Eötvös Loránd Tudományegyetem
Természettudományi Kar

BSc Szakdolgozat
Alkalmazott matematikus szakirány

Nem feltáró bizonyítások és alkalmazásaik

Készítette:
Csatári Jakab

Témavezető:
Hubai Tamás

Belső konzulens:
Grolmusz Vince

2019

Tartalomjegyzék

Köszönetnyilvánítás

Szeretném megköszönni témavezetőmnek, Dr. Hubai Tamásnak rengeteg segítségét és hogy felkeltette érdeklődésemet a számítástudomány témakörében. Köszönöm, hogy segített a témaválasztásban, nagyon jó volt ezzel a témával foglalkozni. Külön köszönöm a közös munkát és a sok energiát, melyet a konzultációkra fordított, hogy alaposságával és tanácsaival egyengette a szakdolgozatom alakulását.

Köszönöm Dr. Grolmusz Vincének, hogy szívesen vállalta, hogy a belső konzulensem legyen és segítette a szakdolgozatom létrejöttét.

Köszönöm családomnak, hogy mindenben támogattak, ezzel is hozzájárulva, hogy a dolgozatom elkészüljön. Köszönöm feleségemnek és nővéremnek, hogy alaposan átnézték a dolgozatomat, és segítségükkel számos nyelvtani hibát ki tudtam javítani.

Bevezetés

Tegyük fel, hogy van egy titok, amit nagyon értékes tudni. Ha tudod ezt a titkot, és elárulod másnak, azzal elveszti az értékét. Ha viszont nem árulod el senkinek, akkor senki sem tudja, hogy tényleg tudod ezt a titkot. Milyen jó lenne, ha be tudnád bizonyítani valakinek, hogy tényleg tudod a titkot, anélkül, hogy elárulnád azt.

A nem feltáró bizonyítások ezt teszik lehetővé. Ez egy matematikai bizonyítási módszer, melyet a 20. század végén kezdtek el kutatni, először 1989-ben Shafi Goldwasser, Silvio Micali, és Charles Rackoff a "The Knowledge Complexity of Interactive Proof-Systems" című tanulmányukban. Azóta rengeteg cikk jelent meg ezel a témakörrel kapcsolatban és igen fontos matematikai eredmények születtek ezen a téren.

A nem feltáró bizonyítás egy olyan eljárás, mely során egy személy egy állítás helyességét tudja bizonyítani, de más információt nem szolgáltat ki.

A dolgozatomban elsősorban a nem feltáró bizonyítások különféle alkalmazásait szeretném bemutatni. Mind matematikai szempontból fontos, absztrakt feladatokra, mind valós gyakorlati problémákra készített alkalmazásokat érinteni fogok, mint például nukleáris fegyver leszerelést, vagy kriptovaluták tranzakcióit.

A szakdolgozatomat négy részre osztottam, az első fejezetben a nem feltáró bizonyítások elméletét szeretném bemutatni, a többi fejezetben pedig eljárásokat mutatok, ahol nem feltáró bizonyításokat használunk. A második fejezetben bevezető példákat mutatok ebben a témakörben, melyek célja, hogy a nem feltáró bizonyítások szemléletét bemutassa. A harmadik fejezetben néhány fontosabb matematikai problémára fogunk látni alkalmazást. Az utolsó fejezetben pedig a gyakorlatban ténylegesen használt protokollokat mutatok be.

1. Nem feltáró bizonyítások és változatai

Matematikai értelemben bizonyításnak hívunk egy állítás helyességét igazoló eljárást. A klasszikus bizonyítási módszerek (pl. direkt, vagy indirekt bizonyítás, teljes indukció) determinisztikusak, nem függnek semmilyen interakciótól. Bárki, aki a bizonyítás helyességét szeretné ellenőrizni, végül egyértelműen győződhet meg róla, hogy valóban helyes, vagy sem.
A nem feltáró bizonyítások ettől kissé eltérő megközelítését használják a bizonyítás fogalmának. Ebben a fejezetben erről a megközelítésről írok, és a nem feltáró bizonyítások különböző változatairól. A nem feltáró bizonyításként fordított "zero knowledge proof" fogalmát Shafi Goldwasser, Silvio Micali és Charles Rackoff vezette be 1989-ben. A magyar irodalomban több helyen nullaismeretű bizonyításnak hívják.

1.1. Nem feltáró bizonyítás

A továbbiakban a következőképp tekintsünk a bizonyításokra: Adott két fél, a **bizonyító** és az **ellenőr**. Továbbá adott egy állítás, melyről a bizonyító tudja, hogy igaz-e, de az ellenőr nem. Az eljárás során a bizonyító azt próbálja meg bebizonyítani az ellenőrnek, hogy az állítás igaz, viszont egyikük sem tekinti a másikat megbízhatónak. Az ellenőr nem tudja, hogy a bizonyító nem próbálja-e meg átverni, úgy hogy egy hamis állítást próbálna meg igaznak feltüntetni. Illetve a bizonyító sem bízik meg abban, hogy az ellenőrnek nincs más szándéka, mint hogy az állítás helyességét megtudja. Ez utóbbinak elsőre nem tűnik úgy, hogy jelentősége lenne, nemsokára viszont látni fogjuk, hogy a nem feltáró bizonyítások legtöbbször interakcióra épülnek. Így az ellenőr hatással lesz arra, hogy a bizonyítás során mit tud meg a bizonyítótól.

Ahhoz, hogy a fenntartásaik ellenére bizonyítékot szolgáltathasson a bizonyító, egy protokollt (előre meghatározott eljárást) fognak használni, melynek a bizonyító eleget tud tenni, amennyiben az állítás valóban igaz. Ezt az eljárást fogjuk bizonyításnak nevezni. Legtöbbször a protokoll nem determinisztikus, egy konkrét kimenetelét bizonyítéknak hívjuk. Azonban arra, hogy a másik fél valóban követi-e a protokollt, csak az egymás közti kommunkációból fognak tudni következtetni.

Jelölés: Ezentúl jelöljük a bizonyítót $\boldsymbol{P}$-vel, az ellenőrt $\boldsymbol{V}$-vel. A bizonyítandó állításra pedig $\boldsymbol{S}$-ként fogunk hivatkozni.

Általában egy bizonyítás során $\boldsymbol{V}$ többet tud meg annál, hogy $\boldsymbol{S}$ igaz. Tekintsük a következő példát:

Adott $n \in \mathbb{N}$. $\boldsymbol{S}$:= "Az n szám k darab prímtényező szorzatára bomlik fel."

Tegyük fel, hogy $\boldsymbol{P}$ ismeri a prímtényezős felbontását n-nek, $\boldsymbol{V}$ viszont nem, és n elég nagy, illetve $\boldsymbol{V}$ nem rendelkezik megfelelő számítási kapacitással ahhoz, hogy az általa ismert algoritmusok bármelyikével (belátható időn belül) faktorizálja. Ekkor ha a bizonyítási eljárás abból áll, hogy $\boldsymbol{P}$ elmondja az összes prímtényezőjét n-nek, akkor bár $\boldsymbol{V}$ nem bízik $\boldsymbol{P}$-ben, de ellenőrizni tudja, hogy ezek a prímtényezők. (Valóban prímek és szorzatuk n.)

Ebben az esetben viszont $\boldsymbol{V}$ nem csak annyit tud meg, hogy $\boldsymbol{S}$ igaz, hanem n faktorizációját is. A nem feltáró bizonyítások olyan eljárások, mely során $\boldsymbol{V}$ nem tud meg semmilyen többletinformációt, legfeljebb annyit, hogy $\boldsymbol{S}$ igaz-e.

Definíció 1.1: Azt a protokollt, mely során $\boldsymbol{P}$ egy $\boldsymbol{S}$ állítást igazol $\boldsymbol{V}$-nek, nem feltáró bizonyításnak nevezzük, ha a következők teljesülnek:

- Teljesség: Ha $\boldsymbol{S}$ igaz, $\boldsymbol{P}$ és $\boldsymbol{V}$ követik a protokollt, akkor az eljárás végén $\boldsymbol{V}$ meggyőződik az állítás helyességéről.
- Megalapozottság: Ha $\boldsymbol{S}$ hamis, akkor $\boldsymbol{V}$ legfeljebb kis valószínűséggel fogadja el igaznak - függetlenül attól, hogy $\boldsymbol{P}$ követi-e a protokollt.
- Nem feltáró: Az eljárás során $\boldsymbol{V}$ nem tud meg mást, csak annyit, hogy $\boldsymbol{S}$ (szinte biztosan) igaz, vagy megbizonyosodik róla, hogy $\boldsymbol{S}$ hamis.

Megjegyzés 1.1.a: Az alatt, hogy $\boldsymbol{V}$ kis valószínűséggel fogadja el igaznak az állítást, azt értjük, hogy valamely $0 \leq \varepsilon$ valós számra annak a valószínűsége, hogy $\boldsymbol{S}$ hamis és $\boldsymbol{V}$ elfogadja, nem nagyobb, mint ε. Jellemzően egy protokoll ismétléseket tartalmaz addig, míg adott $0 < \varepsilon$-nál kisebb nem lesz ez a valószínűség.

Megjegyzés 1.1.b: Alapesetben a definíció feltételei kell teljesüljenek, azonban bizonyos esetekben szeretnénk, hogy egyéb feltételek is teljesüljenek:

- Ha egy külső szemlélődő hallja a $\boldsymbol{P}$ és $\boldsymbol{V}$ közti kommunikációt, nem bizonyosodhat meg semmiről. Nem tudhatja meg sem azt, hogy $\boldsymbol{S}$ igaz-e, sem azt, hogy $\boldsymbol{V}$ meggyőződött-e $\boldsymbol{S}$ helyességéről.
- A nem feltáró feltétel teljesüléséhez csak annyit követelünk meg, hogy $\boldsymbol{V}$ ne tudhasson meg mást, mint hogy $\boldsymbol{S}$ igaz-e, ha $\boldsymbol{V}$ követi a protokollt. Azonban létezik olyan nem feltáró bizonyítás, mely elveszti a nem feltáró tulajdonságát, ha $\boldsymbol{V}$ szándékosan nem követi a protokollt. (Erre fogunk látni példát a **3.2** fejezetben.) Azonban az ilyen protokollokat $\boldsymbol{P}$ szeretné elkerülni, így a továbbiakban elsősorban olyan példákkal foglalkozunk, melyekre $\boldsymbol{V}$ akkor sem tudhat meg semmi többletinformációt, ha nem követi a protokollt.

Megjegyzés 1.2: Ha $\boldsymbol{S}$ hamis, $\boldsymbol{P}$-nek csak a protokoll nem követésével van (ε-nál nem nagyobb) esélye azt $\boldsymbol{V}$-vel elhitetnie.

1.2. Számítási modell

Ahhoz, hogy formalizálni tudjuk $\boldsymbol{P}$-t és $\boldsymbol{V}$-t, szükségünk van valamilyen számítási modellre, illetve meg kell határozzuk, hogy mit értünk a kommunkációjuk alatt.

Legyenek $\boldsymbol{P}$ és $\boldsymbol{V}$ nemdeterminisztikus Turing gépek. $\boldsymbol{S}$ pedig formalizálható a következőképp: $\boldsymbol{S}$:= "$x \in L$, valamely L nyelvre."
(Az előbbi példára: $L = L_k = \{n \in \mathbb{N} : n\ k$ darab prím szorzata$\}$.)

$\boldsymbol{P}$ és $\boldsymbol{V}$ rendelkeznek egy-egy saját inputszalaggal, egy-egy outputszalaggal, illetve van egy közös szalagjuk is. A közös szalagon először x áll. $\boldsymbol{P}$ és $\boldsymbol{V}$ felváltva számolnak és megállásukkor írnak a saját outputszalagjukra, illetve a közös szalagra. Egészen addig számolnak, amíg $\boldsymbol{V}$ saját outputja ELFOGAD, vagy ELUTASÍT. (Ezzel meghatározva, hogy $\boldsymbol{P}$ bizonyítását, azaz $\boldsymbol{S}$-t elfogadja-e.)

Ekkor $\boldsymbol{P}$ és $\boldsymbol{V}$ kommunikációján azt értjük, amit futásuk során a közös szalagra írtak.

Sok nem feltáró bizonyítás azon a feltevésen alapul, hogy valamely függvény inverzét nehéz kiszámolni. Vagyis nem ismert rá hatékony algoritmus, hogy az input hosszában polinomiális időben lehessen kiszámítani. Feltesszük, hogy $\boldsymbol{P}$ és $\boldsymbol{V}$ nem ismer algoritmust nehéznek vélt problémák gyors kiszámítására, és az input hosszában polinomiális algoritmusokat tudnak kiszámolni. Tehát $\boldsymbol{P}$ legjobb esélye $\boldsymbol{V}$ megtévesztésére az inverz megtippelése lesz ezekben az esetekben.

1.3. Nem feltáró bizonyítások változatai

A nem feltáró bizonyításokat sok szempontból tudjuk jellemezni, ezek közül néhány fontosabbat szeretnék felsorolni, melyek megjelennek, illetve végigkísérnek minket a dolgozatban szereplő példákon.

A tudás bizonyítása: A nem feltáró bizonyítás azon alapul, hogy $\boldsymbol{P}$ birtokában van valamilyen információ (melyet $\boldsymbol{V}$ nem tud), azonban az eljárás során ezt a titkát megőrzi. Ha ennek a titkos ismeretnek hiányában $\boldsymbol{S}$ nem bebizonyítható, akkor azzal, hogy $\boldsymbol{P}$ bizonyítja $\boldsymbol{S}$-t, egyúttal bizonyítja tudását $\boldsymbol{V}$-nek. (Azt a bizonyítékot, melyet így szolgáltat $\boldsymbol{V}$-nek, az angol irodalomban zero knowledge proof or knowledge-nek, vagy zero knowledge argument of knowledge-nek nevezik.)
A dolgozatban szereplő példák többségében $\boldsymbol{P}$ a tudását fogja bizonyítani úgy, hogy valamely jól megválasztott $\boldsymbol{S}$-hez szolgáltat nem feltáró bizonyítást.

Interaktív és nem interaktív nem feltáró bizonyítások: A tudás bizonyításában elengedhetetlen szerepet játszik, hogy a nem feltáró bizonyítások nem-determinisztikusak. A nem feltáró bizonyítások mindegyikéhez szükség van $\boldsymbol{P}$-től független eseményre, ez legtöbb esetben interakció $\boldsymbol{V}$-vel.

Interaktív bizonyítás esetén $\boldsymbol{P}$ bizonyítása függ attól, hogy mit kommunikál $\boldsymbol{V}$ neki. Tehát a protokoll olyan, hogy $\boldsymbol{V}$ az eljárás során bizonyos kérdéseket tesz fel, $\boldsymbol{P}$ pedig a kérdések függvényében folytatja a bizonyítását.

