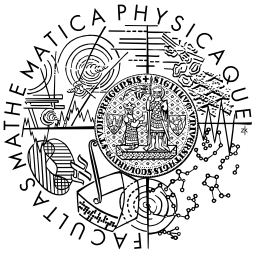

MATEMATICKO-FYZIKÁLNÍ FAKULTA
Univerzita Karlova

BAKALÁŘSKÁ PRÁCE

Martin Struk

Bezpečné sdílené počítání modulo p^k

Katedra algebry

Vedoucí bakalářské práce: doc. Mgr. et Mgr. Jan Žemlička, Ph.D.

Studijní program: Matematika

Studijní obor: Matematika pro informační technologie

Praha 2023

Prohlašuji, že jsem tuto bakalářskou práci vypracoval(a) samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů. Tato práce nebyla využita k získání jiného nebo stejného titulu.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Univerzita Karlova má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle §60 odst. 1 autorského zákona.

V ..... dne ....

Podpis autora

Chtěl bych poděkovat vedoucímu práce doc. Mgr. et Mgr. Janu Žemličkovi, Ph.D. za všechny rady, připomínky, vstřícnost, ochotu a motivaci během vedení této práce.

Název práce: Bezpečné sdílené počítání modulo p^k

Autor: Martin Struk

Katedra: Katedra algebry

Vedoucí bakalářské práce: doc. Mgr. et Mgr. Jan Žemlička, Ph.D., Katedra algebry

Abstrakt: Práce se zabývá odvětvím kryptografie zvaným bezpečné sdílené počítání, což je technika, která umožňuje více stranám spolupracovat na výpočtu jediné funkce tak, že její vstupy zůstanou utajeny. Konkrétněji se práce zabývá bezpečným sdíleným počítáním nad okruhem celých čísel modulo p^k. Práce začíná představením obecného principu protokolů pro bezpečné sdílené počítání, po kterém následuje vybudování potřebné teorie nad komutativními okruhy, která bude v poslední části práce potřeba k popisu a pochopení konkrétního protokolu.

Klíčová slova: bezpečné sdílené počítání, hyper-invertibilní matice, komutativní okruhy, sdílení tajemství, Galoisovy okruhy

Title: Secure multi-party computation modulo p^k

Author: Martin Struk

Department: Department of algebra

Supervisor: doc. Mgr. et Mgr. Jan Žemlička, Ph.D., Department of algebra

Abstract: The thesis deals with a subfield of cryptography called secure multi-party computation which is a technique that allows multiple parties to work together to compute a single function while preserving the privacy of it's inputs. More specifically, the thesis deals with secure multi-party computation over the ring of integers modulo p^k. The thesis begins with an introduction of the general principle of secure multi-party protocols, followed by the construction of the necessary theoretical groundwork over commutative rings, wich will be needed to describe and understand a specific protocol in the last section of the thesis.

Keywords: secure multi-party computation, hyper-invertible matrices, commutative rings, secret sharing, Galois rings

Obsah

Úvod

Obecně o MPC

Nejdříve se seznámíme s tím, co vlastně MPC (Multi-Party Computation) protokoly jsou. MPC je obecně technika, která umožňuje několika stranám spolupracovat při výpočtu nějaké funkce s tajnými vstupy tak, aby se žádná ze stran nemohla dozvědět nic kromě výstupu.

Typickým příkladem takového problému jsou kolegové, kteří chtějí zjistit, jaký je jejich průměrný plat, ale zároveň nechtějí svůj vlastní plat ostatním odhalit. Tento problém by se dal vyřešit tak, že najdou někoho, komu všichni věří, prozradí mu všichni své platy a ten pak průměr spočítá a sdělí jim ho. Přesně tuto stranu navíc MPC protokoly nahrazují. Uvažujme například 4 kolegy s platy 10000, 20000, 30000 a 40000. Protokol by pak mohl fungovat následovně: Každý svůj plat rozdělí na 3 celá (klidně i záporná) čísla, jejichž součet dá dohromady právě plat daného kolegy. Pak každý kolega dostane jedno číslo od každého z ostatních kolegů, ze kterých nemůže zjistit nic o platech ostatních, sečte je a až ze součtů, které dostali všichni kolegové, se udělá průměr. Náš případ by pak tedy mohl vypadat třeba následovně:

Kolega 1: $10000 = 500 + 29500 + (-20000)$
a spočte $5000 + 40000 + 100000 = 145000$

Kolega 2: $20000 = 5000 + 7000 + 8000$
a spočte $29500 + (-2000) + (-1) = 27499$

Kolega 3: $30000 = 40000 + (-2000) + (-8000)$
a spočte $(-20000) + 8000 + (-59999) = (-71999)$

Kolega 4: $40000 = 100000 + (-1) + (-59999)$
a spočte $500 + 7000 + (-8000) = (-500)$

Pak už se jen spočítá $\frac{145000+27499-71999-500}{4} = 25000$, což je správný výsledek ($10000 + 20000 + 30000 + 40000 = 100000$ a $\frac{100000}{4} = 25000$).

MPC protokolů existuje mnoho různě efektivních a především s různou úrovní bezpečnosti pro určité parametry. Bezpečností se ale v této práci příliš zabývat nebudeme. Přestože je samozřejmě velmi důležitá, k popisu a implementaci protokolů potřeba není a práce by pak kvůli ní byla příliš rozsáhlá. Převážně se tedy v tomto ohledu odkáži na jiné zdroje.

