Pregled dokazov brez razkritja znanja

Kim Ana Badovinac
Fakulteta za računalništvo in informatiko
Večna pot 113
Ljubljana, Slovenija
kb6332@student.uni-lj.si

Nino Brezac
Fakulteta za računalništvo in informatiko
Večna pot 113
Ljubljana, Slovenija
nb9762@student.uni-lj.si

Martin Ekar Fidermuc
Fakulteta za računalništvo in informatiko
Večna pot 113
Ljubljana, Slovenija
me4470@student.uni-lj.si

Martin Prajnc
Fakulteta za računalništvo in informatiko
Večna pot 113
Ljubljana, Slovenija
mp1238@student.uni-lj.si

POVZETEK

Dokaz brez razkritja znanja (angl. Zero knowledge proof) je zelo uporaben matematičen problem, kjer udeleženec komunikacije pozna skrivnost in to lahko dokaže, ne da bi jo pri tem razkril. Zaradi izjemne elegantnosti in učinkovitosti se prej omenjeni koncept vedno več uporablja v računalništvu. V delu predstavimo koncept dokaza brez razkritja znanja in podrobna aplikacija le tega v igri Numberlink. Kasneje razširimo pregled področja še na nekatere abstraktne primere in na kriptografske protokole. Za konec predstavimo aplikacijo dokaza brez razkritja znanja na področju Blockchaina.

Ključne besede

algoritmi, dokaz brez razkritja znanja, numberlink, kriptografija, veriga blokov

1. UVOD

Numberlink je miselna uganka, pri kateri moramo povezati pare števil, ki so razpršeni v tabeli. Tabela je velikosti m x n in vsebuje k parov števil. Cilj je povezati vse pare števil, pri čemer se poti ne smejo sekati. Vsako polje lahko vsebuje natanko eno pot. Ni potrebno, da pokrijemo vsa prazna polja. Uganka je popolna, če ima natanko eno rešitev in poti pokrijejo vsa prazna polja.

Problem, ki ga preučevani članek [10] reši je sledeč: predpostavimo, da je Ana strokovnjakinja za igro Numberlink-a in sama sestavi uganko (ki je rešljiva) ter jo da Borutu, da jo ta reši. Borut nima dovolj znanja in izkušenj, da bi rešil uganko, zato trdi, da uganka ni rešljiva. Kako lahko Ana prepriča Boruta, da je uganka rešljiva, ne da bi pri tem razkrila rešitev?

Rešitev problema se skriva v matematičnem postopku, imenovanem dokaz brez razkritja znanja. Cilj članka je zatorej, čim bolj natančno predstaviti originalni problem, nato pa se poglobiti v konkretno tehniko dokaza brez razkritja znanja in izpostaviti nekatera od področij ter situacij, v katerih se uporablja.

2. DOKAZ BREZ RAZKRITJA ZNANJA

Dokaz brez rakritja znanja je interaktiven dokaz med tistim, ki skuša nekaj dokazati (P), in tistim, ki preveri, če je dokaz pravilen (V), pri tem pa je obema dan izračunljiv problem x. Samo P pozna w, ki je rešitev x, predpostavljamo pa, da je izračunljivostna moč V-ja omejena, zato ne more pridobiti w iz x. P želi prepričati V, da pozna w, ne da bi pri tem razkril informacije o njem. Dokaz brez razkritja znanja mora zadostiti naslednjim pogojem.

- **Polnost:** če P pozna w, potem lahko prepriča V z zelo veliko verjetnostjo.
- **Uglašenost:** če P ne pozna w, potem P ne more prepričati V, razen z zelo malo verjetnostjo, ki ji pravilo napaka uglašenosti.
- **Brez razkritja znanja:** V ne more pridobiti nobenih informacij o w, torej obstaja probabilistični algoritem S v polinomskem času, ki ne pozna w, in imajo izhodi algoritma S enako verjetnostne porazdelitve kot izhodi pravega protokola.

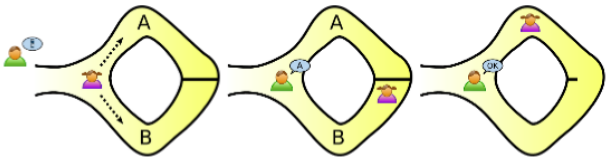
Koncept dokaza brez razkritja znanja so prvič predstavili Shafi Goldwasser, Silvio Micali in Charles Rackoff [4], Goldreich [3] pa je dokazal, da obstaja dokaz brez razkritja znanja za vsak NP problem.

2.1 Abstraktni primeri

2.1.1 Jama Ali Babe

Jama Ali Babe je dokaj preprost primer, s katerim bralcu predstavimo koncept dokazov brez razkritja znanja [9]. Imamo osebi Ano in Bora ter jamo oblike kroga, ki je sestavljena iz A in B poti, jama je prikazana na sliki 1. V notranjosti jame

je stena oz. prehod, ki se odpre le, če oseba v jami izreče skrivno besedo. Predpostavljamo da Ana pozna to skrivno besedo in to želi dokazati Boru, brez da mu jo razkrije. Postopek preverjanja poteka na naslednji način: Ana gre prva v jamo po poti A ali B (Bor ne ve, katero pot je izbrala), za njo gre v jamo Bor ter se ustavi na razpotju, kjer se pot loči v poti A in B. Tam zakliče "pot A" ali "pot B" in potem počaka, da se Ana vrne, postopek se ponovi n-krat. Če torej Ana res pozna skrivno besedo, potem lahko vedno odpre prehod in se vrne po poti, ki jo zakliče Bor. Če pa Ana ne pozna besede, potem pa je $\frac{1}{2}$ verjetnosti, da se Ana vrne po pravi poti. Ker se postopek ponovi velikokrat, je tveganje, da Ana ne bi bila iskrena, zelo majhno.



Slika 1: Jama Ali Babe z magičnim prehodom [8].