Azt mondjuk, hogy a nem feltáró bizonyítás **nem interaktív**, ha csupán $\boldsymbol{P}$ küld üzeneteket $\boldsymbol{V}$-nek, $\boldsymbol{V}$ pedig csak ellenőrzi, hogy a kapott üzenetek alapján igaznak véli-e $\boldsymbol{S}$-t. A nem interaktív nem feltáró bizonyítások előnye, hogy nem szükséges $\boldsymbol{P}$ és $\boldsymbol{V}$ között (egyidejű) kommunikációt biztosítani, $\boldsymbol{P}$ bármikor előállíthatja a bizonyítékot, $\boldsymbol{V}$ pedig ellenőrzi, amikor megkapja az üzenetet, vagy üzenetcsomagot. Másik előnye, hogy az **1.1.b** megjegyzés második feltétele automatikusan teljesül, mivel $\boldsymbol{V}$ nem kérdez, nincs hatással $\boldsymbol{P}$ bizonyítására,

így nem nyerhet ki plusz információt $\boldsymbol{P}$-től. Azonban a nem interaktív megvalósításokhoz is szükség van arra, hogy $\boldsymbol{P}$ valamilyen tőle független inputo(ka)t kapjon, ezt legtöbbször egy $\boldsymbol{P}$ és $\boldsymbol{V}$ által is megbízhatónak vélt random generátor biztosítja.

A dolgozatban szereplő példák többnyire interaktívak, de a **2.6**, **4.2** és **4.4** alkalmazásokban látni fogunk nem interaktív bizonyításokat.

Perfekt, statisztikai és számításbeli nem feltáró bizonyítások: Tekintsük ε-t, ahogy **1.1.a** megjegyzésben megadtuk: ε a hamis állítás elfogadásának valószínűségére felső korlát. Ha a protokoll olyan, hogy $\varepsilon = 0$ -ra fennáll a megalapozottság, akkor **perfekt** nem feltáró bizonyításról beszélünk.

Statisztikai nem feltáró bizonyításról beszélünk, ha a protokoll olyan, hogy bármely $\varepsilon > 0$ -ra elérhető a megalapozottság. Vagyis, ha adott ε, a valószínűsége annak, hogy $\boldsymbol{V}$ elfogad egy hamis $\boldsymbol{S}$-t, nem nagyobb mint ε. Ebből egyértelműen következik, hogy a statisztikai változatnak részhalmaza a perfekt.

A statisztikai nem feltáró bizonyítások sokszor többszörös iterációt tartalmaznak. Nevezzük egy iterációnak a $\boldsymbol{P}$ és $\boldsymbol{V}$ közti kommunikáció egy részét. Ha egy eljárás olyan, hogy egy iteráció során $\boldsymbol{P}$ legfeljebb $\frac{1}{2}$ valószínűséggel veri át $\boldsymbol{V}$-t, akkor azt mégegyszer megismételve, az ismétléskor ugyanúgy $\frac{1}{2}$ a valószínűsége, hogy $\boldsymbol{P}$ át tudja verni $\boldsymbol{V}$-t. Feltéve, hogy a kommunikáció ismétlése az előzőtől független, annak a valószínűsége, hogy $\boldsymbol{P}$ egymás után kétszer is elhitessen egy hamis $\boldsymbol{S}$-t legfeljebb $\frac{1}{2^2}$. Hasonlóan n ismétlés után $\frac{1}{2^n}$.

Sok példában használni fogjuk, hogy ha egy iterációban legfeljebb $\frac{1}{2}$ valószínűséggel fogad el $\boldsymbol{V}$ hamis állítást, akkor az ismétléses módszerrel hatékonyan és tetszőlegesen javíthatunk a megalapozottságon.

A számításbeli nem feltáró bizonyításhoz szükségünk van az egyirányú függvény definíciójára.

Jelölés: $\{0,1\}^* = \{\{0,1\}^n : n \in \{\mathbb{N} \cup \{0\}\}\}$

Definíció 1.2: $f : \{0,1\}^* \to \{0,1\}^*$ egyirányű függvény, ha:

- $\exists c \geq 1$ konstans, hogy $|x|^{\frac{1}{c}} \leq |f(x)| \leq |x|^c$
- $f(x)$ polinom időben számolható.
- $\forall A : \{0,1\}^* \to \{0,1\}^*$ polinom idejű randomizált algoritmusra, ha y egy véletlen eleme $\{0,1\}^n$-nek, akkor annak a valószínűsége, hogy $f(A(f(y))) = f(y)$ elhanyagolható n függvényében.

Ha egy nem feltáró bizonyítás valamely egyirányú függvény nehézségén alapul, akkor az **számításbeli** nem feltáró. Azt mondjuk, hogy a protokoll számításbeli nem feltáró, ha ahhoz, hogy $\boldsymbol{P}$ az **1.1** definícióban megadott kis valószínűségnél nagyobb valószínűséggel tudja átverni $\boldsymbol{V}$-t, ismernie kellene gyors algoritmust az adott egyirányú függvény visszafejtésére.

Többszemélyes nem feltáró bizonyítások: A nem feltáró bizonyítások kiterjeszthetőek. Többszemélyes nem feltáró bizonyítás esetén nem csak egy bizonyító van és általában nem is csak egy állítás, hanem minden bizonyító birtokában van valamilyen ismeretnek. A dolgozatban a többszemélyes változatnak azt a formáját fogjuk használni, ahol adott valahány $\boldsymbol{P}$, akik egyúttal $\boldsymbol{V}$-k is egymásra nézve. Azaz minden bizonyító szeretné meggyőzni a saját tudásáról az összes többi bizonyítót. Erre látni fogunk egy leegyszerűsített példát **2.5**-ben, és egy alkalmazását **4.4**-ben.

2. Szemléltető példák

Ebben a fejezetben néhány egyszerűbb példát szeretnék bemutatni. Ezen példák többségének önmagukban nem feltétlenül van közvetlen gyakorlati haszna, azonban jól bemutatják a nem feltáró módszerek különböző változatainak alkalmazásait. Ezeknek az eljárásoknak célja elsősorban a nem feltáró bizonyítási módszerek szemléltetése. A bemutatott problémák [1], [2], [3], [4] és [5]-ből származnak.

2.1. Színvak barát

Az első példa egy statisztikai (de nem perfekt) nem feltáró bizonyítás lesz.

Adott két személy $\boldsymbol{P}$ és $\boldsymbol{V}$. $\boldsymbol{P}$ teljesen jól lát mindent, a szemei egészségesek, $\boldsymbol{V}$ viszont a zöld és piros színek bizonyos árnyalatait azonosnak látja.
Egy asztalon van két egyforma méretű és formájú golyó, a egyik zöld, a másik piros színű. $\boldsymbol{V}$ a két golyót teljesen egyformának látja, $\boldsymbol{P}$ viszont látja a különbséget a golyók között.

$\boldsymbol{S}$:= {A két golyó különböző színű.} $\boldsymbol{P}$ az állítást nem feltáró módon szeretné bizonyítani $\boldsymbol{V}$-nek. Más szóval $\boldsymbol{P}$ szeretné, ha $\boldsymbol{V}$ tudná, hogy a golyók különböző színűek, de azt nem, hogy melyik a zöld és melyik a piros. Ekkor a protokoll a következőképp nézhet ki:

1. $\boldsymbol{P}$ elfordul és megvárja, míg $\boldsymbol{V}$ az egyik golyót elveszi az asztalról és a háta mögé rejti.
2. $\boldsymbol{P}$ visszafordul és megnézi, melyik maradt az asztalon.
3. $\boldsymbol{P}$ újra megfordul, $\boldsymbol{V}$ eldönti, hogy kicseréli-e az asztalon lévő golyót a másikra, vagy sem.
4. Visszafordulva $\boldsymbol{P}$ megmondja, hogy ki lett-e cserélve a golyó.

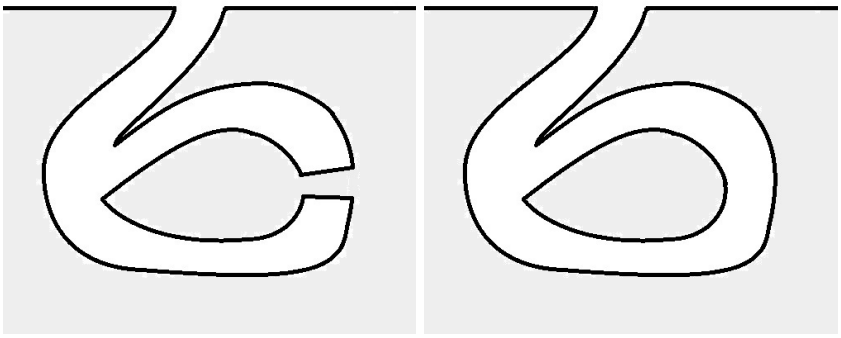
$\boldsymbol{V}$ ekkor még nincs túlságosan meggyőződve (hiszen $\boldsymbol{P}$ $\frac{1}{2}$ eséllyel jól is tippelhetett), ezért megismétlik néhányszor a 3., 4. lépéseket.

Tegyük fel, hogy $\boldsymbol{V}$ 0.0001-nél kisebb valószínűséget szeretne hagyni neki, hogy $\boldsymbol{P}$ átveri. Ekkor a módszert 14-szer megismételve annak a valószínűsége, hogy $\boldsymbol{S}$ hamis és $\boldsymbol{V}$ elfogadja $\frac{1}{2^{14}} \approx 0.00006$. A példában jól látszik az is, hogy "tökéletesen" biztos sosem lehet $\boldsymbol{S}$ helyességében $\boldsymbol{V}$, viszont ha $\boldsymbol{P}$ egyszer rosszul válaszol, akkor $\boldsymbol{V}$ teljesen biztos lehet abban, hogy $\boldsymbol{P}$ átveri, azaz ő sem lát különbséget a golyók színe között.

2.2. Ali baba barlangja

Egy másik közismert bevezető példa a nem feltáró bizonyításokba Ali baba barlangja. Erre a példára két protokollt mutatunk, jól fogjuk tudni szemléltetni, hogyan valósítható meg az **1.1.b** megjegyzés első feltétele a perfektség kárára.

A feladat a következő. Adott egy barlang az ábrán látható módon. Ali baba elől egy rabló mindig el tudott menekülni ebben a barlangban. Ali baba úgy tudja, hogy a barlang úgy néz ki, mint a bal oldali ábrán látjuk. Azonban a rabló ismer egy varázsszót, melyet kimondva a barlang belseje pillanatnyilag átjárhatóvá válik, mint a jobb oldali ábrán.



Ábra a barlangról.

Legyen $\boldsymbol{P}$: a rabló, $\boldsymbol{V}$: Ali baba. $\boldsymbol{P}$ bizonyítani szeretné $\boldsymbol{V}$-nek, hogy körbe tud menni a barlang belsejében, de nem szeretné, ha $\boldsymbol{V}$ megtudná a varázsszót.
Egy lehetséges nem feltáró bizonyítás:

1. $\boldsymbol{V}$ az elágazásnál megáll, $\boldsymbol{P}$ bemegy az egyik folyosón.
2. $\boldsymbol{P}$ kijön a másik folyosón.

Ekkor $\boldsymbol{P}$ titkát $\boldsymbol{V}$ nem tudja meg, viszont biztos lehet abban, hogy $\boldsymbol{V}$ ismeri a varázsszót, különben nem lett volna képes körbemenni. A bizonyítás perfekt, hiszen az eljárás végén $\boldsymbol{V}$ minden kétsége megszűnik, $\boldsymbol{P}$ valóban körbe tudott menni.

Az **1.1.b** megjegyzésben leírt első plusz elvárás azt fogalmazza meg, hogy ha egy harmadik személy (nevezzük $\boldsymbol{M}$-nek) lehallgatja a kommunikációjukat, akkor nem tudhat meg semmit. A példára levetítve ez azt jelenti, hogy $\boldsymbol{M}$ állhat az eljárás közben ugyanott, mint $\boldsymbol{V}$ (azaz az elágazásnál). Ekkor viszont, ha $\boldsymbol{P}$ körbemegy, akkor $\boldsymbol{M}$ is meggyőződik arról, hogy $\boldsymbol{P}$ ismeri a varázsszót.
Ahhoz, hogy ezt elkerüljük, statisztikai nem feltáró bizonyítást használunk a következőképpen:

1. $\boldsymbol{V}$ a barlang bejáratánál áll (nem az elágazásnál). $\boldsymbol{P}$ bemegy a barlangba.
2. $\boldsymbol{P}$ az elágazásnál véletlenszerűen valamelyik folyosón bemegy. ($\boldsymbol{V}$ nem tudja melyikbe ment.)
3. $\boldsymbol{V}$ kisvártatva követi és hangosan kiáltja, hogy a jobb vagy a bal oldali folyosón jöjjön ki.
4. $\boldsymbol{P}$ kijön ott, ahol $\boldsymbol{V}$ várja.

Ha $\boldsymbol{P}$ valóban tudja a varázsszót, bármelyik folyosón ki tud jönni, tehát eleget tud tenni $\boldsymbol{V}$ kérésének, ezt bármennyiszer megismételheti. Azonban ha $\boldsymbol{M}$ végignézi az egész eljárást ($\boldsymbol{V}$ szemszögéből), akkor nem tudhatja, hogy esetleg nem beszéltek-e össze egymással. ($\boldsymbol{P}$ és $\boldsymbol{V}$ megállapodhattak valamilyen szisztéma szerint, hogy mikor melyik oldalon fogja $\boldsymbol{V}$ várni $\boldsymbol{P}$-t.)

Így tehát, $\boldsymbol{P}$ és $\boldsymbol{V}$ elérhetik, hogy egy harmadik fél számára ne szolgáltassanak semmilyen információt, cserébe viszont egy kis esélyt kell hagyjon rá $\boldsymbol{V}$, hogy $\boldsymbol{P}$ esetlegesen hamis állítását elfogadja.

2.3. Pénzfeldobás telefonon

A következő példa arról fog szólni, hogy ha $\boldsymbol{P}$ hoz egy döntést valamivel kapcsolatban, akkor képes olyan bizonyítékot szolgáltatni $\boldsymbol{V}$-nek, mely igazolja, hogy $\boldsymbol{P}$ valóban döntött, de $\boldsymbol{V}$ nem tud meg semmit a döntéséről. Ezt a módszert "kulcsosládikás" módszernek fogjuk nevezni, ugyanis legtöbb esetben később ezt a titkát $\boldsymbol{P}$ szeretné felfedni. (Ez az elnevezés pedig utal az analógiára, hogy $\boldsymbol{P}$ bezárja a titkát egy kulcsosládikába, melyet csak ő képes feloldani később.)
$\boldsymbol{P}$ módszere tehát úgy működik, hogy meghoz egy döntést, erről nem feltáró bizonyítékot szolgáltat $\boldsymbol{V}$-nek, majd később felfedi a titkát, ekkor $\boldsymbol{V}$ ellenőrizheti, hogy a korábbi bizonyíték valóban a felfedett döntéshez tartozik. Az ilyen fajta bizonyítások rendszerint valamilyen egyirányú függvényt használnak.

