikantne razlike povprečnih vrednosti vzorcev nastale pri metodi COMB. To lahko deloma pripišemo sorodnosti metod, saj se pri metodi COMB v kompleksnejših transakcijah, kjer bi izračun vseh kombinacij trajal predolgo, uporablja enaka hevristika, kot pri metodi Haircut. Grafi metode TIHO se med izračunanimi metrikami v povprečju razlikujejo v povprečni stopnji vozlišča in koeficientu nakopičenosti. Pri metodah LIFO in TIHO nismo našli statistično signifikantnih razlik pri nobeni od izbranih metrik.

Na podlagi tega lahko zaključimo, da je metoda Haircut s stališča te metrike bolj primerna od ostalih metod, saj se grafi transakcij, ki jih med preiskovanjem zgradi v povprečju, razlikujejo od grafov iz naključno izbranih transakcij. Metodi FIFO in LIFO sta se tu odrezali najslabše, saj se povprečja metrik, izračunanih nad grafi, v nobenem primeru niso bistveno razlikovala med vzorcema.



Slika 6.2: Histogram vrednosti modularnosti grafa transakcij za metodo Haircut s parametrom $h = 0{,}1$

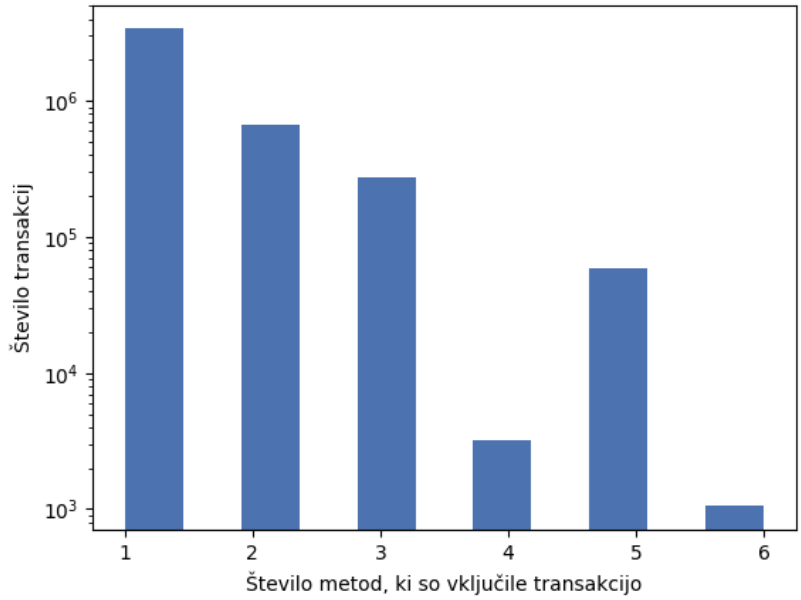
Kot lahko hitro vidimo iz slike 6.3, so grafi, ustvarjeni z različnimi metodami precej drugačni. Metoda Haircut (in v nekaterih primerih tudi COMB) se iz začetne transakcije širi med vse svoje izhode. S širjenjem in pridruževanjem drugih vhodov v transakcije pa te izgubljajo na deležu označenosti, kar je glavni razlog za praviloma krajše veje v grafu, ki ga tvori. Metode FIFO, LIFO in TIHO pa svoje vhode enostavno preslikajo v izhode, ki ob tem ne izgubijo na verjetnosti označenosti. Zaradi tega lahko tvorijo daljše verige, ki jih opazimo ne desni polovici slike 6.3.



Slika 6.3: Na sliki sta prikazana grafa, ki izhajata iz istega izhoda izbrane transakcije. Na levi je graf, ki je nastal z metodo Haircut, na desni pa graf iz metode FIFO.