2.1.2 Dve žogi in barvno slep prijatelj

V tem problemu predpostavljamo, da imamo prijatelja, ki je barvno slep za rdečo in zeleno barvo, in dve žogi, ena je rdeče, druga pa zelena barve. Za prijatelja sta žogi enaki in ni prepričan da se žogi sploh kako razlikujeta. Zato mu želimo dokazati, da sta žogi v resnici drugačni, pri tem pa mu ne želimo izdati, katera žoga je rdeča in katera zelena barve. Dokaz poteka po sledečem postopku: najprej damo prijatelju obe žogi, ta ju skrije za hrbet in nam pokaže eno izmed njih, nato jo skrije nazaj za hrbet in potem razkrije le eno žogico in vpraša "Ali sem zamenjal žogico?". Celoten postopek se ponovi večkrat. Glede na barvo žoge, ki jo prijatelj pokaže, lahko takoj ugotovimo, ali jo je zamenjal, ali ne. Če bi bili žogi popolnoma enaki, potem bi ugibali z verjetnostjo 50%. Glede na to, da je verjetnost, da bi izbrali pravilno odločitev, vedno 50%, je verjetnost, da bi pravilno izbrali odločitev za vse ponovitve postopka zelo majhna. Če bi se postopek ponovil recimo 20 krat, potem naj bi prijatelj postal prepričan, da sta žogi različnih barv.

3. IGRA NUMBERLINK

V izhodiščnem članku [10] je predstavljen protokol za igro Numberlink, ki prevede problem na grafu v problem s štetjem elementov.

3.1 Šahovnica

Na šahovnici velikosti m x n je razporejenih k parov števil $(1, 2, ...k)$. Oštevilčena polja so imenovana končna, neoštevilčena pa prehodna. Veljavna pot med dvema končnima poljema je zaporedje celic (c_1,c_2,...c_t), pri čemer sta c_1 in c_t končni polji z enako številko, ostala polja pa so prehodna. Pri tem mora za prehodna polja veljati, da je c_i sosed c_{i+1} za vsak $i = 1, 2, ..., t-1$. Pot je preprosta, če velja, da sta polji c_i in c_j sosednji in ne velja izraz $j > i+1$.

3.2 Karte

Protokol uporablja dva tipa kart. Kodirne karte so križevi (♣) in srčevi (♡) asi, označevalne karte pa so oštevilčene.

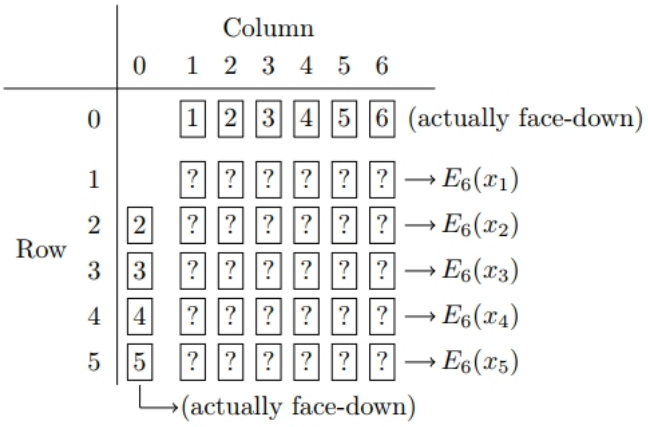
3	3	3	4	4
3	1	3	4	3
2	1	3	4	3
2	1	3	3	3
2	1	1	1	1

Slika 2: Primer rešene popolne Numberlink uganke.

Števila zakodiramo kot $E_y(x)$. Število y predstavlja, koliko križevih kart imamo, pri tem pa je karta na poziciji x srčev as in predstavlja naše zakodirano število. Na primer, $E_4(2)$ bi zapisali tako: ♣♡♣♣

3.3 Tabela kart

Iz kupčkov kodirnih kart, ki predstavljajo števila, zgradimo tabelo velikost a x b. Nad tabelo dodamo vrstico označevalnih kart, ki naraščajo od leve proti desni $(1, 2, ..., b)$. Ob levi strani tabele dodamo stolpec označevalnih kart, ki naraščajo od zgoraj navzdol. Pri tem izpustimo prvo vrstico. Tako imamo karte z vrednostmi $2, 3, ..., a$. Novo dodani stolpec in vrstica imata indeks 0.



Slika 3: Tabela kart pri b = 6.

3.4 Mešanje in preurejanje tabele

Tabeli premešamo stolpce in vrstice z uporabo postopka *double-scramble shuffle*. Postopek za preurejanje je sledeč:

1. Naključno izberemo permutacijo $p = (p_2, p_3, ..., p_a)$ za števila $(2, 3, ..., a)$. Preverjevalec ne sme poznati vrednosti p.
2. Na skrivaj premešamo vrstice $2, 3, ..., a$ glede na permutacijo p.
3. Naključno izberemo permutacijo $q = (q_1, q_2, ..., q_b)$ za števila $(1, 2, ..., b)$. Preverjevalec ne sme poznati vrednosti q.
4. Na skrivaj premešamo stolpce $1, 2, ..., b$ glede na permutacijo q.

V praksi to lahko naredimo tako, da dokazovalec vidno pospravi vsako vrstico kart v lastno kuverto ter kuverte nato na skrivaj premeša. Isto naredi še za stolpce.

Karte je potrebno ob dokazovanju tudi vrniti na njihove prvotne položaje. V ta namen se uporabi protokol za preurejanje. Ta ima naslednje korake:

1. Premešamo tabelo z uporabo *double-scramble shuffle*.
2. Vidno obrnemo vse karte v stolpcu 0. Te karte imajo vrednosti $(p_2, p_3, ..., p_a)$.
3. Vidno preuredimo vrstice $2, 3, ..., a$ glede na permutacijo $p = (p_2, p_3, ..., p_a)$.
4. Vidno obrnemo vse karte v vrstici 0. Te karte imajo vrednosti $(q_1, q_2, ..., q_b)$.
5. Vidno preuredimo stolpce $1, 2, ..., b$ glede na permutacijo $q = (q_1, q_2, ..., q_b)$.

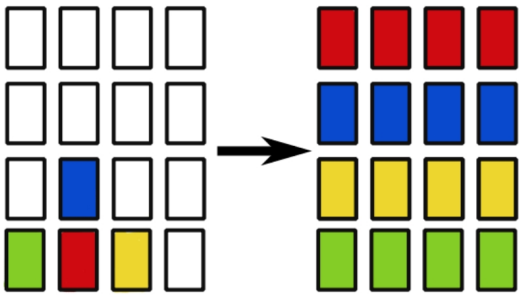
Ker smo na začetku postopka tabelo premešali z *double-scramble shuffle*, so vsi stolpci in vrstice razporejeni tako, da zadostijo vsem možnim permutacijam. Obračanje kart tako ne bo razkrilo nobenega podatka.