Erre a módszerre a legegyszerűbb példa a következő: $\boldsymbol{P}$ és $\boldsymbol{V}$ valamit el szeretnének dönteni egymás között. Erre egy tökéletes módszer a fej vagy írás játék, ez a módszer azonban csak akkor működik jól, ha $\boldsymbol{P}$ és $\boldsymbol{V}$ egy helyen tartózkodik. Tegyük fel, hogy $\boldsymbol{P}$ és $\boldsymbol{V}$ csak telefonon tudnak kommunikálni. Nem feltáró bizonyítást használva így is tudnak fej vagy írást játszani. A következőt szeretnénk tehát elérni: $\boldsymbol{P}$ véletlenszerűen választ fejet vagy írást (mondjuk tényleg feldob egy érmét). Erről nem feltáró bizonyítást küld $\boldsymbol{V}$-nek, $\boldsymbol{V}$ tippel, majd $\boldsymbol{P}$ felfedi választását. Ehhez a négyzetes maradékprobléma nehézségét fogjuk felhasználni.

Definíció 2.1: Legyenek p, q prímek, $n = pq$. $\mathbb{Z}_n^*$ jelölje azt a csoportot, mely n azon maradékosztályait tartalmazza, melyek reprezentatív eleme relatív prím n-nel. Legyen:

$$Q_1 := \{m \in \mathbb{Z}_n^* : \exists x \in \mathbb{Z}_n^*\ x^2 \equiv m \pmod{p} \text{ és } x^2 \equiv m \pmod{q}\} \text{ és}$$
$$Q_2 := \{m \in \mathbb{Z}_n^* : \exists x \in \mathbb{Z}_n^*\ x^2 \not\equiv m \pmod{p} \text{ és } x^2 \not\equiv m \pmod{q}\}$$

Ekkor Q_1 elemeit négyzetelemeknek, Q_2 elemeit álnégyzetelemeknek hívjuk.

Állítás: Legyen $Q = Q_1 \cup Q_2$. (Ezek triviálisan diszjunkt halmazok.) p és q ismeretének hiányában is könnyű egy $m \in \mathbb{Z}_n^*$ elemről megállapítani, hogy $m \in Q$, vagy sem. Azonban nem ismert olyan módszer, mely hatékony annak eldöntésére, hogy egy $m \in Q$ -ról megállapítsuk, hogy m négyzetelem, vagy sem. Ezt hívjuk négyzetes maradékproblémának.

Az előbbieket felhasználva, tehát a protokollunk a következőképp néz ki:

1. $\boldsymbol{P}$ választ két prímet: p, q; $n = pq$ és egy $m \in \mathbb{Z}_n^*$ álnégyzetelemet.
2. $\boldsymbol{P}$ elküldi $\boldsymbol{V}$-nek n, m-et.
3. $\boldsymbol{P}$ választ egy véletlen $x \in \mathbb{Z}_n^*$ -ot, valamint egy $b \in \{0, 1\}$ értéket, ahol a 0 a fejnek, 1 az írásnak felel meg.
4. $\boldsymbol{P}$ elküldi $\boldsymbol{V}$-nek az $y \equiv m^b x^2 \pmod{n}$ értéket.
5. $\boldsymbol{V}$ megtippeli b értékét.
6. $\boldsymbol{P}$ elküldi b, x, q, p értékeket.

Végül $\boldsymbol{V}$ ellenőrizheti, hogy az $y \equiv m^b x^2 \pmod{n}$ egyenlőség fennáll.

Az eljárás jól működik, mert $\boldsymbol{V}$ nem tudja megállapítani b értékét az 5. lépés előtt. Ugyanis $b = 0$ esetén y négyzetelem, de ha $b = 1$, akkor y álnégyzetelem. Ha $\boldsymbol{V}$ bármit meg tudna állapíani b-ről, akkor megtudná oldani a négyzetes maradékproblémát. Ugyanakkor $\boldsymbol{P}$ sem tud csalni, mert ahhoz az kellene, hogy létezzen x_1, x_2, hogy $m^0 x_1^2 \equiv m^1 x_2^2 \pmod{n}$. Azaz: $(x_2 x_1^{-1})^2 \equiv m \pmod{n}$. De ez nem lehet, mert m álnégyzetelem.

Megjegyzés 2.1: Ha m nem álnégyzetelem, akkor a 6. lépésben ez kiderül $\boldsymbol{V}$ számára. $\boldsymbol{P}$-nek érdeke bebizonyítani, hogy $\boldsymbol{V}$ rosszul tippelt, azonban, ha $\boldsymbol{V}$ jól tippel, $\boldsymbol{P}$ lehetséges, hogy nem szeretné, hogy $\boldsymbol{V}$ erről megbizonyosodjon. Ezért a 6. lépésben előfordulhat, hogy az előzőekhez képest inkonzisztens értékeket küld, vagy egyáltalán nem küld semmit. Hogy ezt kiküszöböljük, úgy tekintheti $\boldsymbol{V}$, hogy ha $\boldsymbol{P}$ nem követi a protokollt (pl. m nem álnégyzetelem, vagy az egyenlőség nem áll fenn y-ra), akkor jól tippelt.

2.4. Utolsó lépés tárolása sakkban

A probléma, amiről ebben a részben írok, az előzőhöz hasonló. Itt is a döntés meglétét szeretnénk bizonyítani.

Tekintsük a következő esetet: $\boldsymbol{P}$ és $\boldsymbol{V}$ sakkoznak. A következő lépés $\boldsymbol{P}$-jé, de valamilyen oknál fogva fel kell függesszék a játszmát. Később szeretnék folytatni, viszont egyikük sem szeretné, hogy amíg nem játszanak, a másikuk kigondolhassa következő lépését, hogy ezzel előnyre tegyen szert. Ezért azt szeretnék elérni, hogy $\boldsymbol{P}$ kitalálja mit fog lépni, ezt a döntését a játszma folytatásáig már ne változtathassa meg, viszont $\boldsymbol{V}$ csak akkor tudja meg milyen lépést talált ki $\boldsymbol{P}$, amikor legközelebb folytatni fogják.

Adott tehát egy állás a sakktáblán. $\boldsymbol{P}$ lehetséges lépéseinek száma véges, mivel 9000-nél biztosan kevesebb, így négyjegyű számokkal el tudunk kódolni minden lehetséges lépést. (Bizonyos kifejezetten ritka esetekben is 100-200 körüli lehetséges lépése van $\boldsymbol{P}$-nek.) Így tehát elég a lépéshez rendelt kódra alkalmazni kulcsosládikás módszert.

Vegyük észre, hogy az előző protokollt használhatjuk ennek a feladatnak a megoldásához is. Írjuk át a négyjegyű számokat 2-es számrendszerbe, ekkor legfeljebb 14 jegyű számokat kapunk (ha ennél kevesebb jegyű, egészítsük ki az elején 0-kkal). Ekkor a kettes számrendszerbeli szám minden számjegyéről tud $\boldsymbol{P}$ nem feltáró bizonyítékot szolgáltatni az előző protokollt 14-szer alkalmazva - annyi különbséggel, hogy az 5., 6. lépéseket kihagyja, és csak a következő játszmakor fedi fel $\boldsymbol{P}$ a megfelelő értékeket. $\boldsymbol{P}$ minden iterációban különböző p, q, m, x értékeket érdemes válasszon. (Valójában a legtöbb esetben kevesebb jegyű 2-es számrendszerbeli számokkal is el tudjuk kódolni a lehetséges következő lépéseket.)

Láthatjuk tehát, hogy ha egy problémát hatékonyan tudunk 0-1 kódolni, akkor a kódolásról szolgáltatott nem feltáró bizonyítás többszörös iterációval megvalósítható, ha a fej vagy írás problémára vezetjük vissza. Sőt, a kommunikációt egy üzenettel is meg lehet oldani: az előző fejezetben ismertetett egyirányú függvényre k üzenet esetén $n_1, ..., n_k, m_1, ..., m_k, y_1, ..., y_k$ értékeket egyben elküldheti $\boldsymbol{P}$.

Ez a módszer nem feltétlenül a leghatékonyabb minden problémára, előfordulhat, hogy túl sok/túl nagy méretű üzenetet szükséges $\boldsymbol{P}$-nek lekommunikálnia $\boldsymbol{V}$-vel. A sakkos példára ismertetünk egy másik bizonyítást is:

1. Most $\boldsymbol{P}$ és $\boldsymbol{V}$ a lépések eredeti, 10-es számrendszerbeli kódolását használják. Megállapodnak, egy N számban, hogy $2N$ jegyű összetett számot ne tudjanak polinomiális időben faktorizálni.

2. $\boldsymbol{P}$ kiegészíti a lépést kódoló négyjegyű számot egy p N jegyű prímmé. (Ezt úgy teheti meg, hogy kiegészíti egy véletlen N jegyűvé és prímteszteli. Ha nem prím, újrapróbálkozik. A prímszámtétel miatt várhatóan $log(10^N)$ próbálkozásra van szüksége ehhez.)

3. Hasonlóan generál egy véletlen q $N + 1$ jegyű prímet, majd elküldi az $n = pq$ számot $\boldsymbol{V}$-nek.

4. A következő játszma előtt $\boldsymbol{P}$ felfedi a két prímszámot. $\boldsymbol{V}$ ellenőrzi, hogy a szorzatuk valóban n és p N jegyű.

5. p első négy számjegye megadja a következő lépés kódját.

Így a kommunikáció mennyiségén tényleg spórolunk, egy iterációval értük el ugyanazt, mint az előbb. Sőt ez a módszer általánosan javít egy szám (kódolás) nem feltáró bizonyításának kommunikáció-hatékonyságán.

Megjegyzés 2.2: Ezt a példát annyival érdemes kiegészíteni, hogy minden négyjegyű számhoz rendeljen $\boldsymbol{P}$ és $\boldsymbol{V}$ egy lépést, azaz egy lépést több számmal is elkódolnak. Hiszen, ha $\boldsymbol{P}$ olyan számot egészítene ki, melyhez nem létezett kódolás, azt $\boldsymbol{V}$ nem tudná ellenőrizni, mikor felfüggesztik a játszmát. Ekkor ugyan kiderül a játszma folytatásakor, hogy $\boldsymbol{P}$ csalt, de $\boldsymbol{V}$ már nem tudna mit tenni, az eltelt idő alatt $\boldsymbol{P}$ végiggondolhatta következő lépését, ezzel elrontva a játékot. (Másik lehetőség, hogy ekkor $\boldsymbol{V}$ automatikusan nyer.)

2.5. Kártyázás telefonon

A telefonon kártyázás egy remek példa a többszemélyes nem feltáró bizonyítás bemutatására. Ebben a részben egy kártyapakli szétosztásának a problémájáról lesz szó. Most négyszemélyes modellt fogunk használni, melyben mindenki bizonyító és ellenőr is egyben, ezért a szokásos $\boldsymbol{P}$, $\boldsymbol{V}$ jelölés helyett használjuk az $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$, $\boldsymbol{D}$ jelöléseket.

A következő problémát szeretnénk megoldani: Négy személy $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$, $\boldsymbol{D}$ kártyázni szeretnének (egy hagyományos francia kártyapaklival). Mindezt telefonos kommunikációval szeretnék megoldani, mert mind a négyen különböző helyen tartózkodnak. A feladat azon részével, amikor a kezükben lévő lapokat kijátszák a játékosok, nincs is probléma. Viszont a kártyapakli megfelelő szétosztásának megvalósítása már kérdéses. A játékosok egy olyan eljárást szeretnének, mellyel telefonon tudják négy részre osztani a paklit, nem feltáró módon. Vagyis azt szeretnék, hogy az osztás végén mindenki biztos legyen abban, hogy minden lap (pontosan egyszer) kiosztásra került, viszont mindenki csak a saját kártyalapjait ismerje.

Többszemélyes nem feltáró bizonyításoknál általában elvárjuk (ahogy ez **4.4** -ben teljesülni is fog), hogy minden kommunikáció publikus legyen. Vagyis ha $\boldsymbol{A}$ kommunikál $\boldsymbol{B}$-nek, azt mindenki láthassa, ezzel elérve, hogy bizonyos személyek ne tudjanak együttműködni a többiek kárára. Ebben a feladatban az egyszerűség kedvéért ezt nem várjuk el. A felek bármekkora részhalmaza tud privát csatornán kommunikálni, és feltesszük, hogy nem beszélnek össze. (Olyan kártyajátékot játszanak, melyben nem tudnak előnyre szert tenni azzal, hogy néhányan összebeszélnek.)

Mindez megoldható a következő módon:

1. Minden játékos elkódolja az összes kártyát egy 1-52 közti számmal. Az előző feladattól eltérően most minden játékos magának kódol és nem osztja meg a többiekkel. $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$, $\boldsymbol{D}$ kódolását jelöljük sorra a, b, c, d-vel. A játékosok publikus üzenetben nem feltáró bizonyítást adnak arról, hogy mi a kódolásuk, "kulcsosládikás" módszerrel.

2. Privát üzenetben $\boldsymbol{A}$-nak mindenki elküldi a saját kódolását. $\boldsymbol{B}$ elküldi b-t, $\boldsymbol{C}$ c-t, $\boldsymbol{D}$ d-t.

3. $\boldsymbol{A}$ elkészít egy-egy "szótárat" $a \leftrightarrow b$, $a \leftrightarrow c$, $a \leftrightarrow d$ nyelvek között. $a \leftrightarrow x$ szótár alatt azt értjük, hogy az egyforma lapokat jelölő a kódokhoz (bijektív) hozzárendeli az x kódokat. (De azt nem tartalmazza, hogy melyik laphoz tartoznak.)

4. $\boldsymbol{A}$ elküldi privát üzenetben az $a \leftrightarrow b$ szótárat $\boldsymbol{C}$-nek, az $a \leftrightarrow c$ szótárat $\boldsymbol{D}$-nek és az $a \leftrightarrow d$ szótárat $\boldsymbol{B}$-nek.

5. Ezután a következőképpen osztanak: $\boldsymbol{B}$, $\boldsymbol{C}$ és $\boldsymbol{D}$ nyitnak egy közös privát csatornát ($\boldsymbol{A}$-t szükséges kihagyni, mert ekkor már minden kódolást ismer). Veszik a $H := \{h \text{ kód} : h \in a\}$ halmazt.