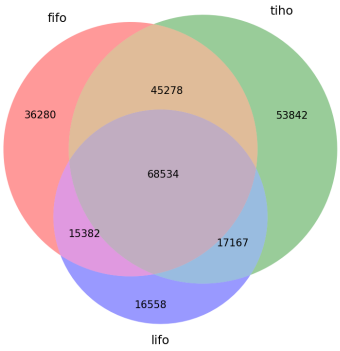
Ker so si grafi, pridobljeni z različnimi metodami med seboj precej različni, nas je zanimalo v koliko grafov je vključena posamezna transakcija, ki jo je označil katerikoli od implementiranih algoritmov. Transakcije, označene z večjim številom metod imajo višjo verjetnost, da so bolj povezane z izbranim izhodom transakcije.

Metode so na vseh testnih primerih skupaj ustvarile grafe s 4.403.664 različnimi vozlišči. Naredili smo analizo, pri kateri smo za vsako označeno vozlišče prešteli število metod, ki ga obiščejo med svojim preiskovanjem. Rezultati so prikazani v obliki histograma na sliki 6.4.



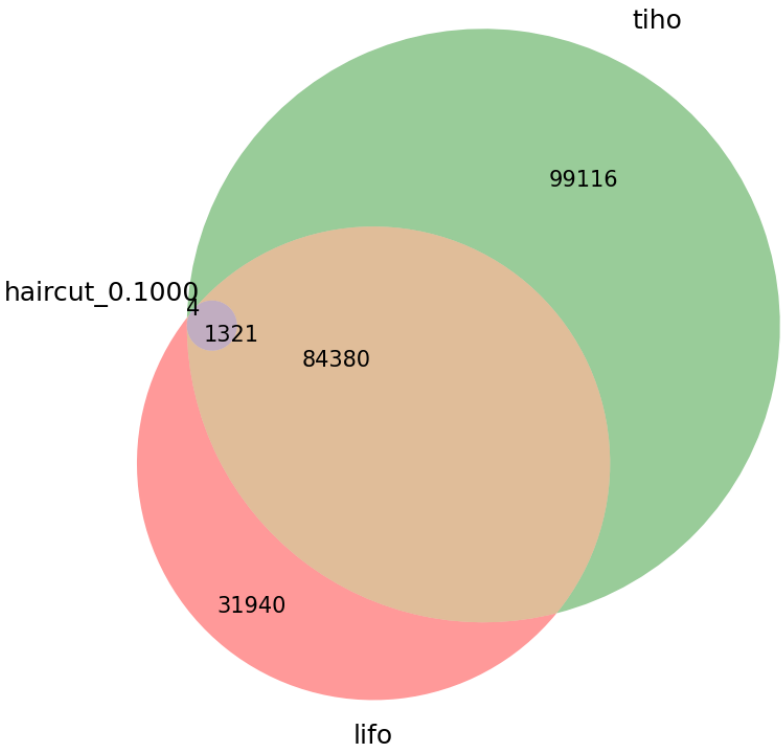
Slika 6.4: Histogram, ki prikazuje število transakcij glede na število metod, ki so jih označile.

Ker smo želeli poleg splošnih informacij o vseh grafih narediti tudi vizualizacijo na izbrani transakciji, smo za to izbrali Vennov diagram. Na sliki 6.5 smo prikazali število transakcij, ki je skupno grafom transakcij z začetkom v isti transakciji. Opazimo, da je dokaj velik delež transakcij skupen vsem trem metodam.



Slika 6.5: Vennov diagram, ki prikazuje množice transakcij, ki so jih označile metode FIFO, TIHO in LIHO iz prvega izhoda transkacije z identifikacijskim nizom: $BDF22D293B2627557EFF4791E12DA58EB19459BA267A542F3304DF7C147E8CD2$

Na sliki 6.6 so prikazani rezultati iste transakcije, podobno kot na sliki 6.5, vendar s to razliko, da smo namesto rezultatov metode FIFO prikazali rezultate metode Haircut($h = 0{,}1$). Iz slike je jasno razvidno, da metode LIFO, TIHO in FIFO označijo precej več transakcij in da sta tudi metodi LIFO in TIHO v tem primeru označili skoraj vse transakcije iz metode Haircut.