3.5 Postopek dokaza

Najprej predpostavimo, da je dana uganka popolna, torej ima samo preproste poti in so vsa polja pokrita. Dokazovalec $P(prover)$ bo preverjevalcu $V(verifier)$ poskušal dokazati dve lastnosti:

1. Vsaka končna celica ima natanko eno sosednje polje z enako številko.
2. Vsaka prehodna celica ima natanko dve sosednji polji z enako številko.

P najprej vidno postavi kupčke kart $E_k(x)$ na vsako končno polje. Nato na skrivaj postavi kupčke kart na vsa prehodna polja s številko x.



Slika 4: Dokazovanje pravilnosti za poljubno končno polje. Karte kupčka s končnega polja so obarvane z rdečo.

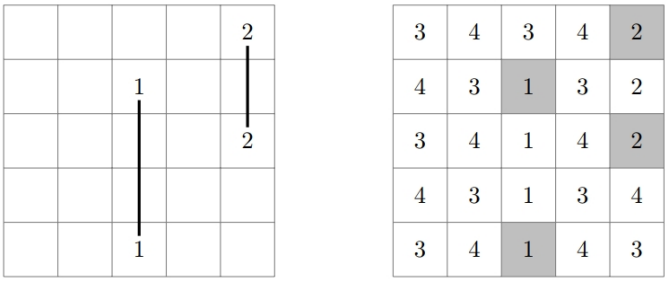
P sedaj lahko prične z dokazovanjem rešitve za končna polja c. Postopek je sledeč:

1. Vzame kupček kart, ki predstavlja končno polje, ter vse sosednje kupčke. Iz kupčkov sestavi novo tabelo kart (Slika 4), pri kateri je kupček končnega polja v prvi vrstici, ostali pa so lahko poljubno razporejeni. Na koncu doda še vrstico in stolpec z označevalnimi kartami.
2. Naredi *double-scramble shuffle*.
3. Vidno obrne vse karte v vrstici 1 in poišče srčevega asa. Recimo, da je ta v stolpcu j.
4. Vidno obrne vse karte v stolpcu j. V stolpcu mora biti poleg asa iz prve vrstice še natanko en srčev as, sicer V zavrne rešitev in postopek se zaključi.
5. Izvede protokol za preurejanje ter nato vidno vrne kupčke kart v prvotno tabelo.

Postopek za preverjanje prehodnih polj je enak kot za končna polja, le da morajo biti pri 4. koraku v stolpcu natanko trije srčevi asi. P izvede postopek za vsa polja tabele. Če je P za vsako polje uspešno dokazal pravilnost rešitve, je dokaz uspel.

3.6 Nepopolne uganke

Pri nepopolnih ugankah moramo najprej preveriti, ali so poti v rešitvi preproste, sicer jih popravimo. Nato vsa prazna polja oštevilčimo. Praznemu polju, ki je v i-ti vrstici in j-tem stolpcu, podamo število $k + 1$, če je rezultat $i + j$ sodo število, ali $k + 2$, če je liho.



Slika 5: Primer rešitve za nepopolno uganko.

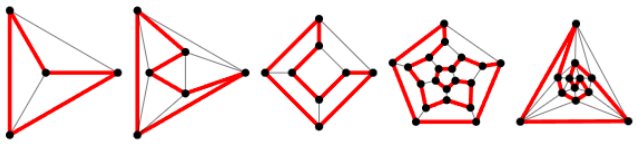
P sedaj dokazuje sledeči trditvi:

1. Vsaka končna celica ima natanko eno sosednje polje z enako številko.
2. Vsaka prehodna celica ima natanko dve sosednji polji z enako številko ali pa ima celica vrednost $k+1$ oziroma $k + 2$.

Postopek dokazovanja je enak kot pri popolni uganki, le da pri preverjanju prehodnih polj na dno tabele dodamo še štiri dodatne vrstice. Dve za število $k + 1$ in dve za število $k + 2$. Dodatne vrstice služijo za preverjanje pravilnosti v primeru, ko ima izbrano prehodno polje vrednost $k + 1$ ali $k + 2$.

4. HAMILTONOV CIKEL

Hamiltonov cikel je cikel, ki obišče vsako vozlišče v grafu natančno enkrat ter se začne in zaključi v istem vozlišču (Slika 6). Hamiltonov cikel v grafu z n vozlišči je cikel z n vozlišči. Če odstranimo eno povezavo iz Hamiltonovega cikla, dobimo Hamiltonovo pot. Ali takšna pot in cikel obstajata v grafu, sta NP-polna problema.



Slika 6: Primeri Hamiltonovih ciklov (povzeto po [14]).

4.1 Osnovni dokaz

Dokazovalec Patrik (Prover) želi dokazati preverjevalcu Viktorju (Verifier), da pozna Hamiltonov cikel v javnem grafu G, ne da bi ga razkril [1]. Viktor pozna graf G, vendar ne pozna Hamiltonovega cikla. Da Patrik dokaže, da pozna cikel, z Viktorjem odigrata več krogov igre, zato je ta dokaz interaktiven in verjetnosten. Uspešen rezultat na koncu zagotavlja Viktorju, da Patrik *verjetno* ne goljufa.

Naslednji koraki so izvedeni k krat, če P in V igrata k krogov.

(P1) Patrik iz grafa G ustvari njemu izomorfen graf H (enaka grafa z različnimi imeni vozlišč). Če Patrik pozna Hamiltonov cikel v grafu G, ga pozna tudi v grafu H, saj pozna preslikavo vozlišč.

(V1) Viktor naključno izbere eno od dveh izbir.

1. Patrik naj razkrije vsa vozlišča izomorfnega grafa H in poda preslikavo vozlišč iz G v H. Viktor lahko preveri, ali sta grafa resnično izomorfna.
2. Patrik naj razkrije Hamiltonov cikel: prevede Hamiltonov cikel iz G na H in odkrije le povezave Hamiltonovega cikla v grafu H. Viktor lahko preveri, ali H res vsebuje Hamiltonov cikel.