Mnoho MPC protokolů pracuje nad konečnými tělesy. Proč tedy pracovat nad okruhem $\mathbb{Z}/p^k\mathbb{Z}$, kde si toho zřejmě spoustu zkomplikujeme? Odpověď je jednoduchá, v některých případech se nám tak běh protokolu značně zefektivní. Vezmeme-li například $\mathbb{Z}/2^{64}\mathbb{Z}$ nebo $\mathbb{Z}/2^{32}\mathbb{Z}$ (počítání modulo 64 nebo 32), budeme počítat podobným způsobem, jakým počítají procesory, na kterých protokol poběží. Zároveň nad těmito okruhy můžeme velmi rychle a efektivně počítat některé užitečné operace, takže se nám běh celkově znatelně urychlí.

Cíle práce

Hlavním cílem práce je detailně vysvětlit teoretické pozadí potřebné k sestavení konkrétního MPC protokolu. V první části práce popíšeme obecnou teorii a nástroje, které budeme dále potřebovat. V druhé části práce si představíme hyper-invertibilní matice, jejichž využití bylo jedním z hlavních přínosů protokolu, který si popíšeme, protože vyznámně zefektivnily jeden z algoritmů využitých v protokolu. O hyper-invertibilních maticích si v této části pomocí teorie z první části dokážeme některá důležitá tvrzení. V třetí části poté popíšeme již zmíněný konkrétní MPC protokol a vysvětlíme, kde se využívá vybudovaná teorie z předchozích částí.

1. Potřebná teorie

1.1 Základní notace a definice

Nejdříve si připomeneme základní značení, které budeme často používat, nebo jsme již použili v úvodu. $\mathbb{Z}$ bude značit okruh celých čísel a $\mathbb{N}$ přirozená čísla (bez 0). Pro komutativní okruh R (dále budeme vždy předpokládat, že je R komutativní okruh s jednotkou) budeme jako R^* značit jeho multiplikativní grupu všech invertibilních prvků a $R[x]$ bude značit okruh všech polynomů v proměnné x s prvky R jako koeficienty. Navíc pro $m \in \mathbb{Z}$, $m \geq 0$ budeme $R[x]_{\leq m} \subset R[x]$ značit množinu všech polynomů z $R[x]$ stupně nejvýše m. Některé definice a tvrzení budeme přebírat především z kurzu algebry a úvodu do komutativní algebry (Stanovský (2022); Kala (2023)). Dále budeme používat definice z lineární algebry (Barto a Tůma (2023)) jako například definici invertibilní matice, inverzní matice, adjungované matice a podobně, které by se nad obecnými komutativními okruhy definovaly totožně jako nad tělesy.

Definice $\emptyset \neq I \subseteq R$ nazveme *ideálem* v R, pokud

- pro každé $a, b \in I$ platí $a - b \in I$
- pro každé $a \in I$ a každé $r \in R$ platí $r \cdot a \in I$

Ideály generované pomocí $a \in R$ značíme $(a) = \{a \cdot r \mid r \in R\}$. Ideál, který lze takto nagenerovat jediným prvkem, nazýváme hlavní ideál. Pro ideály $I, J \subseteq R$ také definujeme součin ideálů jako ideál $IJ = \{\sum_{i=0}^{n} a_i b_i \mid a_i \in I, b_i \in J, n \in \mathbb{N}\}$ a součet ideálů jako ideál $I + J = \{a + b \mid a \in I, b \in J\}$. Pokud $I + J = R$, pak řekneme, že jsou ideály komaximální.

Definice Nechť je I ideál v R. Definujeme ekvivalenci na množině prvků z okruhu R předpisem

$$a \sim b \iff a - b \in I$$

Bloky této ekvivalence jsou rozkladové třídy $[a] = a + I$, na kterých definujeme operace

$$[a] + [b] = [a + b], -[a] = [-a], [a] \cdot [b] = [a \cdot b]$$

Množina bloků s těmito operacemi se nazývá *faktorokruh okruhu R podle ideálu I*,

$$R/I = (\{[a] : a \in R\}, + , - , \cdot ,[0],[1]).$$

Pozorování Operace na blocích jsou dobře definovány a R/I je skutečně okruh (dokázáno v Stanovský (2022)).

Definice *R-modul* je abelovská grupa $(M, + , - ,0)$ spolu s operací $R \times M \to M$, které splňují $\forall r, s \in R, \forall m, n \in M$:

- $r(m + n) = rm + rn$
- $r(sm) = (rs)m$
- $(r + s)m = rm + sm$
- $1m = m$

Dále definujeme *R-modulový homomorfismus* jako funkci $f : M \to N$, kde N a M jsou R-moduly, pro které platí $\forall r \in R, \forall m, n \in M$:

- $f(m+n) = f(m) + f(n)$
- $f(rm) = rf(m)$

Pokud je navíc f bijekce, řekneme, že je to *R-modulový izomorfismus*.

Pozorování Pro $n \in \mathbb{N}$ je R^n s operacemi po složkách R-modul.

Pozorování Pro $m \in \mathbb{N}$ je $R[x]_{\leq m}$ s operacemi z $R[x]$ a násobením skalárem R-modul.