6. $\boldsymbol{B}$, $\boldsymbol{C}$ és $\boldsymbol{D}$ az előbbi csatornán keresztül felváltva választanak maguknak egy H-beli elemet, melyet egyben törölnek is H-ból. Addig, amíg 13 kód nem marad H-ban.

7. $\boldsymbol{C}$ privát üzenetben elküldi $\boldsymbol{B}$-nek azon b kódokat, melyek mefelelnek a $\boldsymbol{B}$ által választott a-beli kódoknak. Hasonlóan $\boldsymbol{D}$ privát módon elküldi $\boldsymbol{C}$-nek a megfelelő c kódokat és $\boldsymbol{B}$ elküldi $\boldsymbol{D}$-nek a d kódokat.

8. Végül egy közös publikus üzenetben $\boldsymbol{A}$ megkapja a H-ban maradt kódokat a-beli kódolással. Azon kódokról, melyeket $\boldsymbol{B}$, $\boldsymbol{C}$, $\boldsymbol{D}$ egymás között osztottak szét, közösen kulcsosládikás módszerrel elküldik $\boldsymbol{A}$-nak, melyik kód melyikükhöz került. Ezután kezdhetik a játszmát.

9. A játszma végén mindenki felfedi a saját kódolását, illetve $\boldsymbol{B}$, $\boldsymbol{C}$ és $\boldsymbol{D}$ feltárja a 8. lépésben nyújtott bizonyítékukat. Ekkor a játékosok összevetik ezeket. $\boldsymbol{B}$, $\boldsymbol{C}$ és $\boldsymbol{D}$ ellenőrzik, hogy a szótárak, melyeket kaptak valóban helyesek, illetve az egymástól kapott kódok helyesek voltak-e. $\boldsymbol{A}$ a 8. lépésben számára küldött bizonyítást ellenőrzi.

Nézzük meg, hogy ez az eljárás valóban eleget tesz a feladat kívánalmainak:

- Minden lapot pontosan egyszer osztanak ki, hiszen H minden eleme pontosan egyszer valamelyik játékoshoz kerül. H-ban pedig az összes kártyának az $\boldsymbol{A}$ által létrehozott kódja van az osztás elején.

- Mindenki tudja az osztás végén, hogy milyen lapjai vannak. $\boldsymbol{A}$ tudni fogja, mert neki 8. lépésben megmondják az ő kódolása szerint. A többieknek pedig a 7. lépésben mondják el a saját kódolásuk szerint. (Fontos látni, hogy a 7. lépésben az a közvetítő személy, aki "lefordítja" a nála lévő szótár szerint, hogy az a-beli kód milyen másik kódot jelent, valójában nem tud semmit. Pl. $\boldsymbol{C}$ az $a \leftrightarrow b$ szótár segítségével meg tudja mondani $\boldsymbol{B}$-nek, hogy a $\boldsymbol{B}$ által választott a-beli kódok milyen b-beli kódok. De azt, hogy ezek mely kártyáknak felelnek meg, csak $\boldsymbol{B}$ fogja tudni.)

- Senki sem tudja az osztás végén, hogy azok a lapok, melyek nem hozzá kerültek, kinél vannak. $\boldsymbol{A}$ azért nem, mert az osztás folyamatából kimarad. $\boldsymbol{B}$, $\boldsymbol{C}$, $\boldsymbol{D}$ pedig az előbbi zárójeles megjegyzés miatt nem.

- A játék végén mindenki ellenőrizni tudja, hogy (önállóan) nem volt képes senki sem csalni. $\boldsymbol{B}$, $\boldsymbol{C}$ és $\boldsymbol{D}$ pontosan tudja a nyelven, hogy kihez milyen lap került, ezért a játék végén elő tudják állítani az $\boldsymbol{A}$ által generált szótárakat. Ez alapján pedig ellenőrizni tudják, hogy mindenki azzal a lappal játszott-e, amit kapott, vagy sem. $\boldsymbol{A}$ pedig a 8. lépésben neki nyújtott bizonyíték feltárásakor tudja ellenőrizni ugyanezt.

Megjegyzés 2.3: Az 1. lépésben nyújtott kulcsosládikás módszer nézhet ki például a következőképpen: A pakli lapjait valamilyen sorrendben közösen sorbarendezik. A játékosok miután létrehozták a saját kódjaikat, ebben a sorrendben elküldik a soron következő kártyalap kódolásáról tett bizonyítékot, úgy ahogyan a sakkos példában tette $\boldsymbol{P}$ egy számra.
Hasonlóan a 8. lépésben ugyanígy járhatnak el. 1-3 számmal kódolják, hogy melyikükhöz került egy lap, majd az a kódhalmaz szerinti növekvő sorrendben küldhetnek bizonyítékot.

2.6. Sudoku megoldás bizonyítása

Ez a példa egy egyszerű szemléltetése annak, hogy a tudás bizonyításához nem feltétlenül szükséges interakció. Most az 1. fejezetben említett $\boldsymbol{P}$-től független esemény kártyakeverésből származó véletlen lesz. Tehát feltesszük, hogy amikor $\boldsymbol{P}$ megkever néhány kártyát, azok véletlenszerű sorrendbe kerülnek.

A példa így szól: $\boldsymbol{P}$ ismeri egy nehéz sudoku feladvány megoldását. $\boldsymbol{P}$ szeretné bizonyítani, hogy van megoldása, de nem szeretné bizonyításával megmutatni, hogy mi a megoldás. (Valójában egy rendesen kitűzött sudoku megoldását megtalálni gyerekjáték egy számítógépnek, tehát a feladat valóban csak szemléltető jellegű.)

Az eljáráshoz számkártyákat fogunk használni. Tegyük fel az egyszerűség kedvéért, hogy $\boldsymbol{V}$ jelen van ott ahol $\boldsymbol{P}$, és tudja ellenőrizni, hogy végig a megfelelő számkártyákat és a megfelelő módon használja. Nézzük tehát a protokollt:

1. $\boldsymbol{P}$ elővesz egy számkártya csomagot, melyben pontosan 27 darab van minden számból 1-9-ig.
2. Minden sudoku mezőre három egyforma számkártyát helyez lefordítva, mégpedig azokat, amelyek a megoldásában az adott mezőre kerülnek. Néhány mező a feladat kitűzésében meg van adva, ezeket felfordítja.
3. Minden lefordított mezőn lévő 3 kártyát megkever és visszateszi a mezőre továbbra is lefordítva.
4. 27 paklit alkot a következő módon: Minden sor legfelső lapjaiból egy-egy paklit képez, így marad minden mezőn 2 számkártya. Majd minden oszlop legfelső lapjaiból képez egy-egy paklit. Végül az egyes blokkokban maradt számkártyákból képez egy-egy paklit.
5. Minden egyes pakilt megkever, majd megmutatja, hogy 1-9-ig minden szám pontosan egyszer szerepel minden pakliban.

A megoldás azért lesz helyes, mert ha valóban van $\boldsymbol{P}$-nek megoldása, akkor így feltenni a számkártyákat nem jelent problémát számára, és (nagy valószínűséggel) bizonyítja ezzel, hogy egy sorban, egy oszlopban és egy blokkban minden szám pontosan egyszer szerepel. Tegyük fel, hogy $\boldsymbol{P}$ úgy próbálna csalni (tehát valójában nem ismeri a megoldást), hogy minden oszop, sor és blokk szerint valahogyan szétosztja az 1-9 lapokat. Ekkor annak az esélye, hogy minden mezőn 3 egyforma lesz, olyan kicsi, mint hogy véletlenszerűen pont megoldja a sudokut. Várhatóan a mezőkön különböző számok lesznek lefordítva, ekkor annak a valószínűsége, hogy nem bukik le, ugyanolyan kicsi, mint annak, hogy az összes mezőn megkevert három lap ugyanolyan permutációban kerül vissza. Ez a valószínűség függ attól, hogy eredetileg hány mező kitöltetlen, ha például 52 mező üres, akkor $\frac{1}{6^{52}} \approx 3.44 \cdot 10^{-41}$-nel egyenlő, ami igazán kicsi. A protokoll megismételhető, amennyiben egy iteráció nem győzné meg $\boldsymbol{V}$-t kellőképpen.

3. Absztrakt matematikai alkalmazások

Az előző fejezetben már láttuk, hogy a nem feltáró bizonyítások hogyan működnek néhány "kevésbé életszerű" példán. Ebben a fejezetben az lenne a célom, hogy nem feltáró eljárásokat mutassak néhány matematikai szempontból fontos problémára is.

3.1. Diszkrét logaritmus

Ez a feladatkör a kriptográfiában széleskörben használt, például személyazonosság igazolására, vagy két fél közti nyilvános kulcscseréhez. A diszkrét logaritmus nehézségét azért szeretik széleskörben használni, mert az inverzét, a modulo hatványozást gyorsan lehet számolni, illetve implementálni is igen könnyű.

Definíció 3.1: Legyen G egy n-edrendű multiplikatív ciklikus csoport. $g \in G$ a generátora (tehát $(n, g) = 1$) és $a \in G$. Ekkor $a \equiv g^x \pmod{n}$ valamely $x \in \mathbb{N}$-re. Azt mondjuk, hogy x az a diszkrét logaritmusa G-n ezen adott g mellett. ($x \pmod{n}$ egyértelmű.)

Jelölés: Ekkor az $x = log_g(a)$ jelölést használjuk.

Bizonyos speciális esetekben a diszkrét logaritmus kiszámítása lehet gyors, azonban megfelelően választott csoport mellett nem ismert hatékony algoritmus, mellyel egy $a \in G$ értékhez diszkrét logaritmust tudnánk számolni. Tipikusan $n = p$ és $G = \mathbb{Z}_p$, ahol p egy nagy prímszám.

Tegyük fel, hogy $\boldsymbol{P}$ ismeri egy $a \in G$ érték diszkrét logaritmusát. Erről szeretne nem feltáró bizonyítékot nyújtani $\boldsymbol{V}$-nek. Ezt a következő módon teheti meg:

$\boldsymbol{S}$: $\boldsymbol{P}$ ismeri $x = log_g(a)$ értéket G-n, ahol G rendje egy p prím.

1. $\boldsymbol{P}$ generál egy random r értéket és kiszámolja $C \equiv g^r \pmod{p}$ -t.
2. $\boldsymbol{P}$ elküldi C-t $\boldsymbol{V}$-nek.
3. $\boldsymbol{V}$ eldönti, hogy megkérdezi $\boldsymbol{P}$-t, hogy mi a generált r érték, vagy pedig megkérdezi $(x + r) \pmod{p - 1}$ értéket.
4. $\boldsymbol{P}$ $\boldsymbol{V}$ választásától függően elküldi a kívánt értéket.
5. (a) Ha $\boldsymbol{V}$ r-et kérdezte meg, akkor ellenőrzi, hogy $C \equiv g^r$ fennáll-e.
   (b) Egyébként ellenőrzi, hogy $g^{(x+r)(mod\ p-1)} \pmod{p} \equiv C \cdot a \pmod{p}$ fennáll-e.
6. Az 1-5. lépéseket néhányszor megismétlik.

Amennyiben $\boldsymbol{S}$ igaz, $\boldsymbol{P}$-nek egyáltalán nem jelent problémát követni a protokollt. Ekkor az 5.b lépésben az egyenlőség fennáll, mert:

$$g^{(x+r)(mod\ p-1)} \pmod{p} \equiv g^x g^r \pmod{p} \equiv a \cdot C \pmod{p}$$

Ha viszont $\boldsymbol{P}$ nem ismeri az x értéket, akkor sokszor egymás után kis eséllyel tudja $\boldsymbol{V}$-t átverni.

Állítás: Tegyük fel, hogy $\boldsymbol{P}$ nem ismeri x-et, ekkor a valószínűsége, hogy egy iterációban nem bukik le $\frac{1}{2}$.

Bizonyítás: Ha $\boldsymbol{P}$ nem ismeri a diszkrét logaritmust, akkor nem tud $\boldsymbol{V}$ bármelyik kérdésére válaszolni, legfeljebb az egyikre.

- Ha $\boldsymbol{P}$ generál egy random r értéket és valóban a $C \equiv g^r$ -et küldi el a 2. lépésben, akkor ha $\boldsymbol{V}$ r-re kérdez rá, akkor $\boldsymbol{P}$ azt el tudja küldeni és az 5. lépésben $\boldsymbol{V}$ jóváhagyja. Azonban, ha $\boldsymbol{V}$ az $(x+r) \pmod{p-1}$ értéket kérdezi meg, akkor $\boldsymbol{P}$ nem tud helytállóan válaszolni. (Fontos, hogy ekkor már a 2. lépésben C értékét elküldte.) Ha tudna, az azt jelentené, hogy ki tudja számolni $log_g(Ca)$-t.
- Ha $\boldsymbol{P}$ nem véletlen generált g^r-t küld el, akkor helyette elküldheti a $C \equiv g^r \cdot a^{-1}$ értéket. Így, ha $\boldsymbol{V}$ az $(x+r) \pmod{p-1}$ értéket kérdezi meg, amire $\boldsymbol{P}$ r-et válaszol, akkor az 5. lépésben $\boldsymbol{V}$ jóváhagyja. (a^{-1} -en az a-val való modulo osztást értjük.) Ha viszont $\boldsymbol{V}$ mégis az r-re kérdez rá, akkor $\boldsymbol{P}$-nek tudnia kellene $(g^r \cdot a^{-1})$ diszkrét logaritmusát kiszámolni.

Tehát, ha $\boldsymbol{P}$ a 2. lépés előtt ($\frac{1}{2}$ valószínűséggel) jól megtippeli, mit fog kérdezni $\boldsymbol{V}$, akkor át tudja verni, ha azonban rosszul tippel, biztosan lebukik.

Azt is érdemes meggondolni, hogy az eljárás valóban nem feltáró. Ugyanis $\boldsymbol{V}$ egy C értéket kap, amiből nem tud logaritmust számolni, hiszen az nehéz. Az $(x+r) \pmod{p-1}$ értékből pedig nem tud meg semmit, hiszen ha r véletlen, akkor ez a szám is véletlen (egyenletes eloszlással) $\mathbb{Z}_{p-1}$ -en.

Tehát a protokollunk egy statisztikai nem feltáró bizonyítás, ahol egy iterációban $\boldsymbol{P}$ $\frac{1}{2}$ valószínűséggel képes átverni $\boldsymbol{V}$-t, azonban kellő ismétlés után $\boldsymbol{V}$ tetszőlegesen megbizonyosodhat arról, hogy $\boldsymbol{P}$ valóban ismeri az adott érték diszkrét logaritmusát.