Slika 6.6: Vennov diagram, ki prikazuje množice transakcij, ki so jih označile metode Haircut($h = 0.1$), TIHO in LIFO iz prvega izhoda transkacije z identifikacijskim nizom: $BDF22D293B2627557EFF4791E12DA58EB19459BA267A542F3304DF7C147E8CD2$

Primerjava števila neporabljenih označenih kovancev in njihove porazdelitve

Metode smo med seboj primerjali tudi glede na število označenih izhodov (UTXO), ki niso bili porabljeni do zadnjega bloka, vključenega v našo podatkovno bazo. Neporabljeni izhodi so na nek način ključen rezultat metod za sledenje. Opravljenih transakcij za nazaj ne moremo razveljaviti, lahko pa se o tem obvesti prejemnike sumljivih kovancev. Zato je ključno, da vemo, kateri neporabljeni izhodi potencialno izvirajo iz kaznivih dejanj. V spodnji

tabeli so prikazani rezultati glede na število neporabljenih označenih izhodov od zaključku izvajanja metod. Pričakovano je najmanjše število izhodov pri metodi TIHO, saj je za metodo značilno, da vedno označi največje izhode in v njih združi sumljive vhode. Zaradi združevanja v največje izhode je teh na koncu seveda manj. Metodi FIFO in LIFO imata zelo podobno število neporabljenih označenih izhodov, kar potrjuje domnevo, da vrstni red izhodov znotraj transakcije nima bistvenega vpliva na število označenih neporabljenih izhodov transakcij. Pri metodi Haircut vidimo, da parameter h vpliva na število označenih izhodov.

Metoda	Število UTXO	Standardna deviacija
Haircut($h = 0{,}1$)	50.34	709.02
Haircut($h = 0{,}05$)	48.73	705.82
FIFO	33.81	330.89
LIFO	24.83	112.70
TIHO	16.47	333.80
COMB	26.98	488.953

Število neporabljenih označenih kovancev je pomembno, saj so ti edini, na katere lahko z opozarjanjem in označevanjem vplivamo. Preostal del grafa transakcij predstavlja le zgodovino in pot, po kateri smo prišli do teh neporabljenih izhodov. Bistveno pomembnejša od njihovega števila, pa je točnost napovedi, s katero določena metoda razkrije, kateri neporabljeni izhodi so povezani z vhodno sumljivo transakcijo. To pa je za zdaj samo z javno dostopnimi podatki izjemno težko preverjati.

Primerjava časov izvajanja metod

Zaradi velikega števila transakcij, ki bi jim potencialno želeli slediti, je v praksi pomemben tudi čas izvajanja metod. Zanimalo nas je, ali obstajajo bistvene razlike med časi, ki so potrebni za izvajanje implementiranih metod na izbranem vzorcu.

V spodnji tabeli so zapisani povprečni časi, standardne deviacije in povprečno število transakcij, ki jih posamezna metoda obišče iz izhodišč, podanih v vzorcu.

Metoda	Čas[s]	Standardna deviacija	Število povezav
Haircut($h = 0.1$)	0.12	0.68	63
Haircut($h = 0.05$)	0.21	1.33	77
FIFO	3.76	15.96	3265
LIFO	3.42	15.02	2943
TIHO	1.60	3.34	2252
COMB	0.31	0.76	6

Iz zgornje tabele je razvidno, da v povprečju največ časa porabita metodi FIFO in LIFO, kar lahko pojasnimo s tem, da obe v transakciji enostavno označita prve oz. zadnje izhode. Na ta način se označbe prenašajo skozi graf transakcij in tudi ob manjših vrednostih ne izginejo, kot na primer pri metodi Haircut. Metoda TIHO za razliko od njiju koncentrira označbe v izhodih z najvišjo vrednostjo, in ko pridemo do neporabljenih izhodov, se preiskovanje za veliko število kovancev naenkrat zaustavi. Metoda COMB je sicer računsko zelo zahtevna, a smo jo učinkovito omejili tako, da pri kompleksnejših transakcijah uporabljamo pravilo iz metode Haircut. Poleg tega ne nadaljujemo sledenja pri izhodih, ki prenašajo majhno število kovancev, kar je najbrž eden ključnih razlogov, da metoda obišče daleč najmanj transakcij. Časi izvajanja so močno odvisni od števila transakcij, ki jih metode obiščejo. Kljub nizkim vrednostim pa lahko pri metodi COMB opazimo, da je čas izvajanja glede na število obiskanih transakcij bistveno večji kot pri ostalih metodah.