(P2) Patrik razkrije ustrezno izbiro.

Če Patrik v vsakem krogu pravilno prikaže željen graf ali Hamiltonov cikel, Viktor sprejme dokaz.

Dokaz brez razkritja znanja zadostuje naslednjim pogojem:

- Polnost: Če Patrik res pozna Hamiltonov cikel v grafu G, lahko zlahka zadovolji izbiram Viktorja. Razkrije mu izomorfni graf H, ali Hamiltonov cikel v H, ki ga lahko konstruira z lastno preslikavo iz G v H.
- Uglašenost: Če Patrik ne pozna Hamiltonovega cikla v G, lahko poskusi goljufati z ugibanjem naslednje Viktorjeve izbire in ustvari graf H ali Hamiltonov cikel za neki izmišljen graf. Vendar ravno zaradi neznanja ne more narediti obojega. V primeru, da bi naredil nasprotno od Viktorjeve izbire, bi ugotovil, da goljufa.
- Brez razkritja znanja: Ko je postopek končan, Viktor nima nobenega dodatnega znanja, ker odgovori Patrika ne razkrijejo prvotnega Hamiltonovega cikla v grafu G. V vsakem krogu se Viktor *nauči* samo preslikavo izomorfizma ali Hamiltonovega cikla v izomorfnem grafu H. V novem krogu ta informacija ni več veljavna, dokler Patrik lahko ustvari drugačen graf H. Viktor bi potreboval oba odgovora v istem krogu, da bi odkril Hamiltonov cikel v prvotnem grafu G. Prav tako nima Patrik nobenega dodatnega znanja, da bi lahko goljufal, ker ne ve naslednje Viktorjeve naključne izbire.

4.2 Fizični dokaz

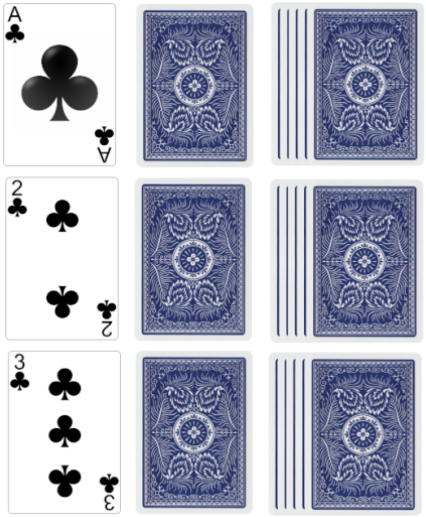
Fizični dokaz brez razkritja znanja lahko dobimo z uporabo igralnih kart z verjetnostjo goljufanja 0 [5]. Karte uporabimo za indeksiranje in informacijo o povezavah v grafu. Vsako povezavo predstavimo z barvo (♡) (srce) ali (♠) (pik). Označujemo pa s številko karte, kjer lahko poljubno razširimo interval številk v primeru velikih grafov. Števila zakodiramo kot $E_y(x)$, kjer število y predstavlja koliko pikovih kart imamo, pri tem pa je karta na poziciji x srčeva. Na primer, $E_3(2)$ predstavimo kot ♠♡♠, kjer je srce na drugem mestu v zaporedju velikosti 3. Za preverjanje Hamiltonovega cikla potrebujemo le $E_1(1) = ♡$, $E_1(0) = ♠$ in $E_n(k)$ za vse $0 <= k <= n$, kjer je n število vozlišč v grafu. Nad tabelo, dodamo vrstico označevalnih kart, kjer so potrebne.

Dokazovalec P zakodira rešitev s kartami. Vsako vozlišče mora biti namreč povezano s točno dvema povezavama, ki sta v ciklu. Hamiltonov cikel mora imeti točno n zaporednih povezav.

1. P naj skrivaj označi vsako povezavo, ki je v Hamiltonovem ciklu z $E_1(1) = ♡$ in vse ostale z $E_1(0) = ♠$.
2. P naj skrivaj označi vsako povezavo v ciklu z $E_n(k)$. Začne pri poljubno izbrani povezavi, ki jo označi z $E_n(1)$, nato nadaljuje z eno sosednjo povezavo, ki jo označi z $E_n(2)$ in tako ponavlja dalje. Sosednja vozlišča so na tak način kodirana z zaporednimi številkami. Povezave, ki niso v ciklu, naj bodo označene z $E_n(0)$.

Dokazovalec P in preverjevalec V nato vidno preverjata vsako vozlišče. Za vsako vozlišče sledi:

1. Iz kupčkov kart položenih na vsaki povezavi iz vozlišča, ki ga preverjamo, zgradimo tabelo. V stolpec 0 dodamo označevalne karte od 1 do d, kjer je d stopnja vozlišča, ki ga preverjamo. Za vsako povezavo iz vozlišča, ki ga preverjamo, v stolpec 1 skrito položimo ♡ ali ♠, glede na to ali spada povezava v Hamiltonov cikel ali ne. V stolpec 2 pa skrito položimo zakodiran kupček številke povezave. (Slika 7) Vse karte skrijemo.
2. Premešamo tabelo tako, da naključno permutiramo vrstice tabele (v nadaljevanju angl. *row scramble shuffle*). Vidno obrnemo vse karte v stolpcu 1 in poiščemo dve ♡ karti v tem stolpcu. Če jih ni, bo V že v tem koraku zavrnil rešitev, ker to pomeni, da ni vsako vozlišče povezano s točno dvema povezavama v ciklu.



Slika 7: Primer tabele s kartami za vozlišče s stopnjo 3 (povzeto po [5]).

3. Recimo, da sta dve ♡ karti v vrsticah i in j. Kupčke v teh vrsticah v stolpcu 2 vzamemo in uporabimo za drugo tabelo. V stolpec 0 nove tabele dodamo označevalne karte od 1 do n. Stolpec 1 naj bo skrito razporejen kupček kart, ki smo ga vzeli iz vrstice i. V stolpec 2 pa enako dodamo kupček kart, ki smo ga vzeli iz vrstice j.

4. Premaknemo vsako karto iz stolpca 2 za eno vrsto navzdol, tako da je zadnja karta zdaj v prvi vrsti.

5. Premešamo tabelo z uporabo *row scramble shuffle*. Vidno obrnemo vse karte v stolpcih 1 in 2 in poiščemo dve ♡ karti v katerikoli isti vrstici. V tem primeru, bi moral V sprejeti to vozlišče.