Lemma 1 Pro každý R-modulový izomorfismus $\varphi : R^n \to R^n$ existuje invertibilní matice $M \in R^{n \times n}$ taková, že $\varphi(\underline{x}) = \underline{x}M$ pro všechna $\underline{x} = (x_1,...,x_n) \in R^n$ (řádkové vektory).

Důkaz. Uvažujme $M = \begin{pmatrix} \varphi(\underline{e_1}) \\ \vdots \\ \varphi(\underline{e_n}) \end{pmatrix}$, kde $\underline{e_i}$ jsou řádkové vektory kanonické báze. Protože je φ izomorfismus, platí

$$\varphi(\underline{x}) = x_1 \cdot \varphi(\underline{e_1}) + ... + x_n \cdot \varphi(\underline{e_n})$$

Je zřejmé, že toto se přesně rovná součinu $\underline{x}M$. Zbývá dokázat, že je M invertibilní. Protože je φ izomorfismem, musí k němu existovat φ^{-1}, které je také izomorfismem. Z toho, co jsme již dokázali, víme, že existuje matice N, která reprezentuje φ^{-1}. Dále víme, že pro všechna $\underline{x} \in R^n$ platí $\varphi(\varphi^{-1}(\underline{x})) = \varphi^{-1}(\varphi(\underline{x})) = id(\underline{x}) = \underline{x}$, z čehož plyne, že $\underline{x}MN = \underline{x}NM = \underline{x}I_n = \underline{x}$ (zde id značí standardně identické zobrazení a I_n jednotkovou matici řádu n). Toto platí pro všechna x, specielně tedy i pro e_i pro všechna i, takže $MN = NM = I_n$, což je přesně definice invertibility. □

Pro zobrazení φ a matici M jako z předchozího lemmatu budeme říkat, že *φ lze reprezentovat maticí M*.

1.2 Čínská zbytková věta a interpolace

Nyní si připomeneme některá důležitá tvrzení a jejich důsledky. Všechna si nebudeme zcela dokazovat, místo toho se odkáži na skripta některých kurzů, kde se daná tvrzení dokazovala.

Pozorování Pro $m \in \mathbb{N}$ a $I_1,...,I_m$ ideály jsou $R/(I_1 \cdot ... \cdot I_m)$ s operacemi jako v definici faktorokruhu okruhu R a $R/I_1 \times ... \times R/I_m$ s operacemi po složkách R-moduly.

Tvrzení 2 [Čínská zbytková věta] Nechť $I_1,..., I_m, m \in \mathbb{N}$ jsou po dvou komaximální ideály v R. Pak zobrazení

$$R/(I_1 \cdot ... \cdot I_m) \to R/I_1 \times ... \times R/I_m$$

$$r + (I_1 \cdot ... \cdot I_m) \mapsto (r + I_1,...,r + I_m)$$

je R-modulový izomorfismus.

Důkaz. Dle Věty 6.3 z Stanovský a Barto (2017) je zobrazení $R/(I_1 \cap ... \cap I_m) \to R/I_1 \times ... \times R/I_m$ dané $r+(I_1 \cap ... \cap I_m) \mapsto (r+I_1,...,r+I_m)$ okruhový izomorfismus. Že jde o R-modulový homomorfismus plyne prímo z definice/pozorování výše. Dokážeme tedy pouze, že pro po dvou komaximální ideály platí $I_1 \cap ... \cap I_m = I_1 \cdot ... \cdot I_m$.

Indukcí podle m. Pro $m = 2$ máme komaximální ideály I_1 a I_2, takže máme $1 = a + b$ pro nějaké $a \in I_1$ a $b \in I_2$. Inkluze $I_1 \cdot I_2 \subseteq I_1 \cap I_2$ je zřejmá z definice ideálů a jejich součinu. Pro opačnou inkluzi mějme libovolné $x \in I_1 \cap I_2$. To si můžeme napsat jako $x \cdot 1 = x \cdot (a + b) = xa + xb$, kde xa i xb musí ležet v $I_1 \cdot I_2$, protože $x \in I_2, a \in I_1$ a $x \in I_1, b \in I_2$. Z toho plyne, že $x \in I_1 \cdot I_2$.

Pro $m > 2$ nejdříve dokážeme, že $I_1 \cdot ... \cdot I_{m-1}$ a I_m jsou komaximální. Uvažujme

$$R = (I_1 + I_m)(I_2 + I_m)...(I_{m-1} + I_m) = I_1 \cdot I_2 \cdot ... \cdot I_{m-1} +$$

$+$ další členy, ze kterých lze ze všech vytknout $I_m \subseteq I_1 \cdot I_2 \cdot ... \cdot I_{m-1} + I_m \subseteq R$.

Z toho nám tedy vyplývá, že $I_1 \cdot I_2 \cdot ... \cdot I_{m-1} + I_m = R$.

Nyní už máme z případu pro $m = 2$ (první rovnost) a indukčního předpokladu (druhá rovnost)

$$(I_1 \cdot ... \cdot I_{m-1}) \cdot I_m = (I_1 \cdot ... \cdot I_{m-1}) \cap I_m = (I_1 \cap ... \cap I_{m-1}) \cap I_m.$$

□

Tvrzení 3 Nechť $g, h \in R[x], h$ monický (vedoucí koeficient je 1). Pak existují jednoznačně určené $q, r \in R[x]$ takové, že

- $g(x) = h(x)q(x) + r(x)$ a
- $deg(r(x)) < deg(h(x))$, kde deg značí stupeň daného polynomu.