Čeprav so povprečni časi zaradi enostavnih hevristik (razen v metodi COMB) in uporabe primernih tehnologij razmeroma nizki, pa posamezne metode za določene vhode vseeno potrebujejo precej časa, kar opazimo predvsem ob sledenju večjemu številu transakcij.

Primerjave metod glede na interpretabilnost

Metode se poleg omenjenih statistično signifikantnih razlik glede na izbrane podatke razlikujejo tudi glede interpretabilnosti dobljenih rezultatov.

Najlažje interpretiramo rezultate metode Poison, saj označimo vse izhode transakcij, ki imajo označen vsaj en vhod. Tako se označeni kovanci širijo iz vseh izhodov vseh transakcij, ki jih obdelamo s to metodo. Njihovo število se tako eksponentno širi po grafu transakcij.

Podobno enostavna je interpretacija pri metodi Haircut, s pomembno razliko, da upoštevamo delež označenih kovancev v transakciji in posledično nehamo slediti kovancem, ko ta pade pod izbrano mejo.

Metodi FIFO in LIFO sta s stališča interpretabilnosti enako zahtevni, saj pri prvi začnemo označevati izhode od prvega proti zadnjem, pri slednji pa ravno obratno. Metodi zaradi svojih lastnosti ohranjata konstantno količino označenih neporabljenih kovancev skozi celoten potek izvajanja metode.

Metoda TIHO je s stališča interpretabilnosti podobna metodama FIFO in LIHO. Na vrstni red označevanja izhodov pa namesto vrstnega reda vplivajo vrednosti izhodov. Tako kot metodi FIFO in LIFO tudi metoda TIHO ohranja konstantno vrednost neporabljenih označenih izhodov.

Interpretabilnost je med predstavljenimi metodami daleč najslabša pri metodi COMB. Pri transakcijah z manjšim številom vhodov in izhodov metoda izračuna vse mogoče kombinacije, med katerimi bi glede na vrednosti lahko prišlo do preslikav med vhodi in izhodi. Po izračunu kombinacij pa za preslikave izračuna še verjetnosti, ki jih nato uporabi pri porazdelitvi označb izhodom transakcije. Kljub temu da metoda v nekaterih primerih lahko pametno prepozna, kateri izhodi pripadajo, katerim (označenim) vhodom pa je s stališča interpretabilnosti daleč najslabša med obravnavanimi metodami.

Uporaba dodatnih podatkov v metodah

Uporaba dodatnih podatkov v metodah nam koristi, da lažje in bolj natančno dosežemo željeni cilj sledenja transakcijam skozi omrežje. V našem primeru

podatki o transakcijah kripto menjalnic omogočajo, da končamo sledenje, ko dosežemo transakcijo povezano s kripto menjalnico. Zaradi zahtev po identifikaciji uporabnikov (KYC) lahko od menjalnice pričakujemo, da pozna identiteto uporabnika, ki je opravil omenjeno transakcijo. Na ta način lahko zaključimo vejo preiskovanja, ki je dosegla menjalnico.

Za potrebe testiranja smo pripravili zbirko podatkov, opisano v podpoglavju 5.3.2, in jo preizkusili z istimi začetnimi transakcijami, kot pri prejšnjih meritvah. Dobljene rezultate smo primerjali glede na število mest, pri katerih smo med vsemi transakcijami prišli do znane transakcije kripto menjalnice. Podatki so razvidni v spodnji tabeli.