3.2. Blum egész felbontása

A következő eljárás nagyon speciális egészhez tartozó prímfelbontásról szól. Ebben a részben először ismertetem a Blum-egészeket, majd a rajtuk értelmezett gyökvonással fogok foglalkozni. A jó nem feltáró protokoll mellett mutatok egy tanulságos rosszat is. (Erről a témáról [2]-ben olvashatunk részletesen.)

Definíció 3.2: Egy $n \in \mathbb{N}$ számot Blum-egésznek hívunk, ha $n = pq$, ahol $p \neq q$ prímek és $p \equiv q \equiv 3 \pmod{4}$.

Tegyük fel, hogy $\boldsymbol{P}$ ismeri egy n Blum-egész felbontását, ahol a p, q prímtényezők elég nagyok. Ezt a tudását szeretné bizonyítani $\boldsymbol{V}$-nek, anélkül, hogy a prímtényezőket elárulná. Ehhez segítségül a továbbiakban ismertetjük a négyzetgyökvonás nehézségét $\mathbb{Z}_n^*$-on.

A **2.1** definícóban már ismertettük, hogy $n = pq$ esetén mit értünk négyzetelemen. Valójában a definíció Q_1-re onnan jön, hogy $a \in \mathbb{Z}_n^*$-hoz létezik-e $x \in \mathbb{Z}_n^*$, hogy $x^2 \equiv a \pmod{n}$ teljesüljön. Most egy a négyzetelemhez szeretnénk megtalálni azon x-eket, melyeknek a a négyzete modulo n. Ekkor ezen x-eket az a négyzetgyökeinek hívjuk.

Állítás: Ha $n = pq$, ahol p, q prímek és $a \in \mathbb{Z}_n^*$ (tehát relatív prím n-nel) négyzetelem, akkor a-nak négy különböző négyzetgyöke van modulo n.

Lemma: (Kínai maradéktétel következménye): $p \neq q$ prímek, $a, b \in \mathbb{Z}$. Ekkor az $x \equiv a \pmod{p}$, $x \equiv b \pmod{q}$ kongruenciarendszernek modulo pq egyértelműen van megoldása.

Bizonyítás: Ha $x^2 \equiv a \pmod{n}$ megoldható, akkor $x^2 \equiv a \pmod{p}$ és $x^2 \equiv a \pmod{q}$ is megoldható. Ezen kongruenciák gyökei legyenek sorra $(x_p, -x_p)$ és $(x_q, -x_q)$. Ekkor egy $x \in \mathbb{Z}_n^*$ négyzetgyöke a-nak, ha kongruens az egyik értékkel $(x_p, -x_p)$-ből és az egyik értékkel $(x_q, -x_q)$-ból modulo pq. Ezeket négyféleképp párosíthatjuk, és a kínai maradéktétel szerint egyértelműen létezik megoldásuk modulo n.

Állítás: Ha n Blum-egész, akkor pontosan egy négyzetgyöke négyzetelem.

Lemma: (Euler-feltétel): $p > 2$ prím és $a \not| p$, ekkor a négyzetelem modulo p akkor és csak akkor, ha $a^{\frac{p-1}{2}} \equiv 1 \pmod{p}$. Ekkor a kongruenciának két megoldása van, és ha az egyik x, akkor a másik $-x \pmod{p}$.

Ha n Blum-egész, akkor a gyökeit p, q ismeretében könnyen meg tudjuk határozni, ugyanis $p \equiv 3 \pmod{4}$ prím és a négyzetelem esetén a két négyzetgyök: $\pm a^{\frac{p+1}{4}}$. Hiszen az Euler-feltétel miatt: $(\pm a^{\frac{p+1}{4}})^2 \equiv a^{\frac{p-1}{2}} a \equiv a \pmod{p}$. Tehát n Blum-egész esetén a négy kongruenciarendszer megoldásával megkapjuk a gyököket. Azonban p, q ismerete nélkül nem ismert polinomiális idejű algoritmus arra, hogy kiszámoljuk $a \in \mathbb{Z}_n^*$ valamely négyzetgyökét.

$\boldsymbol{P}$ tehát ismeri az n Blum-egészhez tartozó p, q prímeket. Ezt a tudását $\boldsymbol{V}$-nek azzal bizonyíthatja, hogy megmutatja, képes gyorsan négyzetgyököt számolni $\mathbb{Z}_n^*$-n.

1. $\boldsymbol{V}$ generál egy véletlen r számot és kiszámolja az $x = r^2$ és $a = x^2$ értékeket.
2. $\boldsymbol{V}$ elküldi a-t $\boldsymbol{P}$-nek.
3. $\boldsymbol{P}$ kiszámolja a gyökeit, ezek: x_1, x_2, x_3, x_4. Megvizsgálja melyikük a négyzetelem, és ezt a számot visszaküldi $\boldsymbol{V}$-nek.
4. $\boldsymbol{V}$ ellenőrzi, hogy a kapott szám x-e. Ha igen, elfogad.

Amennyiben $\boldsymbol{P}$ és $\boldsymbol{V}$ követik a protokollt, ez az eljárás tökéletesen megállja a helyét, ismétlést sem igényel kifejezetten, hiszen nagyon nagy p, q prímek esetén elég valószínűtlen, hogy $\boldsymbol{P}$ képes megfelelően válaszolni ezen prímek ismeretének hiányában. Sajnos az eljárásnak van egy nagy hátránya, nem tesz eleget az

1.1.b megjegyzés második felének. Ha $\boldsymbol{V}$ nem követi a protokollt, előfordulhat, hogy ezzel az eljárással információt nyer ki $\boldsymbol{P}$-től, és megtudhatja p és q értékét. $\boldsymbol{V}$ az első lépésben nem második hatványként állítja elő x-et, hanem csak egy véletlen x-et generál, aminek a négyzetét átküldi. Ekkor amennyiben sem x, sem $-x$ nem négyzetelem, a 3. lépésben kapott válasz egy $y \neq \{x, -x\}$ lesz, így $\boldsymbol{V}$ tudni fogja a mind a négy négyzetgyökét: $x, -x, y, -y$.

Állítás: Ha n Blum-egész, ismert egy $a \in \mathbb{Z}_n^*$, $(a, n) = 1$ szám mind a négy négyzetgyöke: $x, -x, y, -y$, akkor n gyorsan faktorizálható.

Bizonyítás: $x^2 - y^2 \equiv 0 \pmod{n}$. Mivel $y \not\equiv \{x, -x\}$, ezért $x^2 - y^2 \neq 0$, és így $x^2 - y^2$ és n legnagyobb közös osztója valódi osztója n-nek. Az euklideszi algoritmussal pedig polinomiális időben faktorizálható n.

Ahhoz, hogy $\boldsymbol{V}$ ne tudhasson meg semmit, ha nem követi a protokollt, $\boldsymbol{P}$ így bizonyíthat:

1. $\boldsymbol{V}$ egy tetszőleges $a \in \mathbb{Z}_n^*$ számot küld $\boldsymbol{P}$-nek.
2. $\boldsymbol{P}$ generál random r számot és kiszámolja $y \equiv r^2 a \pmod{n}$ -et.
3. $\boldsymbol{V}$ választ egy b = {0,1} értéket, melyet elküld $\boldsymbol{P}$-nek.
4. $\boldsymbol{P}$ kiszámolja a egyik x gyökét: $x^2 \equiv a \pmod{n}$.
   Majd elküldi $z \equiv r \cdot x^b$ -t $\boldsymbol{V}$-nek.
5. $\boldsymbol{V}$ ellenőrzi, hogy fennáll-e: $y \equiv z^2 \cdot a^{1-b} \pmod{n}$. Ha igen, elfogad, különben elutasít.

$\boldsymbol{V}$ így nem tudhat meg semmi többletinformációt a ügyes megválasztásával, mert $\boldsymbol{P}$ egy random r-rel eltolja $\mathbb{Z}_n^*$ -n. Amennyiben $\boldsymbol{P}$ valóban ki tud számolni ilyen x értéket, az egyenlőség fennáll, hiszen:

$$y \equiv r^2 a \equiv r^2 x^2 \equiv (rx^b)^2 \cdot x^{(2-2b)} \equiv z^2 \cdot a \cdot a^{-b} \equiv z^2 \cdot a^{1-b} \pmod{n}$$

3.3. Hamilton kör

A gráfok témakörének egyik legjelentősebb problémája annak megállapítása, hogy létezik-e Hamilton-kör egy n csúcsú gráfon. Ez a döntési probléma NP-teljes, ha létezik polinomiális algoritmus rá, akkor P = NP. Most azzal foglalkozunk, hogy ha $\boldsymbol{P}$ ismer Hamilton-kört egy adott gráfban, azt hogyan bizonyíthatja statisztikai nem feltáró módon.

Legyen tehát G elég nagy gráf, melynek $\boldsymbol{P}$ ismeri egy Hamilton-körét. G-t generálhatja $\boldsymbol{P}$ úgy, hogy vesz egy n hosszú kört, majd a csúcsok között felvesz még valahány élet. $\boldsymbol{P}$ az ismeretét $\boldsymbol{V}$-nek bizonyíthatja anélkül, hogy $\boldsymbol{V}$ megtudná mely élek sorozata alkotja a Hamilton-kört. Az eljárás a következőképp működik:

1. $\boldsymbol{P}$ generál egy G-vel izomorf H gráfot. (Azaz átcímkézi a csúcsokat.)
2. $\boldsymbol{P}$ kulcsosládikás módszerrel bizonyítékot küld arról, hogy H rögzített és $\boldsymbol{P}$ később nem fogja megváltoztatni. Ezt úgy teszi meg, hogy H minden uv élét külön-külön valamely egyirányú függvénnyel elkódolja, és az elkódolást küldi tovább.
3. $\boldsymbol{V}$ választ, hogy $\boldsymbol{P}$ mutassa meg a G és H közti izomorfiát, vagy pedig mutassa meg a Hamilton-kört H-ban.
4. (a) Ha az izomorfiára kíváncsi $\boldsymbol{V}$, akkor $\boldsymbol{P}$ elküldi a G és H közti csúcsmegfeleltetést, valamint minden kódolva elküldött élet feltár $\boldsymbol{V}$-nek.
   (b) Ha pedig a Hamilton-kört szeretné megtudni $\boldsymbol{V}$, akkor $\boldsymbol{P}$ csak azokat az éleket fedi fel $\boldsymbol{V}$-nek, melyek alkotják azt.
5. 1-4. lépéseket néhányszor megismétlik.

A protokoll egy iterációjában $\boldsymbol{P}$ az eddigiekhez hasonlóan átverheti $\boldsymbol{V}$-t, ha előre jól tippeli meg, vajon $\boldsymbol{V}$ mire lesz kíváncsi a 3. lépésben. Attól, mert $\boldsymbol{P}$ nem ismer Hamilton-kört G-ben, még képes generálni G-vel izomorf gráfot. Illetve könnyen megteheti, hogy készít olyan H-t, melyben ismer Hamilton-kört és ugyanannyi éle van, mint G-nek. Azonban az, hogy $\boldsymbol{P}$ egyidejűleg ismer Hamilton-kört H-ban és H izomorf G-vel, természetesen ekvivalens azzal, hogy ismer Hamilton-kört G-ben. Tehát, ha $\boldsymbol{P}$ át szeretné verni $\boldsymbol{V}$-t, legfeljebb csak az egyik feltételnek tehet eleget egy iterációban.

Tehát N ismétlés után $\boldsymbol{V}$ $1 - \frac{1}{2^N}$ biztos lehet abban, hogy $\boldsymbol{P}$ nem hazudik.

3.4. Gráf háromszínezése

Egy másik fontos NP-teljes probléma a gráfok háromszínezhetőségének kérdése. Ebben a részben a bizonyító által ismert háromszínezés létezéséhez szeretnék nem feltáró bizonyítási módszert bemutatni.

Definíció 3.3: Azt mondjuk, hogy egy G gráf háromszínezhető, ha a csúcsai három partícióba oszthatóak úgy, hogy egy partíción belül két csúcs között nem fut él.

Adott egy G gráf, n darab számozott csúccsal. G-nek $\boldsymbol{P}$ ismeri egy háromszínezését. $\boldsymbol{P}$ a bizonyításához azt fogja felhasználni, hogy ha adott három szín (partíció), akkor ugyanahhoz a háromszínezéshez ezeket a színeket hatféleképpen társíthatja, így gyakorlatilag tekinthetünk úgy rá, hogy 6 különböző színezést ismer. Legyen tehát adott három szín, a protokoll a következő:

1. $\boldsymbol{P}$ az általa ismert színezéshez rendeli a három színt (választ egyet a 6 féle társítás közül).
2. A csúcsok számozása szerint növekvő sorrendben egyirányú függvénnyel elkódolva továbbítja $\boldsymbol{P}$ $\boldsymbol{V}$-nek a színezést minden csúcsra. Ezt megteheti úgy, hogy 0, 1, 2 felelnek meg egy-egy adott színnek, így egy 3-as számrendszerbeli n hosszú szám megadja a színezést. $\boldsymbol{P}$ nem az egész számot kódolja el, hanem n egymást követő üzenetben rejti el a megfelelő helyiértékek értékeit.

3. ***V*** rákérdez egy tetszőleges élre. Ez az él fusson az i-edik és j-edik csúcs között.

4. ***P*** felfedi az i-edik és j-edik csúcshoz tartozó szín értékét.

5. ***V*** ellenőrzi, hogy a két csúcs valóban különböző (0,1,2) színű-e. Ha ez nem teljesül, ***V*** tudja, hogy ***P*** hazudik, ezért elutasít. Különben elfogadja.

6. 1-5. lépéseket megismétlik néhányszor.

Fontos, hogy az ismétléseknél ***P*** mindig választhat más permutációt a színezések közül, így ***V*** akárhányszor ismételteti is meg az eljárást, nem tudhat meg semmi mást, csak hogy az általa kérdezett él csúcsai különböző színűek-e.

3.5. Gráf nemizomorfizmus

Végül ebben a fejezetben még egy módszert szeretnék ismertetni. Eddig NP-beli problémaköröket vizsgáltunk, melyekben ha rendelkezünk a problémához tartozó tanúval, akkor az polinomiális időben ellenőrizhető. Most azzal szeretnék ebben a részben foglalkozni, hogy a gráf izomorfizmus nemlétezéséhez hogyan generálható polinomiális időben ellenőrizhető bizonyíték, méghozzá nem feltáró.

A gráfizomorfiára nem ismert polinomiális idejű algoritmus, most térjünk el kissé az első fejezetben ismertetett modelltől és tegyük fel, hogy ***P*** képes gyorsan megállapítani, hogy két gráf izomorf-e egymással. ***V*** viszont továbbra is csak polinomiális időben tud ellenőrizni.