Důkaz. Toto tvrzení bylo pro obory dokázáno v kurzu algebry (Tvrzení 3.2 Stanovský (2022)). Toho, že je R obor, se využilo pouze při důkazu jednoznačnosti. My zde místo toho využijeme podmínky, že h musí být monický. Nechť $g(x) = h(x)q_1(x) + r_1(x) = h(x)q_2(x) + r_2(x)$. Pak $h(x)(q_1(x) - q_2(x)) = r_2 - r_1$, z čehož plyne, že $h(x)$ dělí $r_2(x) - r_1(x)$. Ale $deg(r_2(x) - r_1(x)) < deg(h(x))$, takže $r_2(x) - r_1(x) = 0$, čili $r_2(x) = r_1(x)$. Nyní máme $h(x)q_1(x) = h(x)q_2(x)$. Protože je $h(x)$ monický, je zároveň nenulový. Pokud je tedy $q_1(x)$ nebo $q_2(x)$ nulové, musí být obě nulová, tudíž $q_1(x) = q_2(x)$. Pokud je $q_1(x)$ i $q_2(x)$ nenulové, bude vedoucí koeficient levé strany rovnice roven vedoucímu koeficientu $q_1(x)$ a vedoucí koeficient pravé strany bude roven vedoucímu koeficientu $q_2(x)$, protože vedoucí koeficient $h(x)$ je 1. Z toho především plyne, že se při součinu vedoucí koeficienty obou stran nevynulují. Tím jsme přesně nahradili využití předpokladu ze skript, že R musí být obor, takže tedy i v našem případě $q_1(x) = q_2(x)$. Zbytek důkazu tedy už projde i v našem případě. Existenci lze dokázat pomocí algoritmu pro dělení se zbytkem, který zde také funguje. □

Důsledek 4 (1) Nechť je $h(x) \in R[x]$ monický a $deg(h(x)) = d$. Pak máme R-modulový izomorfismus

$$R[x]_{\leq d-1} \to R[x]/(h(x))$$

$$f(x) \mapsto f(x) + (h(x))$$

(2) Pokud $h(x) = x - \alpha$, $\alpha \in R$, pak máme R-modulový izomorfismus

$$R[x]/(x-\alpha) \to R$$

$$f(x) + (x-\alpha) \mapsto f(\alpha)$$

Důkaz. (1) Z definice okamžitě vidíme, že zobrazení je R-modulový homomorfismus. Uvažujme f,$g \in R[x]_{\leq d-1}$, takové, že $f(x) + (h(x)) = g(x) + (h(x))$. To znamená, že $f(x) - g(x) \in (h(x))$, neboli $h(x)$ dělí $f(x) - g(x)$, ale $deg(h(x)) = d$ a $deg(f(x)) = deg(g(x)) \leq d-1$, takže $f(x) - g(x) = 0$ (Tvrzení 3). Z toho plyne, že $f(x) = g(x)$, takže je zobrazení prosté. Že je zobrazení na, plyne z toho, že $deg(f(x) + (h(x))) < d$ pro všechna $f \in R[x]$, takže $f(x) \bmod (h(x)) \in R[x]_{\leq d-1}$. Tento prvek se tedy zobrazí na $f(x) + (h(x)) \in R[x]/(h(x))$.

(2) Označme si dané zobrazení jako φ a mějme f,$g \in R[x]$. Nejprve dokážeme, že je zobrazení korektní (neboli že nezáleží na výběru reprezentanta rozkladové třídy) a prosté. Předpokládejme tedy nejdříve, že $f(\alpha) = g(\alpha)$ a dokažme, že $f(x) + (x-\alpha) = g(x) + (x-\alpha)$. Z $f(\alpha) - g(\alpha) = 0$ máme $(f-g)(\alpha)$, takže je α kořenem $(f-g)(x)$. To znamená, že $(x-\alpha)$ dělí $(f - g)(x)$ (Tvrzení 3 + Tvrzení 3.3 z Stanovský (2022)), takže máme $(f-g)(x) = f(x) - g(x) \in (x-\alpha)$, což přesně odpovídá definici ekvivalence pro faktorokruhy, neboli

$$f(x) + (x-\alpha) = g(x) + (x-\alpha).$$

Naopak předpokládáme-li, že $f(x) + (x-\alpha) = g(x) + (x-\alpha)$, pak $(x-\alpha)$ dělí $(f-g)(x)$, takže musí platit $(f-g)(x) = (x-\alpha) \cdot u(x)$ pro nějaký polynom $u \in R[x]$ (Tvrzení 3). Dosazením α dostaneme

$$f(\alpha) - g(\alpha) = (f-g)(\alpha) = (\alpha - \alpha) \cdot u(\alpha) = 0 \cdot u(\alpha) = 0$$

Zobrazení je tedy korektní a prosté. Zároveň je na, protože všechny konstantní polynomy se nám zřejmě zobrazí na celé R. Zbývá dokázat, že je to R-modulový homomorfismus.