Karte je potrebno vrniti na prvotne položaje s postopkom preurejanja:

- Vse karte skrijemo in premešamo tabelo z *row scramble shuffle*.
- Vidno obrnemo označevalne karte v stolpcu 0 in z njihovo pomočjo preuredimo vrstice v pravilni vrstni red.
- Premaknemo vsako karto iz stolpca 2 za eno vrsto navzgor, tako da je prva karta zdaj v zadnji vrsti.

Če V v prejšnjem koraku ni sprejel vozlišča, ponovimo korake 4-5. V primeru, da še vedno ni dveh ♡ kart v isti vrstici, bo V v tem koraku zavrnil rešitev, Sicer sprejme vozlišče in naredimo postopek preurejanja, po katerem vrnemu kupčke kart v prvotno tabelo.

Preverjevalec V mora preveriti in sprejeti vsa vozlišča ter na koncu po premešanju vrstic in razkritju kart, še globalno preveriti, da obstaja točno en $E_n(i)$ za vsak $1 <= i <= n$. Šele v tem primeru preverjevalec sprejme rešitev.

Dokaz brez razkritja znanja zadostuje naslednjim pogojem:

- Polnost: Če dokazovalec poda pravi Hamiltonov cikel in postavi karte po pravilih, ima vsako vozlišče točno dve povezavi označeni z ♡ karto in bo preverjevalec sprejel vsako vozlišče in nato cikel.
- Uglašenost: Če dokazovalec ne poda pravega Hamiltonovega cikla, lahko poskusi goljufati z ugibanjem. Vendar bo s preverjanjem končnega števila vozlišč preverjevalec zavrnil graf, ker se bo zgodilo, da ima vozlišče več kot 2 povezavi ali pa bo v ciklu, ki ni Hamiltonov.
- Brez razkritja znanja: Ko je postopek končan, V nima nobenega dodatnega znanja. Karte so bile pred vsemi postavljene v tabele in premešane, vendar so bile večinoma skrite, hrbti kart pa so po izgledu enaki. Preverjevalec lahko dobi informacijo le ko so karte odkrite, vendar je v teh korakih prikazan le posamezen stolpec in karte so bile premešane.

5. KRIPTOGRAFIJA

Dokazi brez razkritja znanja imajo zagotovo zelo perspektiven pomen in njihova uporaba se iz dneva v dan veča, saj omogočajo povečanje zaščite in varnosti informacijskih sistemov in podatkov. Dokaz brez razkritja znanja je mogoče uporabiti pri marsikaterem problemu, kot recimo pri izomorfizmu grafov, za vse jezike v NP ipd. Pomembno vlogo pa igrajo tudi pri kriptografskih protokolih, in sicer pri procesu avtentikacije. Ena izmed najbolj znanih aplikacij je Fiat-Shamir-jev protokol.

5.1 Fiat-Shamirjev identifikacijski protokol

Fiat-Shamirjev indentifikacijski protokol je primer aplikacije dokaza brez razkritja znanja, ki se uporablja za potrebe identifikacije entitete. Predpostavimo da imamo entiteto Patrika (Prover), ki ima neko skrivnost S, ki jo lahko ve le on. Na drugi strani pa imamo Viktorja (Verifier), kateremu Patrik želi dokazati avtentičnost svoje identitete (torej, da ne gre za prevaranta), torej da dokaže, da pozna skrivnost S. Bistvo je v glavnem v tem, da Patrik Viktorju ne želi razkriti skrivnosti.

Fiat-Shamirjev protokol je sicer le osnova za protokole, ki temeljijo na dokazu brez razkritja znanja, v praksi se v modernih sistemih uporabljajo drugi protokoli, kot npr. Feige-Fiat-Shamirjev in Guillou-Quisquaterjev. Je pa prej omenjeni protokol pomemben, saj predstavi lastnosti, ki so pomembne v bolj sofisticiranih shemah. Protokol se izvede v dveh fazah [8]:

Inicializacija

1. Center zaupanja T izbere in javno objavi modul RSA, torej $n = pq$, kjer je n javen, p in q pa sta zasebna
2. Patrik izbere skrivnost s, ki je tuja n, tako da velja: $1 \leq s \leq n - 1$. nato izračuna $v = s^2 mod n$ in registrira v pri T-ju, kot svoj javni ključ.

Protokol identifikacije

Naslednji koraki so izvedeni t krat, vsakič z uporabo neodvisnega meta kovanca.

(P1) Patrik izbere naključno število r, $1 \leq r \leq n-1$ in pošlje $x = r^2 mod n$ Viktorju

(V1) Viktor naključno izbere bit $e \in \{0,1\}$ in pošlje e Patriku

(P2) Patrik izračuna y in ga pošlje Viktorju, kjer je $y = r$, če je $e = 0$ in $y = rsmodn$, če je $e = 1$

(V2) Viktor zavrne, če $y = 0$ ali $y^2 \ncong x\, v^e \,(mod\, n)$

Če je Viktor izvedel vse iteracije t, potem sprejme.

Fiat-Shamir identifikacijski protokol ustreza pogojem dokazu brez razkritja znanja.

- Polnost: predpostavimo, da ima dokazovalec Patrik skrivnost s. To pomeni, da lahko vedno preverjevalcu Viktorju dostavi $y = r$ ali $y = rs$, če ta za to zaprosi. Zanesljiv preverjevalec bo torej vedno uspešno izvedel vse iteracije z verjetnostjo 1.
- Uglašenost: predpostavimo, da Patrik nima skrivnosti s. Potem lahko Viktorju vedno dostavi enega od $y = r$ ali $y = rs$. Torej bo zanesljivi Viktor zavrnil y z verjetnostjo $\frac{1}{2}$ v vsaki iteraciji, kar predpostavlja, da verjetnost ne-prepoznave goljufanja znaša 2^{-t}.
- Brez razkritja znanja: Vsaka informacija, ki se razkrije v vsaki iteraciji je $x = r^2 \mod n$ in $y = r$ ali $y = rs$. Takšne pare (x,y) se simuliramo tako, da naključno izberemo y ter potem definiramo $x = y^2$ ali $x = \frac{y^2}{v}$. Takšni pari, ki sicer v protokolu niso generirani na enak način, so računsko neločljivi.