Metoda	Število izhodov, ki dosežejo kripto menjalnice
Haircut($h = 0{,}1$)	0
Haircut($h = 0{,}05$)	0
FIFO	10015
LIFO	9487
TIHO	4412
COMB	38

Z metodo Haircut nam nikoli ni uspelo priti do transakcije kripto menjalnice. Veliko bolj pogosto se je to zgodilo z metodama FIFO in LIFO. Metoda TIHO je ravno tako dokaj pogosto dosegla znane transakcije, a vseeno precej manj pogosto, saj metoda označene vhode združi in preslika v največje izhode. Na tem mestu je treba poudariti, da omenjene tri metode v primerjavi z ostalimi označijo precej več transakcj in posledično večkrat dosežejo znane transakcije iz kripto menjalnic. Metoda COMB je v primerjavi s prej omenjenimi tremi metodami - če gledamo absolutne vrednosti - znane transakcije dosegla dokaj redko. Glede na nizko število transakcij, ki jih metoda v povprečju obišče, pa so rezultati zelo spodbudni.

Verjamemo, da ima uporaba dodatnih podatkov v metodah za sledenje transakcijam veliko uporabno vrednost, ki pa se v rezultatih pokaže šele, ko imamo na voljo dovolj veliko podatkovno bazo vseh transakcij različnih

menjalnic. Za praktično uporabo je pomembno, da taka podatkovna baza ne potrebuje osebnih podatkov o osebah, povezanih z določeno transakcijo, vendar le označbo, da poznamo osebe, odgovorne za to transakcijo.

Poglavje 7

Zaključek

V magistrski nalogi smo želeli narediti primerjavo med metodami za sledenje označenim kovancem skozi graf Bitcoin transakcij. Tako sledenje je zaradi psevdonimnosti uporabnikov in velikega števila transakcij zahtevna naloga. Za namene testiranja in primerjave smo implementirali metode iz članka [35] in predlagali novo metodo COMB. Pripravili smo vzorec transakcij, ki glede na spletno stran[1] spadajo med sumljive transakcije in kontrolni vzorec, kjer smo izbrali naključne transakcije iz istega obdobja. Vse implementirane metode smo pognali na obeh vzorcih, da bi pridobili podatke o tem kakšne rezultate nam dajejo posamezne metode.

V poglavju 6 smo opisali primerjavo rezultatov med obema vzorcema in med samimi metodami. Na pridobljenih grafih smo izračunali metrike, opisane v podpoglavju 4.2 in ugotovili, da se med vzorcema v največ metrikah razlikujejo grafi, ki so nastali z metodo Haircut.

Poleg oblike grafov smo pogledali tudi preseke množic transakcij, narejenih z različnimi grafi. Daleč največ transakcij je bilo označenih z manj kot tremi metodami. Pri praktični uporabi bi bilo pametno transakcijam, označenim s strani več različnih metod, pripisati večjo verjetnost za povezavo z izvirno transakcijo.

Ker so neporabljeni izhodi transakcij edini, na katere je udeležence smi-

[1] www.bitcoinabuse.com

selno opozoriti (v kolikor smo s pomočjo metod ugotovili povezavo s sumljivo transakcijo), smo spremljali njihovo število med metodami.

Zaradi izjemno velikega preiskovalnega prostora (vse transakcije v omrežju Bitcoin) in potencialno velikega števila transakcij, ki bi jim želeli slediti, smo se pri načrtovanju in implementaciji osredotočili na hitrost. V poglavju 6 smo tako med seboj primerjali povprečne čase izvajanja med posameznimi metodami. Ugotovili smo, da je čas preiskovanja močno odvisen od števila vozlišč, ki jih metoda obišče.

Vsem metodam smo dodali tudi možnost, da uporabijo podatke, ki niso del verige podatkov in se tako bolje odločajo. V tem primeru so bile to transakcije s strani kripto menjalnic, ki so obvezane k temu, da preverjajo identitete svojih uporabnikov. Na izbranem vzorcu smo izvedli preiskovanje z vsemi metodami in primerjali število transakcij iz naše podatkovne baze, na katere so transakcije naletele.

Nadaljnje delo

Implementirane metode predstavljajo dobro osnovo za sledenje označenim transakcijam skozi graf Bitcoin transakcij. Ocenjujemo, da v primeru bolj enostavnih transakcij z majhnim številom vhodov in izhodov delujejo dokaj učinkovito in dajo pričakovane rezultate. V kompleksnejših grafih transakcij, kjer imamo več vhodov in izhodov, imajo predstavljene metode premalo podatkov za optimalno odločanje. V takih primerih so izjemnega pomena drugi podatki, ki jih metode lahko izkoriščajo.