$$\varphi(f(x) + (x-\alpha)) + \varphi(g(x) + (x-\alpha)) = f(\alpha) + g(\alpha) = (f+g)(\alpha) =$$

$$= \varphi((f+g)(x) + (x-\alpha)) = \varphi((f(x) + (x-\alpha)) + (g(x) + (x-\alpha)))$$

Že $r \cdot \varphi(f(x) + (x-\alpha)) = \varphi(r \cdot (f(x) + (x-\alpha)))$ pro $r \in R$ je zřejmé, protože nezáleží na tom, jestli do polynomu nejdříve dosadíme α a poté ho vynásobíme r, nebo nejdříve jeho koeficienty vynásobíme r a poté dosadíme α. φ je tedy R-modulový homomorfismus. □

Tento důsledek bychom jednoduše dostali pomocí 1. věty o izomorfismu. Museli bychom ale dokázat její verzi pro okruhy, kterou bychom už jinde nepoužili. Proto je důkaz proveden takto přímočaře.

Definice Mějme α_1,...,$\alpha_n \in R$. Řekneme, že tyto prvky tvoří *výjimečnou posloupnost v* R, pokud $\forall i,j \in \{1,...,n\}, i \neq j$ platí $\alpha_i - \alpha_j \in R^*$. Také definujeme *Lenstrovu konstantu okruhu* R jako maximální možnou délku výjimečné posloupnosti v R, pokud je R konečný okruh (s nekonečnými okruhy pracovat nebudeme).

Tvrzení 5 Nechť $\alpha_1,...,\alpha_m \in R$ tvoří výjimečnou posloupnost v R. Pak je zobrazení

$$R[x]_{\leq m-1} \to R \times ... \times R$$
$$f(x) \mapsto (f(\alpha_1),...,f(\alpha_m))$$

R-modulovým izomorfismem. Pro každá $x_1,...,x_m \in R$ tedy existuje jednoznačně určený interpolační polynom $f \in R[x]$ takový, že $deg(f(x)) \leq m-1$ a $f(\alpha_i) = x_i$ pro všechna i.

Důkaz. Mějme $h(x) := (x-\alpha_1)\cdot...\cdot(x-\alpha_m)$. Podle Důsledku 4 máme R-modulový izomorfismus

$$R[x]_{\leq m-1} \to R[x]/(h(x))$$
$$f(x) \mapsto f(x) + (h(x))$$

Dále díky Tvrzení 2 máme R-modulový izomorfismus

$$R[x]/(h(x)) \to R[x]/(x-\alpha_1) \times ... \times R[x]/(x-\alpha_m)$$
$$f(x) + (h(x)) \mapsto (f(x) + (x-\alpha_1),...,f(x) + (x-\alpha_m))$$

Toto tvrzení můžeme použít, protože ideály $(x-\alpha_i)$ jsou po dvou komaximální pro všechna i díky tomu, že $\alpha_i - \alpha_j$ jsou (z předpokladu) invertibilní pro všechna $i \neq j$. V ideálu $(x-\alpha_i)+(x-\alpha_j)$ totiž zřejmě leží $(x-\alpha_j)-(x-\alpha_i) = \alpha_i - \alpha_j$. Tím pádem v něm leží i $(\alpha_i - \alpha_j)\cdot(\alpha_i - \alpha_j)^{-1} = 1$ a jakýkoliv ideál obsahující 1 už musí být roven celému okruhu. Nakonec podle Důsledku 4 dostaneme pro každé i izomorfismus

$$R[x]/(x-\alpha_i) \to R$$
$$f(x) + (x-\alpha_i) \mapsto f(\alpha_i)$$

Složením těchto zobrazení získáme požadovaný R-modulový izomorfismus. □

Definice Nechť je $\alpha_0,...,\alpha_n \in R$ pro $n \in \mathbb{N}$ výjimečná posloupnost a $m \in \mathbb{N}$, $m \leq n$. Definujeme R-modul *vektorů částí tajemství* jako

$$C_m = \{(f(\alpha_0),...,f(\alpha_n)) \mid f \in R[x]_{\leq m}\}.$$

Pozorování C_m je s operacemi definovanými po složkách R-modul.

Algoritmus 1 (Shamirovo sdílení tajemství nad R mezi n stran)

- *Vstup:* $\alpha_0,...,\alpha_n \in R$,která tvoří vyjímečnou posloupnost, $a \in R$.
- Zvolíme libovolný polynom $f \in R[x]_{\leq m}$ takový, že $f(\alpha_0) = a$.
- Pro $i = 1,...,n$ zvolíme za i-tou část tajemství a prvek $f(\alpha_i)$ a tento prvek obdrží i-tá strana.

Poté pokud známe alespoň $m+1$ částí tajemství, můžeme pomocí interpolace zjistit polynom f, ze kterého zjistíme i tajemství ($f(\alpha_0) = a$). Všimněme si také, že $(f(\alpha_0),...,f(\alpha_n)) \in C_m$ z výše uvedené definice. Můžeme si tedy také představit, že v prvním kroku algoritmu místo polynomu f vybíráme vektor částí tajemství $x = (x_0,...,x_n) \in C_m$ takový, že $x_0 = a$.

Pozorování Hodnota m z předchozího algoritmu musí být menší než Lenstrova konstanta okruhu R. To znamená, že počet stran, které sdílejí tajemství, nemůže tuto hodnotu překročit.