S: Két adott gráf: G_1, G_2 nem izomorfak.

1. ***V*** választ egyet a két gráf közül, és egy vele izomorf G_3 -at generál úgy, hogy átcímkézi a csúcsokat.

2. ***P*** Leellenőrzi, hogy melyik gráfból kaphatta ***V*** G_3 -at. És aszerint válaszol $\{1, 2\}$ -t, hogy G_1, vagy G_2 -vel izomorf.

3. ***V*** leellenőrzi, hogy valóban a várt választ kapja-e ***P***-től.

4. 1-3. lépéseket valahányszor megismétlik.

Ha kellően sokszor ismételve ***V*** mindig jó választ kap a 2. lépésben, eléggé biztos lehet abban, hogy ***P*** nem veri át azzal, hogy véletlenszerűen válaszol, egyúttal abban, hogy ***S*** igaz.

4. Alkalmazások különböző területeken

A következő fejezetben a nem feltáró bizonyítások gyakorlati alkalmazásairól szeretnék írni, és egyúttal bemutatni, hogy különböző területeken milyen hasznos lehet egy olyan protokoll készítése és követése, mely nem szolgáltat ki érzékeny információkat. Az itt bemutatott alkalmazások a gyakorlatban szükségszerűen körülményesebbek és terjedelmesebbek. A célom nem a megvalósítások teljes leírása, hanem a különböző területek alkalmazásainak ismertetése, illetve az alkalmazott nem feltáró bizonyítások szerepének és implementációjának leírása.

4.1. Autentikáció

A nem feltáró bizonyítások a kriptográfia különböző területein megjelennek, az egyik legfontosabb alkalmazásuk a személyazonosítás. Adott egy szerver és egy felhasználó, egyik fél sem bízik meg a másikban. A szerver nem tudja, hogy valóban a felhasználó szeretne-e bejelentkezni, míg a felhasználó nem tudja, hogy a szerver megfelelően bizalmasan kezeli-e az azonosítását. A felhasználó szeretné elkerülni, hogy ha valaki hozzáfér a szerverhez, megszerezhesse a jelszavát. Hogyan azonosítsa tehát magát a felhasználó? Erre ma különböző protokollokat használnak a szerverek. Amelyek nem feltáró bizonyításokat alkalmaznak, elég nagy biztonságot tudnak biztosítani mind a felhasználók, mind saját maguk számára. Először egy leegyszerűsített eljárást szeretnék bemutatni, majd pedig mutatok egy, a gyakorlatban használt protokollt. (Ez utóbbi részletesebben [6]-ban található.)

Az autentikáció általában jelszavas azonosítás, azonban most a felhasználó nem bízik meg a szerverben, illetve abban, hogy csak a szervernek szolgáltat ki információt. Ezért nem akarja a jelszavát elárulni. Legyen innentől $\boldsymbol{P}$: a felhasználó, $\boldsymbol{V}$: a szerver. A célunk tehát a következő: $\boldsymbol{P}$ úgy szeretné bizonyítani, hogy ő az, hogy a jelszavának ismeretéről nem feltáró bizonyítékot szolgáltat $\boldsymbol{V}$-nek, aki a bizonyítékot ellenőrizni tudja. (Az ilyen bizonyítékot "zero knowledge password proof"-nak nevezik angolul.) Mindezt interaktívan fogják elérni.

Legyen $n = pq$, melyet a szerver, azaz $\boldsymbol{V}$ generál két nagy p, q prímből és ezeket a prímeket nem tárolja el. n nyilvános, azonban p és q értékeit nem ismeri egyik felhasználó sem. Minden felhasználó jelszava egy $\mathbb{Z}_n^*$ -beli szám legyen, melynek a négyzetét publikussá teszi bárki számára. Ha tehát $\boldsymbol{P}$ jelszava x, akkor közzéteszi az $a = x^2$ számot. Ekkor $\boldsymbol{P}$ autentikációja a következő módon történhet:

1. $\boldsymbol{P}$ vesz egy véletlen $r \in \mathbb{Z}_n^*$ számot és elküldi $y \equiv r^2 \pmod{n}$ számot.
2. $\boldsymbol{V}$ választ egy $b \in \{0, 1\}$ értéket, melyet visszaküld $\boldsymbol{P}$-nek.
3. $\boldsymbol{P}$ elküldi a $z \equiv r \cdot x^b \pmod{n}$ értéket.
4. $\boldsymbol{V}$ jóváhagyja $\boldsymbol{P}$ identitását, ha 1-3. lépések néhányszori ismétlése után $z^2 \equiv y \cdot a^b$ mindig fennáll.

Mivel a publikus, az ellenőrzést $\boldsymbol{V}$ képes megtenni $b = 1$ esetén is. Ha $\boldsymbol{P}$ valóban az általa ismert x -szel számol, akkor az egyenlőség fennáll:

$$z^2 \equiv r^2 \cdot x^{2b} \equiv r^2 a^b \equiv y \cdot a^b \pmod{n}$$

A **3.2** fejezetben látott eljáráshoz hasonlóan itt sem tud meg $\boldsymbol{V}$ plusz információt, mert a véletlen r miatt z egyenletes eloszlású valószínűségi változó $\mathbb{Z}_n^*$-on. $\boldsymbol{P}$ pedig csak x ismeretében képes ilyen bizonyítékot készíteni, hiszen $b = 0$ esetén azt bizonyítja, hogy valóban ismeri az elküldött y gyökét. A $b = 1$ esetben pedig azt bizonyítja, hogy az y gyökével eltolt szám valóban az ő jelszava. Ha ez a kettő egyszerre áll fenn, az bizonyítja, hogy $\boldsymbol{P}$ ismeri x-et. Így ismétlésekkel $\boldsymbol{V}$ eléggé biztos lehet $\boldsymbol{P}$ személyazonosságában.

SRP: A következő protokoll egy eljárás szerver és felhasználó közötti kriptográfiai kulcs előállítására, mely azon alapul, hogy a felhasználó ismeri a jelszavát (PAKE protokoll). Tehát $\boldsymbol{P}$ és $\boldsymbol{V}$ egy közös titkos kulcsot fognak létrehozni $\boldsymbol{P}$ jelszavának segítségével. Az SRP (Secure Remote Password) protokoll azért jött létre, hogy bizonyos támadások ellen védje a $\boldsymbol{P}$ és $\boldsymbol{V}$ közti kommunikációt. A főbb támadási pontok, melyek ellen véd: a kommunikáció lehallgatása, "man in the middle" támadás (egy harmadik személy közvetíti a kommunikációt $\boldsymbol{P}$ és $\boldsymbol{V}$ között), a jelszó szótár alapú megtippelése, illetve a szerverhez való hozzáférés. Az SRP-6a verzióját fogom nagyvonalakban ismertetni, ehhez szükség van néhány jelölésre.

Jelölés: Legyen $N = 2p + 1$ alakú nagy prím, ahol p prím és g generátora $\mathbb{Z}_N$-nek. H legyen (megfelelően biztonságos) egyirányú hash függvény néhány paraméterre. (H meghívható egy vagy több paraméterrel is.)

A szerver a jelszavakat nem tárolja az adatbázisában, helyette minden felhasználóhoz egy (x, v) értékpárt tárol. Ha a felhasználó jelszava $pass$, akkor $x = H(s, pass)$, ahol s egy véletlen érték, melyet csak a felhasználó ismer, és $v = g^x$. (A továbbiakban ismertetett protokollt megelőzi egy beregisztrálási folyamat, melyben x értékét közli a felhasználó a szerverrel.) Tegyük fel, hogy $\boldsymbol{P}$ egy beregisztrált felhasználó (tehát már tartozik hozzá (x, v)). Az eljárás így néz ki:

1. $\boldsymbol{P}$ generál egy véletlen a-t és elküldi $A = g^a$ -t $\boldsymbol{V}$-nek.
2. $\boldsymbol{V}$ véletlen b értéket generál és elküldi $B = v \cdot g^b$ -t.
3. $\boldsymbol{P}$ és $\boldsymbol{V}$ is kiszámolja $u = H(A, B)$ -t.
4. Mindketten kiszámolnak egy közös kulcs értéket:
   (a) $\boldsymbol{P}$ számolja: $x = H(s, pass)$ és $S = (B \cdot g^{-x})^{a+ux}$
   (b) $\boldsymbol{V}$ számolja: $S = (Av^u)^b$

A $K = H(S)$ érték lesz a közös kulcsuk.

Ekkor $\boldsymbol{P}$ és $\boldsymbol{V}$ ugyanazt a kulcsot kapják, ugyanis:

$$(B \cdot g^{-x})^{a+ux} = (v \cdot g^{b-x})^{a+ux} = (g^x \cdot g^{b-x})^{a+ux} = (g^b)^{a+ux} \tag{1}$$

$$(Av^u)^b = (g^a v^u)^b = (g^a g^{xu})^b = (g^{a+xu})^b = (g^b)^{a+ux} \tag{2}$$

Ekkor tehát $\boldsymbol{P}$ és $\boldsymbol{V}$ is rendelkezik egy közös K kulccsal. Ezt a K értéket minden belépéskor újragenerálják. Ezt a kulcsot munkamenetkulcsnak (session key-nek) nevezzük. Már csak egymásnak kell bizonyítsák, hogy ugyanazt a kulcsot kapták. Jelöljük S_P, S_V -vel rendre a $\boldsymbol{P}$ és $\boldsymbol{V}$ által kiszámolt S értéket. Ezt bizonyíthatják egymásnak a következő üzenetekkel:

1. $\boldsymbol{P}$ elküldi $\boldsymbol{V}$-nek $M_1 = H(A, B, S_P)$ -t.
2. $\boldsymbol{V}$ ellenőrzi, hogy H valóban M_1 -et adja $H(A, B, S_V)$ -re. Ha igen, visszaküldi $M_2 = H(A, M_1, S_V)$ -t. Ha nem, elutasít (és nem is küld vissza bizonyítékot).
3. $\boldsymbol{P}$ leellenőrzi, hogy $M_2 = H(A, M_1, S_P)$. Ha igen, tudja, hogy valóban a szerverrel kommunikál.

Ez a személyazonosítási eljárás nem feltáró bizonyítás, ugyanis, ha $\boldsymbol{P}$ ismeri a jelszavát, akkor $\boldsymbol{V}$ erről meggyőződik. Ha $\boldsymbol{P}$ nem ismeri a jelszavát, akkor S értékét nem tudja kiszámolni, ennek hiányában jól megválasztott H mellett M_1 -et sem képes kiszámolni, tehát $\boldsymbol{V}$ meggyőződik róla, hogy $\boldsymbol{P}$ nem ismeri a jelszót. Valamint az eljárás során sem $\boldsymbol{V}$, sem egy harmadik fél nem tudhatja meg $\boldsymbol{P}$ jelszavát. A protokoll során valójában $\boldsymbol{V}$ is bizonyítja $\boldsymbol{P}$-nek, hogy ő a hiteles szerver. A $\boldsymbol{P}$ = szerver, $\boldsymbol{V}$ = felhasználó szerepcserével is teljesülnek a nem feltáró bizonyítás definíciójának feltételei.

A protokoll legfőbb érdemei, hogy a korábban említett, jól ismert támadások ellen biztonságos. Az SRP egy titkos kulcscsere protokollt foglal magában, ezzel védekezve a kommunikáció lehallgatása ellen, a szótár alapú tippelések (brute-froce dictionary attacks) ellen védekezhet a szerver, mivel a bizonyítás interaktív, így sokszori rossz próbálkozás esetén letilthat a szerver. Ha pedig valaki hozzáfér az adatbázishoz, nem tud meg semmit a felhasználók jelszaváról, és nem is tud belépni az ő nevükben.

4.2. Tranzakciók jóváhagyása blokkláncon

A kriptovaluták eredete a 20. század végére vezethető vissza, azonban az utóbbi évtizedben lettek egyre népszerűbbek. 2009-ben megjelent a Bitcoin, mely az első decentralizált elektronikus pénznem, ezt követően számos hasonló valuta látott napvilágot. A Zcash egy a Bitcoinból eredő kriptovaluta, mely a felhasználói számára teljesen privát tranzakció lehetőségét biztosítja. Mindezt a nem feltáró bizonyítások megfelelő alkalmazásával érik el, pontosabban zkSNARK-okkal, melyek számításbeli nem feltáró bizonyítások. Ebben a részben a Zcash implementációjáról és a zkSNARK-okról szeretnék írni. Mivel az implementáció és a mögötte rejlő absztrakció önmagában is elég nagy terjedelmű anyag, ezért célom, hogy a nem feltáró bizonyítások szempontjából is fontos tranzakció validációt emeljem ki. Ennek megvalósítását pedig nagyvonalakban szeretném

bemutatni. (Néhány helyen a leegyszerűsítés miatt a valódi implementációtól kissé el fogok térni.) Mielőtt a Zcashről írnék, néhány fogalmat szükséges ismertetni:

• **Kriptovalutának** nevezünk olyan elektronikus pénznemet, amely előállításának szabályozásához, illetve az átutalások ellenőrzéséhez kriptográfiai eszközöket használnak. Jellemzően nem központosított fizetőeszközökről van szó (decentralizáltak), nem tartozik hozzájuk központi adatbázis.

• A valuta használói ellenőrzik és hagyják jóvá az egyes tranzakciókat, ennek eléréséhez **blokklánc** technológiát használnak. A blokklánc elkódolt adatok egyre bővülő "listája", melyben minden elem a közvetlen előtte lévő blokk hash-ét tartalmazza. Ennek a célja, hogy ha bármely korábbi tranzakciót utólag próbálná valaki módosítani, akkor azzal az összes azt követő adat módosulna. A blokklánc általában publikus és a felhasználók hagyják jóvá. A hatékonyság miatt a tranzakciókat minden egyes blokkon egy Merkle fában szokták tárolni.

• A **Merkle fa** egy olyan fa adatszerkezet, mely kriptográfiai hash függvényekkel kódolja el a leveleket. Továbbá minden szülő a gyerekeinek a hash függvénye. Az előnye a Merkle fáknak, hogy nagy adathalmaz esetén egy elem tartalmazását igen gyorsan lehet ellenőrizni benne adott úgynevezett Merkle bizonyíték mellett, amely nagyvonalakban egy útvonal az elem hash-ét tartalmazó levéltől a gyökérig.