5.2 Diffie-Hellmanov algoritem za izmenjavo ključev

Diffie-Hellmanov algoritem za izmenjavo ključev sta v sodelovanju leta 1976 izumila Whitfield Diffie in Martin Hellman in to je bila prva praktična metoda za vzpostavitev skupne skrivnosti dveh udeležencev preko nezavarovanega komunikacijskega kanala [6]. Protokol uporablja multiplikativno grupo celih števil po modulu p$< Z_{p*}, x >$, kjer je p praštevilo. Morata pa udeleženca izbrati tudi število g, ki je generator v grupi. Postopek uporablja javne podatke, ki se zaradi problema rešljivosti lahko pošljejo tudi po nezavarovanem kanalu, na koncu imata udeleženca generirano skupno (enako) število, ki je tajna, ta skrivnost se kasneje uporablja za simetrično šifriranje tajnih podatkov. Koraki v protokolu so sledeči[6]:

1. Ana izbere veliko naključno število x (skrivno), tako da velja $0 < x < p$ in izračuna $R_1 = g^x mod\, p$.
2. Bor prav tako izbere veliko naključno število y (skrivno), tako da velja $0 < y < p$ in izračuna $R_2 = g^y mod\, p$.
3. Ana pošlje R_1 Boru.
4. Bor pošlje R_2 Ani.
5. Ana izračuna $K_{Ana} = (R_2)^x mod\, p$
6. Bor izračuna $K_{Bor} = (R_1)^y mod\, p$

K_{Ana} in K_{Bob} sta enaka, to je njuna skupna skrivnost.

Vendar ima prej omenjeni algoritem za izmenjavo ključev 2 glavni ranljivosti, in sicer: napad nad diskretni logaritem (angl. Discrete Logarithm attack) in napad srednjega moža (angl. man-in-the-middle attack). V nadaljevanju bomo predstavili Dokaz brez razkritja znanja za Diffie-Hellmanov algoritem za izmenjavo ključev, ki je odporen na napad srednjega moža.

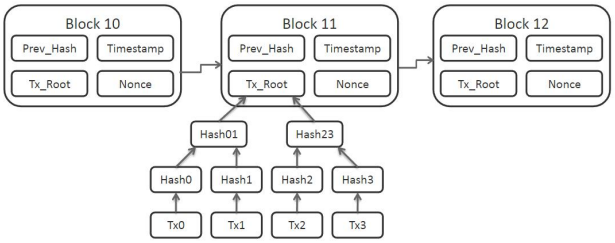
Dokazovalec (Patrik) želi dokazati preverjevalcu (Viktor) doakazati, da pozna skrivnost, tako da izračuna ključ K in Viktorju pošlje nazaj njegovo sporočilo (R_2), ki je kriptirano s skrivnim ključem K. Prav tako bo Viktor kriptiral svoj izračun (R_2) z skrivnim ključem K in če se šifra ujema, potem sprejme, sicer zavrne. Postopek je sledeč [6]:

1. Patrik izbere veliko naključno število x (skrivno), tako da velja $0 < x < p$ in izračuna $R_1 = g^x mod\, p$.
2. Patrik pošlje R_1 Viktorju.
3. Viktor prav tako izbere veliko naključno število y (skrivno), tako da velja $0 < y < p$ in izračuna $R_2 = g^y mod\, p$, $K_{Viktor} = (R_1)^y mod\, p$ in $C_1 = E(K_{Viktor}, R_2)$
4. Viktor pošlje $(R_2|C_1)$ Patriku.
5. Patrik izračuna $K_{Patrik} = (R_2)^x mod\, p$, dešifrira $R_2' = D(K_{Patrik}, C_1)$ in preveri ujemanje $R_2 = R_2'$. Če se ujemata, potem nadaljuje, sicer zavrne, saj je Viktor nepošten.
6. Patrik šifrira $C_2 = E(K_{Patrik}, R_1|R_2)$ in pošlje Viktorju.
7. Viktor dešifrira C_2 in dobi R_1' in R_2'.
8. Viktor preveri ujemanje $R_1 = R_1'$, če se ujema, potem sprejme, sicer zavrne.

Algoritem za izmenjavo skrivnosti ustreza pogojem za dokaz brez razkritja znanja:

- Polnost: Če sta Patrik in Viktor poštena, potem se mora protokol končati po končnem številu korakov z (sprejemom) ali (zavrnitvijo). Razlog za to je končna odločitev, ki je odvisna od izračunane vrednosti skrivnega ključa, ki je za oba udeleženca enak, ta je lahko enak [sprejmemo] ali različen [zavrnemo].
- Uglašenost: če Patrik izračuna nepravilno vrednost skrivnega ključa K, potem bo njegovo sporočilo $C_2' = E(KPatrik, R_2)$ različno od sporočila Viktorja; $C_2 = E(KViktor, R_2)$.
- Brez razkritja znanja: Ko je postopek končan, ne bosta imala udeleženca nobenega dodatnega znanja oz. informacije razen tistih števil, ki sta jih izračunala sama.

6. VERIGA BLOKOV



Slika 8: Zgradba verige blokov (povzeto po [13]).

6.1 Splošno o verigi blokov

Veriga blokov (angl. blockchain) je porazdeljena podatkovna baza, seznam povezanih zapisov, ki se nahaja na več vozliščih (blokih). Strukturo vidimo na sliki 8. Vsak blok vsebuje:

- množico digitalno podpisanih transakcij (lahko dokažemo njihov izvor),
- časovni žig in povezavo do prejšnjega bloka (kriptografska zgoščena vrednost),
- kriptografsko zgoščeno vrednost povzetka celotnega bloka .

Na ta način je opisano trenutno stanje ter celotna zgodovina vseh transakcij od začetka. Transakcije zapisane v bloku ni mogoče retroaktivno spreminjati, ker je za to operacijo potrebno ponovno generirati zgoščeno vrednost bloka [12].

Dva osnovna tipa verig blokov sta javni in zasebni tip.