1.3 Galoisovy okruhy

Galoisovy okruhy si můžeme představit jako okruhovou verzi konečných těles (neboli Galoisových těles). Budeme s nimi pracovat, protože obsahují $\mathbb{Z}/p^k\mathbb{Z}$ jako podokruh a navíc mají relativně vysokou Lenstrovu konstantu. Důkazy z této podsekce vynecháme, čtenář je může najít v Wan (2003).

Definice *Galoisovým okruhem* nazveme okruh $R := (\mathbb{Z}/p^k\mathbb{Z})[y]/(h(y))$, kde p je prvočíslo, $k \in \mathbb{N}$ a $h(y) \in (\mathbb{Z}/p^k\mathbb{Z})[y]$ je nekonstantní monický polynom takový, že $h(y) \bmod p$ je ireducibilní polynom v $\mathbb{F}_p[y]$. Takový Galoisův okruh budeme také značit $GR(p^k, d)$, kde d značí stupeň polynomu $h(y)$ (toto značení můžeme používat, protože všechny Galoisovy okruhy se stejnými parametry p,k,d jsou navzájem izomorfní).

Poznámka 6 Lenstrova konstanta okruhu $R = GR(p^k, d)$ je p^d.

Poznámka 7 Nechť $R = GR(p^k, d)$. Pak $(p) \subsetneq R$ je maximální ideál a máme $R/(p) \cong \mathbb{F}_{p^d}$. Specielně tedy máme homomorfismus $\pi : R \to \mathbb{F}_{p^d}$, kterému říkáme *redukce modulo p*.

Poznámka 8 Nechť $R = GR(p^k, d)$. Pak existuje $\mathbb{Z}/p^k\mathbb{Z}$-modulový izomorfismus $(\mathbb{Z}/p^k\mathbb{Z})^d \to R$, který zobrazí každý prvek $\underline{e_j} = (0,...,1,...,0)$ kanonické báze $(\mathbb{Z}/p^k\mathbb{Z})^d$ na $y^j + (h(y))$, kde $h(y)$ je jako v definici Galoisova okruhu. Také máme vnoření $\mathbb{Z}/p^k\mathbb{Z} \hookrightarrow R$ dané $x \mapsto x + h(y)$.

Navíc můžeme prvky R reprezentovat následujícím způsobem. Protože víme, že $R/(p) \cong \mathbb{F}_{p^d}$, máme prvek $\xi \in R^*$ řádu $p^d - 1$. Definujeme $\mathcal{I} = \{0,1,\xi,...,\xi^{p^d-2}\}$. Nyní můžeme libovolné $a \in R$ reprezentovat jako $a = \sum_{i=0}^{k-1} a_i p^i$, kde $a_0,...,a_{k-1} \in \mathcal{I}$. Všimněme si, že redukce modulo p (π z Poznámky 7) je definováno jako $\pi(a) = a_0$.

Díky tomuto rozkladu vidíme, že pro $u \in \mathbb{N}$ p^u dělí a právě tehdy, když $a_i = 0$ pro všechna $i < u$. Pokud takové u existuje, tak můžeme definovat a/p^u jako

$$a/p^u = a_u + a_{u+1}p + ... + a_{k-1}p^{k-u-1}.$$

Všimněme si, že $a/p^u \equiv a_u (\bmod\ p)$ a pokud je u největší možné (neboli takové, že p^u dělí a, ale p^{u+1} nedělí a) a a je nenulový prvek R, pak $a/p^u \in R^*$.

Příklad Jako příklad si uvedeme Galoisův okruh $R = GR(2^2,2)$. Ten můžeme zkonstruovat například jako $(\mathbb{Z}/4\mathbb{Z})[y]/(y^2+y+1)$, protože je očividné, že y^2+y+1 (označíme si tento polynom $h(y)$) je nekonstantní monický ireducibilní polynom nad $\mathbb{F}_2$. R má tedy 16 prvků tvaru $ay + b + (h(y))$ pro $a,b \in \{0,1,2,3\}$. Není těžké (například hrubou silou) ověřit, že invertibilní prvky jsou právě prvky tvaru $ay+b+(h(y))$, kde alespoň jedno z čísel a,b je rovno 1 nebo 3. Pokud tedy budeme hledat výjimečnou posloupnost, potřebujeme, aby rozdíl každých dvou prvků měl alespoň jednu z hodnot a,b lichou.

Podle Poznámky 6 má nad tímto okruhem nejdelší výjimečná posloupnost $p^d = 2^2 = 4$ prvky. Zvolme jako první prvek nějaké $ay + b + (h(y)) \in R$, kde a,b jsou oba sudé. Hledáme 3 prvky tvaru $cy + d + (h(y)) \in R$, aby měl jejich rozdíl s naším prvkem alespoň jeden z koeficientů $a - c$,$b - d$ (zároveň ale i $c - a$,$d - a$) lichý. To znamená, že c nebo d musí být liché, protože rozdíl sudého a lichého čísla je lichý a zároveň rozdíl lichého a sudého čísla je lichý. Máme tedy 3 možnosti:

c liché, d liché, obě lichá. Vezměme například za liché koeficienty 1 a za sudé 2. Pak dostaneme posloupnost

$$2y + 2 + (h(y)),\ y + 2 + (h(y)),\ 2y + 1 + (h(y)),\ y + 1 + (h(y)).$$

Nyní můžeme jednoduše výpočtem všech možných rodílů ověřit, že tato posloupnost je opravdu výjimečná (každý rozdíl bude mít alespoň jeden z koeficientů buď 1, nebo 3, takže bude invertibilní).