Zanimivi se nam zdijo predvsem podatki o:

- transakcijah in naslovih znanih entitet, ki preverjajo svoje uporabnike (in sledijo smernicam KYC).
  V tem primeru lahko preiskovanje v določeni veji zaključimo, saj glede transakcije obstajajo dodatne informacije pri znani entiteti.

- Podatki o tem, katere transakcije in naslovi pripadajo isti entiteti.
  S podatki naslovih, ki so pod nadzorom iste entitete lahko transakcije

poenostavimo, saj vhode in izhode, ki pripadajo isti entiteti med seboj združimo in tako zmanjšamo njihovo število. Poleg tega pa bi jih lahko v določenih primerih, ko bi nekateri vhodi in izhodi pripadali isti entiteti, z veliko verjetnostjo med seboj povezali. Razvrščanje naslovov v skupine je problem poznan kot razvrščanje naslovov v gruče (angl. address clustering) in je v svetu Bitcoina dokaj dobro raziskan [32, 31].

Za nadaljnji razvoj bo ključna uporaba dodatnih podatkov, ki bodo metodam omogočali bolj natančno sledenje označenim kovancem skozi omrežje. Iz pridobljenih rezultatov ni mogoče določiti absolutno najboljše metode za sledenje, saj ima vsaka svoje prednosti in slabosti. Z njihovim vzporednim izvajanjem in združevanjem rezultatov bomo lahko v prihodnosti kovancem skozi omrežje sledili bolj natančno. Implementirane metode omogočajo tudi vzporedno sledenje več označenim kovancem hkrati, kar predstavlja dodaten potencial za izboljšavo sledenja, saj sumljive transakcije med seboj niso povsem izolirane in se njihovi vhodi in izhodi v nekaterih primerih med seboj prepletajo. Vzporedno izvajanje terja svoj davek, saj za to potrebujemo precej več delovnega pomnilnika in procesorske moči.

Literatura

[1] Nakamoto, S. & Others Bitcoin. *A Peer-to-peer Electronic Cash System.* (2008)

[2] Coinmarketcap, `https://coinmarketcap.com/`, [Dostopano 20.11.2021]

[3] Rogaway, P. & Shrimpton, T. Cryptographic Hash-Function Basics: Definitions, Implications, and Separations for Preimage Resistance, Second-Preimage Resistance, and Collision Resistance. (2004)

[4] Bakhtiari, S., Safavi-Naini, R., Pieprzyk, J. & Others Cryptographic hash functions: A survey. (Citeseer,1995)

[5] Dang, Q. & Others Secure hash standard. (2015)

[6] Mendel, F., Pramstaller, N., Rechberger, C. & Rijmen, V. On the collision resistance of RIPEMD-160. *International Conference On Information Security.* pp. 101-116 (2006)

[7] R. Maitra, What The Heck is Schnorr, `https://medium.com/bitbees/what-the-heck-is-schnorr-52ef5dba289f` [Dostopano: 20.11.2021]

[8] BRECELJ, B. Podpisi brez možnosti zanikanja. (2020), https://repozitorij.uni-lj.si/IzpisGradiva.php?lang=slv&id=120373

[9] Bitcoin Wiki - Secp256k1, `https://en.bitcoin.it/wiki/Secp256k1`, [Dostopano 20.11.2021]

[10] Wikipedia, Forth (programming language), https://en.wikipedia.org/wiki/Forth_(programming_language), [Dostopano: 20.11.2021]

[11] Coursera - Bitcoin Scripts, https://www.coursera.org/lecture/cryptocurrency/bitcoin-scripts-HWjti, [Dostopano: 20.11.2021]

[12] IETF - The Base58 Encoding Scheme, https://tools.ietf.org/id/draft-msporny-base58-01.html, [Dostopano: 20.11.2021]

[13] Calvo Pardo, H., Mancini, T. & Olmo, J. Machine Learning the Carbon Footprint of Bitcoin Mining. (CEPR Discussion Paper No. DP16267,2021)