- V javnih verigah so transakcije javno dostopne vsem, ter v njih lahko sodeluje kdor koli z internetno povezavo. Za vzpostavitev sistema nagrajevanja in kaznovanja članov v njih pridejo v poštev znanja ekonomije in teorije iger (angl. game theory). Pošteni člani, ki delujejo v prid sistema so nagrajeni in nepošteni so kaznovani. Najbolj znana primera sta Bitcoin in Ethereum.
- Zasebne verige so delno regulirane, saj v njih in njihovih transakcijah lahko sodelujejo le določeni člani (npr. znotraj enega podjetja). Njihova varnost temelji na poznavanju članov. Zato ni potrebe po ekonomskih spodbudah, tj. kriptovalutah, ter energijsko zahtevnim rudarjenjem. Posledično so bolj učinkovite in skalabilne, ampak hkrati ne zadoščajo definiciji popolne decentraliziranosti verige blokov [2].

6.2 Motivacija - dokaz brez razkritja znanja v verigi blokov

Podajmo en slikovit primer. Vsak Bitcoin račun/denarnica ima enolično določen naslov, ki je javno dostopen in zapisan ob vsaki transakciji. Posledično poznavanjem le-tega je možno rekonstruirati celotno zgodovino transakcij. Kaj če želimo to preprečiti? Oseba A želi nakazati osebi B znesek X, ampak kako brez razkrivanja zasebnih podatkov prepričati osebo B da oseba A zares ima dovolj sredstev na svojem računu? Lahko v proces vpeljemo mediatora, entiteto ki ji obe strani zaupajo in bo zagotovila pošteno transakcijo. Toda, ob drugi strani s takšno odločitvijo smo odstranili decentraliziranost verige.

Na tem mestu postaja motivacija očitna, želeli bi obdržati decentralizirano strukturo javnih verig, hkrati pa imeti zasebnost zasebnih verig. Denimo da akterji v verigi blokov imajo nek skupni cilj. Želijo ostvariti ta cilj, ki je ostvarljiv le s skupnimi močmi, ampak hkrati nočejo razkriti zaupnih podatkov. Torej, gre se za paradoksalno situacijo podobno klasičnem "smrtnem objemu" (angl. deadlock). Prav v ta namen so se začele verige blokov oplemenjevati z mehanizmi dokaza brez razkritja znanja. Dokaz brez razkritja znanja diskretno in s skoraj stoprocentno gotovostjo prepriča akterje da je zadovoljen predpogoj, ne da bi jim odkril kakršnokoli novo koristno informacijo .

6.3 zkSNARK

zkSNARK (angl. Zero-knowledge succinct non-interactive argument of knowledge) je ena najbolj znanih inačic klasičnega dokaza brez razkritja znanja.

Na kratko predstavimo princip delovanja algoritma, podobno kot je opisan v [7]. zkSNARK se sestoji iz treh algoritmov G, P, V.

Algoritem G (angl. generator) kot vhod dobi skriti parameter λ ter program C, generira pa dokazovalni ključ pk in verifikacijski ključ vk, kar vidimo v enačbi 1.

Algoritem P (angl. prover) izračuna dokaz. Kot vhod dobi pk, javni vhod x in "pričo" w, generira pa dokaz π, kot je razvidno v enačbi 2.

Algoritem V (angl. verifier) preveri dokaz. Kot vhod dobi vk, x in π ter vrne $true$ če je dokaz pravilen, sicer vrne $false$. Matematično to zapišemo v enačbi 3.

$$G(\lambda, C) = (pk, vk) \tag{1}$$

$$\pi = P(pk, x, w) \tag{2}$$

$$V(vk, x, \pi) == (\exists\ w, takšen\ da\ velja\ C(x, w)) \tag{3}$$

Ključne lastnosti zkSNARK-a so sledeče:

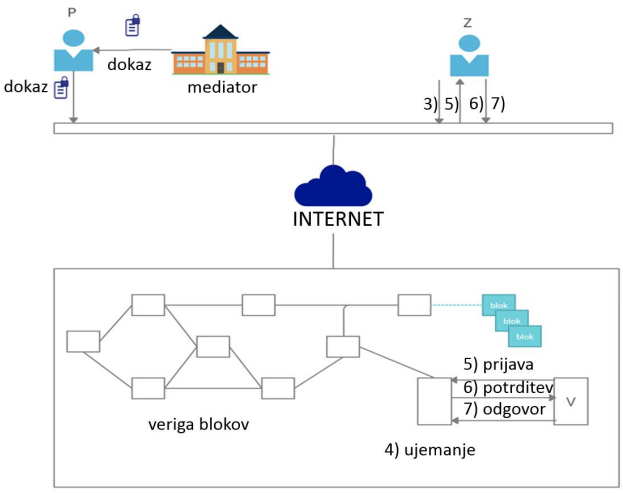
- je dokaz brez razkritja znanja indirektnega tipa
- je izvedljiv v polinomskem času
- sam dokaz ima velikost le nekaj bajtov
- verifikacijo izvaja zelo hitro

Te lastnosti so predpogoj in razlog zakaj so zkSNARK modeli uporabni tudi v resničnih scenarijih. Dokaz je indirektnega tipa, zaradi česa je proces olajšan, ker ni potrebe po direktni point-to-point komunikaciji med dokazovalcem in verifikatorjem. Potem so ključni časovno-prostorski prihranki.

Dokaz je izjemno majhen, in se izvaja v polinomskem času, kar pomeni da pojav veliko zaporednih zahtevkov, če je dokaz pravilno implementiran, ne bo preveč obremenila verigo blokov.

Konkretno na primeru verige blokov se zkSNARK izvede takole (proces je slikovito ponazorjen na sliki 9):

1. Entiteta mediator generira vk in pk, nato še dokaz π, ki dokazuje ne vrednost računa osebe P, temveč dejstvo da je ta vrednost znotraj potrebnega intervala.
2. Oseba P vstavi dobljeni dokaz v verigo blokov.
3. Oseba Z zahteva verifikacijo, preverjanje verodostojnosti dokaza. Pošlje paket, ki vsebuje verifikacijsko nalogo, oznako (angl. tag), nagrado ter rok za odgovarjanje.
4. Vozlišče v verigi preveri ujemanje oznak. V primeru zadetka posreduje zahtevo verifikacijskem vozlišču V.
5. V nazaj pošlje ujemajočo oznako in časovni žig.
6. V primeru pravilne oznake in pravočasneg odgovora Z dokončno potrdi zahtevek dokaza. Vozlišču V pošlje potrditveno sporočilo.
7. V izvede algoritem. Dokaz pošlje nazaj, spet z ujemajočo oznako in časovnim žigom
8. Z dobi odgovor. Nagrada se dodeli V [11].