$$(2y + 2) - (y + 2) = y \wedge (y + 2) - (2y + 2) = 3y$$

$$(2y + 2) - (2y + 1) = 1 \wedge (2y + 1) - (2y + 2) = 3$$

$$(2y + 2) - (y + 1) = y + 1 \wedge (y + 1) - (2y + 2) = 3y + 3$$

$$(y + 2) - (2y + 1) = 3y + 1 \wedge (2y + 1) - (y + 2) = y + 3$$

$$(y + 2) - (y + 1) = 1 \wedge (y + 1) - (y + 2) = 3$$

$$(2y + 1) - (y + 1) = y \wedge (y + 1) - (2y + 1) = 3y$$

Nyní zkusme nad tímto R najít množinu $\mathcal{I}$. Potřebujeme tedy nějaký prvek řádu $p^d - 1 = 2^2 - 1 = 3$. To je například $y + (h(y))$, protože $y^2 = 3y + 3$ a $y^3 = 1$. Takže $\mathcal{I} = \{0,1,y,3y + 3\}$. Nyní můžeme podle Poznámky 8 napsat libovolné $a \in R$ jako

$$a = \sum_{i=0}^{k-1} a_i p^i = a_0 \cdot 2^0 + a_1 \cdot 2^1, a_0,a_1 \in \mathcal{I}$$

Například rozklady prvků naší výjimečné posloupnosti, kterou jsme našli výše, budou vypadat následovně:

$$2y + 2 = 0 \cdot 2^0 + (3y + 3) \cdot 2^1$$

$$y + 2 = y \cdot 2^0 + 1 \cdot 2^1$$

$$2y + 1 = 1 \cdot 2^0 + y \cdot 2^1$$

$$y + 1 = (3y + 3) \cdot 2^0 + (3y + 3) \cdot 2^1$$

2. Hyper-invertibilní matice

Definice Řekneme, že matice $M \in R^{i\times j}$ je *hyper-invertibilní*, pokud pro všechny podmnožiny indexů $I \subseteq \{1,...,i\}$ a $J \subseteq \{1,...,j\}$ takové, že $|I| = |J| > 0$, je matice M_I^J invertibilní. Zde M_I značí podmatici všech řádků s indexy z I, M^J značí podmatici všech sloupců s indexy z J a $M_I^J := (M_I)^J$.

Hyper-invertibilní matice budeme používat kvůli jejich vlastnosti, která nám umožní velmi efektivně generovat a kontrolovat korektnost hodnot v našem protokolu. Tuto vlastnost na konci této sekce dokážeme. Navíc jsme z jedné hyper-invertiblní matice schopni z definice triviálně získat hyper-invertibilní matice nižších řádů. Nejdříve si ale ukážeme, že lze takové matice zkonstruovat (a zároveň v důkazu ukážeme jak to provést).

Tvrzení 9 (Konstrukce hyper-invertibilní matice) Nechť $n, k \in \mathbb{N}$, p je prvočíslo a $R = GR(p^k, d)$, pro který platí $p^d \geq 2n$. Ať je navíc $\alpha_1,...,\alpha_{2n}$ výjimečná posloupnost v R. Pak existuje R-modulový izomorfismus

$$R^n \to R^n$$

$$(f(\alpha_1),...,f(\alpha_n)) \mapsto (f(\alpha_{n+1}),...,f(\alpha_{2n}))$$

Tento izomorfismus lze reprezentovat hyper-invertibilní maticí

$$\begin{pmatrix} \prod_{k=2}^{n} \frac{\alpha_{n+1}-\alpha_k}{\alpha_1-\alpha_k} & \dots & \prod_{k=2}^{n} \frac{\alpha_{n+n}-\alpha_k}{\alpha_1-\alpha_k} \\ \vdots & & \vdots \\ \prod_{k=1}^{n-1} \frac{\alpha_{n+1}-\alpha_k}{\alpha_n-\alpha_k} & \dots & \prod_{k=1}^{n-1} \frac{\alpha_{n+n}-\alpha_k}{\alpha_n-\alpha_k} \end{pmatrix}.$$

Důkaz. Daný izomorfismus dostaneme tak, že dvakrát použijeme Tvrzení 5. Uvažujme tedy $\varphi_1 : R[x]_{\leq n-1} \to R \times ... \times R$ dané $f(x) \mapsto (f(\alpha_1),...,f(\alpha_n))$ a $\varphi_2 : R[x]_{\leq n-1} \to R \times ... \times R$ dané $f(x) \mapsto (f(\alpha_{n+1}),...,f(\alpha_{2n}))$. Chceme zjistit, jak vypadá φ_1^{-1}. Uvažujme pro $\underline{e_i} = (e_{i1},...,e_{in})$ (i-tý vektor kanonické báze, takže $e_{ii} = 1$, jinak $e_{ij} = 0$ pro $i \neq j$)

$$\varphi_1^{-1}(\underline{e_i}) = \sum_{j=1}^{n} e_{ij} \cdot \prod_{k=1,k\neq j}^{n} \frac{x-\alpha_k}{\alpha_j-\alpha_k} = \prod_{k=1,k\neq i}^{n} \frac{x-\alpha_k}{\alpha_i-\alpha_k}$$