[14] CNBC, Bitcoin's wild ride renews worries about its massive carbon footprint, https://www.cnbc.com/2021/02/05/bitcoin-btc-surge-renews-worries-about-its-massive-carbon-footprint.html, [Dostopano: 20.11.2021]

[15] Bashir, I. Mastering blockchain. (Packt Publishing Ltd,2017)

[16] Lamport, L., Shostak, R. & Pease, M. The Byzantine generals problem. *Concurrency: The Works Of Leslie Lamport*. pp. 203-226 (2019)

[17] Gencer, A., Basu, S., Eyal, I., Van Renesse, R. & Sirer, E. Decentralization in bitcoin and ethereum networks. *International Conference On Financial Cryptography And Data Security*. pp. 439-457 (2018)

[18] Kent P., Bain T. Cryptocurrency Mining For Dummies (2019)

[19] Lin, Q., Li, C., Zhao, X. & Chen, X. Measuring decentralization in bitcoin and ethereum using multiple metrics and granularities. *2021 IEEE 37th International Conference On Data Engineering Workshops (ICDEW)*. pp. 80-87 (2021)

[20] Antonopoulos, A. Mastering Bitcoin: Unlocking Digital Crypto-Currencies. (O'Reilly Media, Inc.,2014)

[21] Zhou, Q., Huang, H., Zheng, Z. & Bian, J. Solutions to scalability of blockchain: A survey. *IEEE Access*. **8** pp. 16440-16455 (2020)

[22] Blockchair - Privacy-o-meter, `https://blockchair.com/api/docs#link_M6` [Dostopano: 20.11.2021]

[23] Bitcoin wiki - Privacy, `https://en.bitcoin.it/wiki/Privacy`, [Dostopano: 20.11.2021]

[24] Maurer, F. A survey on approaches to anonymity in Bitcoin and other cryptocurrencies. *Informatik 2016*. (2016)

[25] Bryans, D. Bitcoin and money laundering: mining for an effective solution. *Ind. LJ*. **89** pp. 441 (2014)

[26] Christin, N. Traveling the Silk Road: A measurement analysis of a large anonymous online marketplace. *Proceedings Of The 22nd International Conference On World Wide Web*. pp. 213-224 (2013)

[27] Paquet-Clouston, M., Haslhofer, B. & Dupont, B. Ransomware payments in the bitcoin ecosystem. *Journal Of Cybersecurity*. **5**, tyz003 (2019)

[28] Foley, S., Karlsen, J. & Putniņš, T. Sex, drugs, and bitcoin: How much illegal activity is financed through cryptocurrencies?. *The Review Of Financial Studies*. **32**, 1798-1853 (2019)

[29] Crypto Crime Summarized: Scams and Darknet Markets Dominated 2020 by Revenue, But Ransomware Is the Bigger Story, `https://blog.chainalysis.com/reports/2021-crypto-crime-report-intro-ransomware-scams-darknet-markets`, [Dostopano: 20.11.2021]

[30] Sharma, A. & Bhatia, A. Bitcoin's Blockchain Data Analytics: A Graph Theoretic Perspective. *ArXiv Preprint ArXiv:2002.06403*. (2020)

[31] Harrigan, M. & Fretter, C. The unreasonable effectiveness of address clustering. *2016 Intl IEEE Conferences On Ubiquitous Intelligence & Computing, Advanced And Trusted Computing, Scalable Computing And Communications, Cloud And Big Data Computing, Internet Of People, And Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld)*. pp. 368-373 (2016)

[32] Ermilov, D., Panov, M. & Yanovich, Y. Automatic bitcoin address clustering. *2017 16th IEEE International Conference On Machine Learning And Applications (ICMLA)*. pp. 461-466 (2017)

[33] De Meo, P., Ferrara, E., Fiumara, G. & Provetti, A. Generalized louvain method for community detection in large networks. *2011 11th International Conference On Intelligent Systems Design And Applications*. pp. 88-93 (2011)