A Bitcoin pseudoanonim, minden felhasználóhoz egy publikus cím kötődik. A blokklánc alapján visszafejteni egy címhez tartozó személy személyazonosságát nem lehet. Azonban bizonyos dolgok lekövethetőek, például egy címhez tartozó tranzakcióforgalom. Ennek kiküszöbölésére született a Zcash, melynek célja az volt, hogy olyan átutalásokra biztosítson lehetőséget, melyben a feladó és a küldő címe is el van kódolva, illetve a tranzakció pontos összege is. Tehát egy felhasználó csak annyit láthat, hogy valaki valakinek valamennyit utalt; azon kívül, hogy egy tranzakció megtörtént, nem tudhat meg semmit. Ez önmagában nem jelent problémát, azonban biztosítani szükséges azt is, hogy a blokklánc jóváhagyható legyen.

Zcash: A Zcash-ben vannak transzparens és védett címek. A transzparens címek közötti tranzakciók úgy működnek, mint a Bitcoin esetében, számunkra a védett címek közötti átutalások lesznek most fontosak. A Zcash-ben a tranzakciókat úgynevezett **note**-ok teszik lehetővé. Egy note-hoz tartozik valahány egységnyi valuta (ZEC), egy szériaszám és egy publikus kulcs. NOTE(v, s, k) jelölje, hogy v összeg van benne, s szériaszámot tartalmaz és k a publikus kulcs. Minden note-nak van egy tulajdonosa, aki rendelkezik a k-hoz tartozó privát kulccsal, mely lehetővé teszi, hogy elutalja a rajta levő összeget. Amikor egy felhasználónak utalnak valamekkora összeget, létrejön egy új note (commitment), amihez csak ő fogja ismerni a privát kulcsot. A valaha létrejött note-ok egy Merkle fában tárolódnak, ezen kívül pedig van egy érvénytelenítési halmaz, amibe azoknak a szériaszámoknak a hash-e kerül, amelyek már elköltött note-hoz tartoznak. Ezeket a hash függvénnyel elkódolt számokat **nullifiereknek** nevezik.

Legyen ***A*** és ***B*** két személy, ***A*** szeretne védett utalást intézni ***B***-nek, és az egyszerűség kedvéért tegyük fel, hogy ***A*** NOTE(v, s_1, k_A)-ján levő összeg pontosan lefedi az átutalni kívánt összeget. Ekkor a tranzakció így néz ki:

- ***A*** a privát kulcsával hozzáfér a note-hoz, így imeri a note-beli v, s_1 értékeket.
- ***A*** új nullifiert hoz létre: hash(s_1), ezt beteszi az érvénytelenítési halmazba.
- ***A*** létrehoz egy új note-ot v értékkel, melyet ***B*** publikus k_B kulcsával kódol el (így biztosítva, hogy azt ***B*** tudja majd elkölteni), majd ezt hozzáadja a Merkle fához.
- ***A*** közzéteszi a Merkle fa és az érvénytelenítési halmaz tranzakció utáni állapotát és egy nem feltáró bizonyítékot, mely azt bizonyítja, hogy a tranzakció helyes. Ha a felhasználók elfogadják a bizonyítékot, akkor elfogadják a tranzakciót, egyúttal a megváltoztatott állapotát a Merkle fának és az érvénytelenítési halmaznak.

Ennek a megvalósításnak az előnye, hogy bár minden elköltött note-hoz tartozik egy nullifier, csupán a két módosított adatszerkezetből nem tudják összekapcsolni a többiek, hogy melyik nullifier melyik note-hoz tartozik. Ahhoz, hogy a többi felhasználó jóvá tudja hagyni ezt a tranzakciót, az szükséges, hogy megbizonyosodjanak róla, hogy bárki is hajtotta végre:

1. Az elköltött note létezik (benne van a Merkle fában).
2. Ehhez a note-hoz tartozó nullifierrel bővítette az érvénytelenítési halmazt.
3. A nullifier nem szerepelt korábban az érvénytelenítési halmazban (azaz a note még nem volt elköltve).
4. Az újonnan létrejövő note-hoz nem tartozik nagyobb összeg, mint az érvénytelenítetthez.

A 3. feltételt ellenőrizni tudja mindenki az érvénytelenítési halmaz tranzakció előtti és utáni állapotát összehasonlítva. A többi feltételhez azonban létezik egy f ellenőrzőfüggvény, melyre a következő igaz: $f(m_1, m_2, n_A, n_B, v, p_A, p_B)$ pontosan akkor 1, ha m_1, m_2 a Merkle fa tranzakció előtti és utáni állapotának gyökerei, n_A, n_B az érvénytelenített és a létrehozott note-ok (ebben a sorrendben), v a tranzakció összege, p_A Merkle bizonyíték, hogy n_A note-on legalább v összeg van m_1-ben, p_B Merkle bizonyíték, hogy m_1-ből m_2-t kapjuk a tranzakcióval és a tranzakcióra teljesülnek a fent említett feltételek. Azt, hogy pontosan hogyan működik f, most nem szükséges látnunk. ***A*** bizonyítani szeretné, hogy ismer olyan n_A, n_B, v, p_A, p_B értékeket, amely a Merkle fát a tranzakcióval egyik állapotból a másikba változtatja, anélkül, hogy ezeket az értékeket elárulná, vagyis, hogy f ezen paraméterek mellett 1-et ad. Mindezt megteheti a zkSNARK-ok segítségével.

zkSNARK: Ez egy mozaikszó, az angol "zero knowledge Succint Non interactive ARgument of Knowledge" fogalom rövidítése. Annyit jelent, hogy a tudás olyan számításbeli nem interaktív nem feltáró bizonyítása, amely nagyon gyor-

san ellenőrizhető. A bizonyíték elkészítése lehet, hogy hosszabb ideig tart, de jóváhagyni nagyon gyorsan (ezred másodperc nagyságrendű idő alatt) lehetséges.

A Zcash az előbb említett paraméterek ismeretének bizonyításához készített zkSNARK-ot a következő módon: f valamilyen aritmetikai műveletek sorozatából áll, melyből először egy aritmetikai hálózatot hoztak létre (hasonló a Boole hálózathoz, csak $\vee, \wedge$ műveletek helyett összeadások és szorzások vannak benne), majd abból egy kvadratikus aritmetikai programot (röviden QAP). Egy QAP alatt azt értjük, hogy adottak meghatározott polinomok $p_1, p_2, ..., p_n$ és egy t célpolinom. Azt mondjuk, hogy a QAP-t kielégítik a $\lambda_1, ..., \lambda_n$ együtthatók, ha t osztja az ezekel az együtthatókkal vett lineáris kombinációját a többi polinomnak. Egy QAP kielégíthetőségének eldöntése NP-beli. A Zcash esetében, ha $\boldsymbol{A}$ ismeri f paramétereit, akkor abból tud kielégíthetőséget egy QAP-hez, és ezt bizonyítja végül zkSNARK-kal. (Feltesszük, hogy a polinomok legfeljebb d-edfokúak.)

zkSNARK QAP-re: Adottak $p_1, p_2, ..., p_n$ polinomok és t célpolinom. Legyen $P(x) = \sum_{i=1}^{n} \lambda_i p_i(x)$. $\boldsymbol{A}$ a $P(x) = h(x)t(x)$ egyenlőséget szeretné belátni ahhoz, hogy megmutassa, $(\lambda_1, ..., \lambda_n)$ kielégíti a QAP-t, de P-t (vagyis a λ_i értékeket) azonban nem szeretné nyilvánosságra hozni.
A pontos protokoll sokkal hosszabb kifejtést igényelne mint amennyi a szakdolgozatom keretébe belefér, azonban nagyon jó és részletes leírás van erről a Zcash hivatalos honlapján [7] és [8]-ban, a teljes protokoll pedig [9]-ben található. Most csak az eljárás néhány fontosabb elemét emelem ki.

A Zcash-ben ezek a polinomok magas fokszámúak, emiatt nagyon lassú a valódi polinomokkal számolni. Ezért az ellenőrzéshez véletlen kiértékelést használnak, ami azt jelenti, hogy egy véletlen s pontban nézik meg, hogy fennáll-e az egyenlőség. Így sokkal gyorsabb a kiértékelés, ez a megközelítés viszont a bizonyosság rovására megy, hiszen lehet, hogy az ellenőrzött s pontban fennáll $P(s) = h(s)t(s)$, de nem mindenhol.

Az egyenlőség két oldalán legfeljebb d-ed fokú polinomok állnak, ha különbözőek, legfeljebb d pontban lehetnek egyenlőek. Mivel s véletlen pont, kicsi a valószínűsége, hogy ha nem minden x-re áll fenn az egyenlőség, akkor s-ben mégis teljesüljön. Azonban, ha $\boldsymbol{A}$ ismeri s-t, akkor esetleg készíthet olyan polinomot, mely s-ben éppen kielégíti az egyenlőséget, így elfogadtathatna érvénytelen tranzakciót is.

Ennek kiküszöbölésére a Zcash-ben homomorfikus elrejtést használnak: E homomorfikus elrejtés, ha $E(ax+by) = aE(x)+bE(y)$ minden a, b, x, y-ra. Vagyis, ha $\boldsymbol{A}$ nem az s pontot ismeri, hanem az $E(1)$, $E(s)$, $E(s^2)$, ..., $E(s^d)$ értékeket, akkor minden legfeljebb d-edfokú p polinomra ki tudja számolni $E(p(s))$-t. Ha $\boldsymbol{A}$ a bizonyításban P-t ily módon kiértékeli, tehát $E(P(s))$-t számol, akkor $\boldsymbol{A}$ nem tudja meg, mely s-re értékelte ki a polinomot, és a bizonyítékából nem tudja meg senki, hogy mi volt P.

Megjegyzés 4.1: A valódi protokollban $\boldsymbol{A}$-nak még bizonyítania kell, hogy valóban $E(P(s))$-t számolta ki, nem pedig valami más értéket.

4.3. Nukleáris rakétafejek

A nem feltáró bizonyításoknak elsősorban kriptográfiai alkalmazásai vannak. Ebben a részben viszont egy másik területen elért eredményét szeretném bemutatni. Alex Glaser, Boaz Barak és Rob Goldston egy fizikai alkalmazását fejlesztették ki a nem feltáró bizonyításoknak, mellyel két lényegében egyforma objektumot képesek összehasonlítani. A protokollt nukleáris rakétafejek azonosítására találták ki, ezen a területen bizonyos adatok kifejezetten bizalmasak, ezért a nem feltáró jellege igazán hasznos a gyakorlatban. Ezt az eljárást szeretném nagyvonalakban ismertetni. (Bővebben [10] és [11]-ben lehet erről olvasni.)

Az eljárás bemutatása előtt azonban szükséges betekinteni a nukleáris fegyverzet-ellenőrzés ide kapcsolódó részébe. A nukleáris fegyverek leszerelésére több, egyre komplexebb egyezmény született az Amerikai Egyesült Államok és Oroszország között. Nagy problémát jelent, hogy zárt tartályokban őrzött rakétafejeket, hasadó anyagot, illetve különböző nukleáris fegyver alkatrészeket be tudjanak azonosítani. Egy nukleáris rakétafej beazonosításán azt értjük, hogy egy ellenőr (a másik fél) jóváhagyja, hogy a tartályban lévő objektum bizonyos tulajdonságai megfelelnek az elvárt paramétereknek. Azonban ezek a tulajdonságok igen bizalmas adatok, melyeket nem szeretnének egymásnak kiszolgáltatni. Emiatt a feladat kissé ellentmondásos, hiszen az ellenőrnek anélkül kell jóváhagynia a fegyvert, hogy tudná ezeket a paramétereket. Ma két módszert használnak egymással párhuzamosan, ugyanis mind a kettőnek megvan a maga előnye és hátránya. Az egyik módszer azon alapul, hogy egy olyan rendszert hoznak létre, amely képes megmérni a szükséges tulajdonságokat, azonban ezeket a bizalmas adatokat nem közli, csupán a tényt, hogy az adatok megfelelnek-e a paramétereknek. A másik módszer pedig a sablonhoz való társítás. Ez utóbbit úgy valósítják meg, hogy vesznek egy hitelesnek vélt sablon rakétafejet (pl. egy korábban a másik módszerrel beazonosítottat) és azt ellenőrzi az ellenőr, hogy ugyanolyanok-e. Ezt úgy szeretnénk, ha el tudná érni, hogy eközben ne tudjon meg semmit se a sablonról, se a vizsgált fejről. Ehhez a megközelítéshez készült a nem feltáró bizonyítás, amelyet ismertetni szeretnék.

Az eljárás célját általánosabban is megfogalmazhatjuk. Azt szeretnénk, hogy $\boldsymbol{P}$ n darab objektum egyformaságát tudja $\boldsymbol{V}$-nek bizonyítani nem feltáró módon. Természetesen a való életben tökéletesen nem lesznek teljesen ugyanolyanok az objektumok, így $\boldsymbol{P}$ és $\boldsymbol{V}$ között kell előzetesen egy megállapodás, hogy milyen kísérleti pontossággal számoljanak. A protokoll fizikai megvalósításához az összehasonlítandó objektumokat egy előre meghatározott szögből bombázzák neutronokkal, a hitelesítendő objektum mögé pedig előretöltött buborékdetektorokat helyeznek. Ezután pedig azt vizsgálja $\boldsymbol{V}$, hogy ezeken a detektorokon milyen hatást váltott ki a sugárzás.

A buborékdetektorokról: Egy objektum vizsgálata úgy néz ki, hogy ezt az objektumot egy oldalról neutronokkal bombázzák, a másik oldalon pedig a neturonok szóródását vizsgálják az úgynevezett buborékdetektorokkal. Ezek a detektorok ampullákba töltött folyadékok, amikben a neturonok hatására makroszkopikus buborékok alakulnak ki.

A detektorok úgy vannak elhelyezve a másik oldalon, hogy ha előtöltetlen (alap) állapotban vannak, akkor az objektumról egy egyedi képet alkotnak a buborékok. Az objektum mögé helyezett buborékdetektorokat detektorhalmaznak fogom hívni.

Ha a detektorok tulajdonosa tudja, hogy milyen objektumot fognak velük vizsgálni, akkor az arról alkotott kép "komplementer képével" előtölthetőek a detektorok. Ez alatt azt értjük, hogy ha előtöltetlen detektorokat sugárzunk úgy, hogy nincs semmilyen objektum előttük, ugyanazt kapjuk, mint ha az objektum komplementer képével előtöltötteket sugároznánk úgy, hogy az objektum előttük van. Erre a tulajdonságra alapozva tudjuk elkészíteni a nem feltáró bizonyítást:

1. ***P*** előállít valahány (n-nél több) detektorhalmazt.
2. ***V*** választ közülük n-et, melyet használni fognak a bizonyítás során, a többit pedig ***P*** ***V*** rendelkezésére bocsátja, hogy ellenőrizhesse őket.
3. Megállapodnak, hogy az objektumokat milyen szögből és milyen energiával fogják vizsgálni.
4. ***P*** ennek megfelelően előtölti a detektorokat a vizsgálandó objektumok komplementerével.
5. Ezután ***V*** összepárosítja a vizsgálandó objektumokat véletlenszerűen a detektorhalmazokkal. Ezen párosítás mellett elvégzik a neutronsugárzást.
6. ***V*** jóváhagyja, hogy az objektumok egyformák, ha minden detektorhalmazon végül a detektorok úgy töltődnek, ahogy elvárnák, ha objektum nélkül sugároznák az előtöltetlen detektorokat. (Valójában néhány buborék keletkezni fog. Az elvárás az, hogy a buborékdetektorokon létrejött makroszkopikus buborékok független Poisson eloszlásúak legyenek minden detektorhalmazon.)