Slika 9: zkSNARK v verigi blokov (povzeto po [11]).

6.4 zkSNARK modeli

Kot vidimo proces je dokaj standardiziran. Ključna razlika pa je v zgradbi osrednjega algoritma - dejanska implementacija dokaza brez razkritja znanja, ki se izvede v koraku 7. Algoritem je odvisen od treh parametrov, c - velikosti dokaza, d - globine dokaza, s - velikosti vhodov dokaza (pozor: velja $c \gg d$ in $c \gg s$). V tabeli 1 so navedene značilnosti in časovno-prostorske zahtevnosti sodobnih zkSNARK modelov, ki ta korak izvedejo na zelo različne načine. Tabela povzema izjemno bogato analizo modelov, ki so jo naredili Sun in sod. v [11]. Modeli, ki so opisani so: Ben-Sassonov, Ligero, Bulletproof, Hyrax, Aurora ter Libra.

	varna inicializacija (mediator)	zahtevnost dokaza	zahtevnost verifikacije	velikost dokaza	tehnika
Ben-Sassonov	Da	$O(c\log c)$	$O(1)$	$O(1)$	kvadratne enačbe
Ligero	Ne	$O(c\log c)$	$O(c)$	$O(\sqrt{c})$	interaktiven dokaz
Bulletproof	Ne	$O(c)$	$O(c)$	$O(\log c)$	diskreten logaritem
Hyrax	Ne	$O(c\log c)$	$O(\sqrt{s}+d\log c)$	$O(\sqrt{s}+d\log c)$	interaktiven dokaz
Aurora	Ne	$O(c\log c)$	$O(c)$	$O(\log^2 c)$	interaktiven dokaz
Libra	Da	$O(c)$	$O(d\log c)$	$O(d\log c)$	interaktiven dokaz

Table 1: Ključne značilnosti zkSNARK modelov

Metrika, ki je v praksi najbolj relevantna je zagotovo zahtevnost dokaza. V tem prednjačita modela Bulletproof in Libra, ki uspešno izvedeta dokaz v linearnem času. Libra izhaja iz zelo učinkovitega dokaza, ki so ga izpeljali Goldwasser in sod. ter preverljive sheme polinomskega dodeljevanja, ki so jo opisali Zhang in sod. Velika prednost Libre je ta da zahteva le enkratno inicializacijo, preko zaupne entitete, dočim Ben-Sassonov vedno vnovič zahteva inicializacijo. Zaradi najboljših rezultatov v ključni metriki in zelo kompetitivnih rezultatih v ostalih, denimo da je Libra najbolj konkurenčna implementacija zkSNARK-ov v verigi blokov.

7. ZAKLJUČEK

Tehnika dokaza brez razkritja znanja je sprva morda zelo neintuitivna, a je zato izjemno elegantna in zanimiva, na koncu se je izkazalo, tudi zelo uporabna. Metodo dokaza smo spoznali na področju enigmatike, preko originalnega članka o Numberlinku. Nato smo se prepričali da tudi obstoj Hamiltonovega cikla lahko dokažemo na isti način. Zgodbo smo končali z manj zabavnimi, a bolj praktičnimi področji, kriptografijo in verigo blokov. Lahko torej sklepamo: tehnika dokaza brez razkritja znanja je vsestransko uporabna, saj smo predstavili uspešne primere iz teorije grafov, teorije iger in kriptografije.

8. LITERATURA

[1] M. Blum. How to prove a theorem so no one else can claim it. In *Proceedings of the International Congress of Mathematicians*, volume 1, page 2. Citeseer, 1986.

[2] D. Gašperlin. *Integracija verige blokov in tehnologij semantičnega spleta*. PhD thesis, Univerza v Ljubljani, Fakulteta za računalništvo in informatiko, 2021.

[3] O. Goldreich, S. Micali, and A. Wigderson. Proofs that yield nothing but their validity or all languages in np have zero-knowledge proof systems. *J. ACM*, 38(3):690–728, jul 1991.

[4] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208, 1989.

[5] E. Hart and J. A. McGinnis. Physical zero-knowledge proofs for flow free, hamiltonian cycles, and many-to-many k-disjoint covering paths. *arXiv preprint arXiv:2202.04113*, 2022.

[6] M. K. Ibrahem. Modification of diffie-hellman key exchange algorithm for zero knowledge proof. pages

147–152, 04 2012.

[7] C. Lundkvist. Introduction to zk-snarks with examples, 2017. https://media.consensys.net/introduction-to-zksnarks-with-examples-3283b554fc3b, dostopano 08.05.2022.

[8] A. Mohr. A survey of zero-knowledge proofs with applications to cryptography. 01 2007.

[9] J.-J. Quisquater, M. Quisquater, M. Quisquater, M. Quisquater, L. Guillou, M. A. Guillou, G. Guillou, A. Guillou, G. Guillou, and S. Guillou. How to explain zero-knowledge protocols to your children. In G. Brassard, editor, *Advances in Cryptology — CRYPTO' 89 Proceedings*, pages 628–631, New York, NY, 1990. Springer New York.

[10] S. Ruangwises and T. Itoh. Physical zero-knowledge proof for numberlink puzzle and k vertex-disjoint paths problem. *New Gen. Comput.*, 39(1):3–17, apr 2021.

[11] X. Sun, F. R. Yu, P. Zhang, Z. Sun, W. Xie, and X. Peng. A survey on zero-knowledge proof in blockchain. *IEEE Network*, 35(4):198–205, 2021.

[12] R. Taş and Tanrıöver. Building a decentralized application on the ethereum blockchain. In *2019 3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*, 2019.

[13] M. Wander. Bitcoin blockchain structure, 2013. https://upload.wikimedia.org/wikipedia/commons/5/55/Bitcoin_Block_Data.svg, dostopano 08.05.2022.

[14] E. W. Weisstein. Hamiltonian cycle. *From MathWorld–A Wolfram Web Resource.*, 2003.