[34] Ferguson, N. & Schneier, B. Practical cryptography. (Wiley New York,2003)

[35] Tironsakkul, T., Maarek, M., Eross, A. & Just, M. Probing the mystery of cryptocurrency theft: An investigation into methods for cryptocurrency tainting analysis. *ArXiv Preprint ArXiv:1906.05754*. (2019)

[36] Kočnik, U. Preprečevanje pranja denarja s kriptovalutami. (2019), https://repozitorij.uni-lj.si/IzpisGradiva.php?lang=slv&id=110217

[37] Anderson, R. Making Bitcoin Legal (Transcript of Discussion). *Cambridge International Workshop On Security Protocols*. pp. 254-265 (2018)

[38] The Blockchain Trilemma: Decentralized, Scalable, and Secure?, `https://medium.com/certik/the-blockchain-trilemma-decentralized-scalable-and-secure-e9d8c41a87b3`, [Dostopano: 20.11.2021]

[39] Learn me a Bitcoin: P2PKH, `https://learnmeabitcoin.com/technical/p2pkh`, [Dostopano: 20.11.2021]

[40] Analyst: Over 98% Of Bitcoin Mining Hardware Will Become Obsolete After BTC Halving,`https://bitcoinexchangeguide.com/analyst-over-98-of-bitcoin-mining-hardware-will-become-obsolete-after-btc-halving/`, [Dostopano: 20.11.2021]

[41] Mapping the Major Bitcoin Forks, `https://www.visualcapitalist.com/major-bitcoin-forks-subway-map/`, [Dostopano: 20.11.2021]

[42] Welch, B. The generalization of 'STUDENT'S'problem when several different population varlances are involved. *Biometrika*. **34**, 28-35 (1947)

[43] Lin, Y., Wu, P., Hsu, C., Tu, I. & Liao, S. An evaluation of bitcoin address classification based on transaction history summarization. *2019 IEEE International Conference On Blockchain And Cryptocurrency (ICBC)*. pp. 302-310 (2019)

[44] Fleder, M., Kester, M. & Pillai, S. Bitcoin transaction graph analysis. *ArXiv Preprint ArXiv:1502.01657*. (2015)

[45] Lv, X., Zhong, Y. & Tan, Q. A Study of Bitcoin De-Anonymization: Graph and Multidimensional Data Analysis. *2020 IEEE Fifth International Conference On Data Science In Cyberspace (DSC)*. pp. 339-345 (2020)

[46] Oggier, F., Phetsouvanh, S. & Datta, A. BiVA: Bitcoin Network Visualization Analysis. *2018 IEEE International Conference On Data Mining Workshops (ICDMW)*. pp. 1469-1474 (2018)

[47] Spagnuolo, M., Maggi, F. & Zanero, S. Bitiodine: Extracting intelligence from the bitcoin network. *International Conference On Financial Cryptography And Data Security*. pp. 457-468 (2014)

[48] Kalodner, H., Möser, M., Lee, K., Goldfeder, S., Plattner, M., Chator, A. & Narayanan, A. Blocksci: Design and applications of a blockchain analysis platform. *29th USENIX Security Symposium (USENIX Security 20)*. pp. 2721-2738 (2020)

[49] Tironsakkul, T., Maarek, M., Eross, A. & Just, M. Tracking Mixed Bitcoins. *Available At SSRN*. (2020)

[50] Hong, Y., Kwon, H., Lee, J. & Hur, J. A practical de-mixing algorithm for bitcoin mixing services. *Proceedings Of The 2nd ACM Workshop On Blockchains, Cryptocurrencies, And Contracts*. pp. 15-20 (2018)

[51] Hu, Y., Seneviratne, S., Thilakarathna, K., Fukuda, K. & Seneviratne, A. Characterizing and Detecting Money Laundering Activities on the Bitcoin Network. *ArXiv Preprint ArXiv:1912.12060*. (2019)

[52] Ziegeldorf, J., Matzutt, R., Henze, M., Grossmann, F. & Wehrle, K. Secure and anonymous decentralized Bitcoin mixing. *Future Generation Computer Systems*. **80** pp. 448-466 (2018)