Az eljárás fizikai nem feltáró bizonyítás, ez azt jelenti, hogy a protokoll során valamilyen fizikai mérést végzünk, mely rendelkezhet valamilyen mértékű hibával. Ekkor a nem feltáró bizonyítás definíciója teljesül, ha ez a hiba nem nagyobb, mint amit az eljárás során elvárunk. (Fontos, hogy ha az elvárt hiba mértéke nagyobb, akkor egy iterációban kevésbé meggyőző ***V*** számára a mérés eredménye.) Ebben a protokollban, ha ***P*** valóban nukleáris rakétafejet szeretne hitelesíteni és a mérési hiba nem túl nagy, akkor ***V*** meggyőződik erről az eljárás végére. Ha ***P*** a hiteles rakétafejtől különböző objektumokat szeretne hitelesíteni, akkor legjobb esetben is $\frac{1}{n}$ valószínűséggel képes átverni ***V***-t. Az eljárás során ***V*** nem tud meg semmit az objektumokról, ha az objektumok egyformák (nem nézi meg őket, de feltesszük, hogy abban biztos, hogy az egyik hiteles). Egy külső szemlélő sem tudhat meg semmit, ugyanis ha ***V*** és ***P*** összebeszélnek, akkor társíthatnak nem egyforma objektumokat a saját komplementerükhöz.

4.4. Szavazási modell

Végül a nem feltáró bizonyításoknak egy szavazási sémára készített alkalmazását szeretném bemutatni, melyet Josh D. Cohen és Michael J. Fischer publikált

1985-ben ([12]). A szavazási modell célja az volt, hogy a szavazók kriptográfiai eljárással tudják a szavazatukat meghozni úgy, hogy minden kommunikáció nyilvános legyen, a szavazatok összeszámlálhatóak legyenek, azonban egyikük se tudja visszafejteni, hogy mások mit szavaztak. Az eljárás során egy titkosított igen-nem "szavazócédulát" készít minden szavazó, aminek segítségével titkosan igent vagy nemet szavaz. A szavazatokat pedig egy kitüntetett személy, a kormány számolja össze és hirdeti ki. (Nem elvárás, hogy a kormány ne tudja meg, ki mire szavazott, csak hogy a többi szavazó ne tudja.) Az eljárás jelentős eredménye, hogy kiszűri a csaló szavazókat (akik nem követik a protokollt), emellett pedig, ha a kormány - akár más csaló szavazókkal összebeszélve - hamis eredményt hirdet ki, azt a többi szavazó szinte biztosan észreveszi. Ebben a részben csak az eljárás menetét szeretném bemutatni. Továbbá a példa célja az lenne, hogy megmutassa, hogyan készíthető nem feltáró bizonyítás nem interaktív módon, hogy hogyan képes egy jeladó kiváltani az interakció szerepét. Ráadásul ez az alkalmazás egy nagyon jó példa a többszemélyes változatra, ugyanis minden résztvevő bizonyítékot fog szolgáltatni a saját döntéséről a többieknek.

Tekintsük a következő modellt: adott J darab szavazó és egy kitüntetett személy (a kormány), aki a szavazatokat össze fogja számolni. A szavazókat jelöljük $\boldsymbol{V_1}$, $\boldsymbol{V_2}$, ..., $\boldsymbol{V_J}$ -vel, a kormányt pedig $\boldsymbol{G}$-vel. A szavazók nem bíznak sem egymásban, sem $\boldsymbol{G}$-ben. Adott továbbá egy globális óra, és egy jeladó, melyeket jóváhagytak a szavazásban résztvevők. $\boldsymbol{V_1}$, ..., $\boldsymbol{V_J}$, $\boldsymbol{G}$ rendelkeznek egy saját publikus táblával, melyet bárki olvashat, de mindenki csak a saját táblájára írhat. Minden táblára való kommunikációt ellát a globális óra egy időbélyeggel, melynek a sémában annyi szerepe van, hogy a szavazás fázisai egy időpontban lezárhatóak legyenek, és csak az addig kiírt üzenetek számítsanak. A jeladó pedig a következő módon működik: ha bárki a tábláján kér egy üzenetet a jeladótól, akkor válaszul kap egy $\{0,1\}$ bitet, melyet a jeladó aláír, így mindenki hitelesíteni tudja, hogy a bit a jeladótól érkezett.

A szavazási sémához egy speciális $\mathbb{Z}_n$ csoporton fog minden szavazó elkészíteni egy igen-nem szavazócédulát, melyet a többiek képesek jóváhagyni. Legyen $r > J$ prím, valamint p, q nagy prímek, hogy $r|(p-1)$ és $r \nmid (q-1)$ és $n = pq$. Az ilyen n értékre azt mondjuk, hogy megengedhető r-re.

Definíció 4.1: Ha adott n, r a fenti módon és adott n-hez relatív prím y, akkor i egész szám esetén i-szavazatnak hívunk egy $w \in \mathbb{Z}_n$ -t, ha a következők teljesülnek:

- $(w, n) = 1$
- $0 \leq w < r$
- Valamely x -re $w = y^i x^r \pmod{n}$

Állítás: Tegyük fel, hogy nem létezik x, melyre $y = x^r$. Ekkor egy $w \in \mathbb{Z}_n$ pontosan egy $0 \leq i < r$-re lesz i-szavazat.

Most igen-nem szavazatokat szeretnénk összeszámolni, ezért minden szavazó készít egy szavazócédulát: a 0-szavazatok fognak megfelelni a nem szavazatnak, az 1-szavazatok az igennek. Azt mondjuk hogy egy (w_1, w_2) értékpár igen-nem szavazat (vagy érvényes szavazócédula), ha $w_1 = 0, w_2 = 1$, vagy $w_1 = 1, w_2 = 0$.

Állítás: Ha $w_1, ..., w_k$ mindegyike egy igen-nem szavazat értékpár egyike és $k < r$, akkor $W = \prod_{i=1}^{k} w_i$ egy t-szavazat, ahol t az 1-szavazatok száma $w_1, ..., w_k$ között.

A fenti állításokat nem triviális belátni, azonban szükségesek a protokoll helyességéhez. Az eljárás négy részre oszlik fel, minden részben a résztvevők elvégezhetik az összes lépést egymástól függetlenül, azonban a következő rész előtt meg kell várják egymást. A protokoll a következő:

1. Ezt a részt $\boldsymbol{G}$ hajtja végre.
   (a) Választ adott számú (az egyszerűség kedvéért legyen 2^N darab) $r_i > J$ értéket, majd kiír a táblájára véletlenül választott (n_i, y_i) párokat, hogy n_i megengedhető r_i -re és y_i relatív prím n_i -vel.
   (b) A jeladótól N darab random bitet kér, melyek sorozata meghatároz egy kettesszámrendszerbeli m számot.
   (c) Minden $i \neq m$ -re kiírja, hogy n_i mely p_i és q_i szorzata, valamint r_i-t. Innentől $n = n_m, y = y_m$ és mindent modulo n számolnak.
2. Ezt a részt minden $\boldsymbol{V_j}$ végrehajtja.
   (a) Választanak adott számú f_i, g_i értékeket. $(i = 0..k)$ Szavazócédulákat gyártanak úgy, hogy minden i-re az értékpár $f_i^r \pmod n$ és $yg_i^r \pmod n$ valamilyen sorrendben.
   (b) A jeladótól eggyel kevesebb bitet kérnek, mint ahány szavazócédulát gyártottak. b_i az i-edik bit. $(i = 1..k)$
   (c) Ezután a legelső cédulájukról nem árulnak el semmit. A többiről pedig, ha $b_i = 0$, akkor kiírják az f_i, g_i -t, ha $b_i = 1$, akkor az $f_i \cdot f_0^{-1}, g_i \cdot g_0^{-1}$ értékeket.
3. Ezt a részt minden $\boldsymbol{V_j}$ végrehajtja.
   (a) Leadják a szavazatukat. Ha igennel szeretnének szavazni, akkor yg_0^r -et, ha pedig nemmel, akkor f_0^r -et írnak ki.
4. Ezt a részt $\boldsymbol{G}$ hajtja végre.
   (a) Megszámolja az érvényes 0-1 szavazatok számát. K legyen azon szavazatok száma, melyek érvényes szavazócéduláról származnak. Adott számú n-hez relatív prím c_i értéket generál és közzéteszi $C_i = c_i^r$ értékeket.
   (b) A jeladótól annyi bitet kér, ahány értéket generált.
   (c) Kiszámolja az x, t értékeket, melyre $W = y^t x^r$. Kiírja a szavazás végeredményét: $(t, K - t)$. Minden i-re, ha az i-edik bit, amit a jeladó adott 1, akkor c_i -t írja ki, ha pedig 0, akkor pedig $c_i x$ -et.

A nem feltáró bizonyítás alkalmazása a protokoll 1,2,4 részeiben jelenik meg. Minden részben az (a) lépésben meghozott döntésükről szolgáltatnak bizonyítékot a résztvevők a (c) lépésben. Itt a (b) lépésekben mindenki által megbízhatónak vélt jeladó szolgáltatja a bizonyítótól független inputot, melyet eddig az ellenőr biztosított. Nézzük meg mindhárom nemfeltáró részt (1, 2, 4) a protokollban:

1. rész: Ebben a részben $\boldsymbol{G}$ azt bizonyítja, hogy mindegyik (n_i, y_i) értékpárra n megengedhető r_i-re és y_i relatív prím n_i-vel. Minden $i \neq m$ -re elárulja a prímtényezőket, így a szavazók ellenőrizni tudják. $\boldsymbol{G}$ nem tudja előre m értékét, ezért csak $\frac{1}{2^N}$ valószínűséggel tud olyan értékpárt generálni, amelyre ez nem teljesül és a szavazók nem veszik észre.

2. rész: Ebben a részben minden $\boldsymbol{V_j}$ bizonyítja mindenki másnak, hogy a legelső értékpárjuk 0-1 szavazat. Azon értékpárokra, ahol f_i, g_i -t fedik fel, azt bizonyítják, hogy azok az értékpárok 0-1 szavazatok. Ahol pedig $f_i \cdot f_0^{-1}, g_i \cdot g_0^{-1}$ -t fedik fel, ott azt bizonyítják, hogy ezek az értékpárok ugyanolyan típusúak, mint az első. (Azaz, ha az első 0-1 szavazat, akkor ezek is.) Ez utóbbi azért igaz, mert az első értékpárral összehasonlítva kiszámolható $(f_i f_0^{-1})^r$ és $(y g_i (y g_0)^{-1})^r$, így ellenőrizhető, hogy valóban az r-edik gyöküket tették közzé ezen értékeknek.

4. rész: Ebben a részben pedig $\boldsymbol{G}$ bizonyítja, hogy a valós végeredményét közli a szavazásnak. Ugyanis a szavazók ellenőrizhetik, hogy C_i valóban egy véletlen c_i r-edik hatványa, vagy pedig, hogy $C_i K = y^t (c_i x)^r$. Ezzel ellenőrizve, hogy valóban t-szavazat a K.

Mind a három nem feltáró bizonyíték statisztikai, mert az első részben $\frac{1}{2^N}$, a 2. és 4. részben pedig minden i-re $\frac{1}{2}$ valószínűséggel készíthet a protokolltól eltérve olyan értéket, mely nem buktatja le, hogy (a) lépésben nem követte a protokollt. Láthatjuk tehát, hogy sok esetben, ha adott egy interaktív nem feltáró bizonyítás $\boldsymbol{P}$ és $\boldsymbol{V}$ között és van egy jeladó, melyben megbíznak, akkor a bizonyítás átalakítható nem interaktívvá úgy, hogy $\boldsymbol{V}$ interakcóját a jeladó helyettesíti.

Felhasznált irodalom

1. Lovász László (1992) Algoritmusok bonyolultsága. Egyetemi jegyzet, Typotex (2014)

2. Wettl Ferenc: Varázslók titkai - a nem feltáró bizonyítás, Új matematikai mozaik, Typotex (2002)

3. Quisquater JJ. et al. (1990) How to Explain Zero-Knowledge Protocols to Your Children. In: Brassard G. (eds) Advances in Cryptology - CRYPTO' 89 Proceedings. CRYPTO 1989. Lecture Notes in Computer Science, vol 435. Springer, New York, NY

4. KöMaL cikk (1985) Kártya telefonon, avagy tanuljunk nyelveket! http://db.komal.hu/scan/1985/04/98504145.g4.png, ill. http://db.komal.hu/scan/1985/04/98504146.g4.png

5. Zero-Knowledge Proofs Explained: Non-Interactive Zero-Knowledge Proofs. Zero-Knowledge Proofs Explained Part 2: Non-Interactive Zero-Knowledge Proofs, 7 Dec. 2017, https://www.expressvpn.com/blog/zero-knowledge-proofs-explained-non-interactive-zero-knowledge-proofs/

6. SRP Protocol Design, http://srp.stanford.edu/design.html

7. Zcash original website - https://z.cash/technology/zksnarks/

8. Christian Reitwießner - zkSNARKs in a Nutshell https://chriseth.github.io/notes/articles/zksnarks/zksnarks.pdf

9. Diara Hopwood, Sean Bowe, Taylor Hornby, Nathan Wilcox - Zcash Protocol Specification (Version 2019.0.8) https://github.com/zcash/zips/blob/master/protocol/protocol.pdf (Last commit at 29 Nov. 2019)

10. Philippe, S., Goldston, R., Glaser, A. et al. A physical zero-knowledge object-comparison system for nuclear warhead verification. Nat Commun 7, 12890 (2016) doi:10.1038/ncomms12890

11. Alex Glaser, Boaz Barak, Rob Goldston - A New Approach to Nuclear Warhead Verification Using a Zero-Knowledge Protocol http://www.princeton.edu/∼ aglaser/PU066-Glaser-Barak-Goldston-2012.pdf

12. Josh D. Cohen and Michael J. Fischer. A robust and verifiable cryptographically secure election scheme (extended abstract). In 26th FOCS, pages 372–382. IEEE Computer Society Press, October 1985.