Označíme-li tento polynom f, můžeme snadno dosazením ověřit, že $\underline{e_i}$ se rovná $(f(\alpha_1),...,f(\alpha_n))$ pro všechna i, takže je φ_1^{-1} opravdu inverzním zobrazením k φ_1. Pak máme tedy $\varphi_2(\varphi_1^{-1}(\underline{e_i})) = (\prod_{k=1,k\neq i}^{n} \frac{\alpha_{n+1}-\alpha_k}{\alpha_i-\alpha_k},..., \prod_{k=1,k\neq i}^{n} \frac{\alpha_{n+n}-\alpha_k}{\alpha_i-\alpha_k})$, což nám pro všechny vektory kanonické báze dává matici

$$M = \begin{pmatrix} \prod_{k=2}^{n} \frac{\alpha_{n+1}-\alpha_k}{\alpha_1-\alpha_k} & \dots & \prod_{k=2}^{n} \frac{\alpha_{n+n}-\alpha_k}{\alpha_1-\alpha_k} \\ \vdots & & \vdots \\ \prod_{k=1}^{n-1} \frac{\alpha_{n+1}-\alpha_k}{\alpha_n-\alpha_k} & \dots & \prod_{k=1}^{n-1} \frac{\alpha_{n+n}-\alpha_k}{\alpha_n-\alpha_k} \end{pmatrix},$$

kterou lze náš R-modulový izomorfismus $R^n \to R^n$ reprezentovat. Zbývá tedy dokázat, že je M hyper-invertibilní. Dokážeme nejprve, že matice je invertibilní právě tehdy, když je její determinant invertibilním prvkem v R. Mějme tedy invertibilní matici $A \in R^{n\times n}$ a označme I_n jednotkovou matici řádu n (standardní značení). Pak

$$1 = det(I_n) = det(A \cdot A^{-1}) = det(A) \cdot det(A^{-1})$$

Z toho plyne, že $det(A)^{-1} = det(A^{-1})$, takže $det(A) \in R^*$.

Pro opačnou implikaci předpokládejme, že $det(A) \in R^*$. Naprosto stejně jako v lineární algebře bychom i nad komutativním okruhem dokázali, že $A \cdot adj(A) = det(A) \cdot I_n$, takže $A \cdot (det(A)^{-1} \cdot adj(A)) = I_n$, protože je $det(A)$ invertibilní. Máme tedy $A^{-1} = det(A)^{-1} \cdot adj(A)$, takže je A invertibilní.

Pomocí tohoto tvrzení nyní dokážeme, že je M hyper-invertibilní. Již víme, že je M invertibilní z Lemma 1. Z toho máme $det(M) \in R^*$. Podívejme se nyní na determinant matice, již dostaneme odebráním jednoho řádku a jednoho sloupce z M. Bez újmy na obecnosti odeberme n-tý sloupec a n-tý řádek a označme tuto matici N.

$$det(N) = \begin{vmatrix} \prod_{k=2}^{n} \frac{\alpha_{n+1}-\alpha_k}{\alpha_1-\alpha_k} & \cdots & \prod_{k=2}^{n} \frac{\alpha_{2n-1}-\alpha_k}{\alpha_1-\alpha_k} \\ \vdots & & \vdots \\ \prod_{\substack{k=1 \\ k\neq n-1}}^{n} \frac{\alpha_{n+1}-\alpha_k}{\alpha_{n-1}-\alpha_k} & \cdots & \prod_{\substack{k=1 \\ k\neq n-1}}^{n} \frac{\alpha_{2n-1}-\alpha_k}{\alpha_{n-1}-\alpha_k} \end{vmatrix} =$$

$$\begin{vmatrix} \frac{\alpha_{n+1}-\alpha_n}{\alpha_1-\alpha_n} \cdot \prod_{k=2}^{n-1} \frac{\alpha_{n+1}-\alpha_k}{\alpha_1-\alpha_k} & \cdots & \frac{\alpha_{2n-1}-\alpha_n}{\alpha_1-\alpha_n} \cdot \prod_{k=2}^{n-1} \frac{\alpha_{2n-1}-\alpha_k}{\alpha_1-\alpha_k} \\ \vdots & & \vdots \\ \frac{\alpha_{n+1}-\alpha_n}{\alpha_{n-1}-\alpha_n} \cdot \prod_{k=1}^{n-2} \frac{\alpha_{n+1}-\alpha_k}{\alpha_{n-1}-\alpha_k} & \cdots & \frac{\alpha_{2n-1}-\alpha_n}{\alpha_{n-1}-\alpha_n} \cdot \prod_{k=1}^{n-2} \frac{\alpha_{2n-1}-\alpha_k}{\alpha_{n-1}-\alpha_k} \end{vmatrix} =$$

$$\prod_{k=1}^{n-1} \frac{1}{\alpha_k - \alpha_n} \cdot \prod_{k=1}^{n-1} (\alpha_{n+k} - \alpha_n) \cdot \begin{vmatrix} \prod_{k=2}^{n-1} \frac{\alpha_{n+1}-\alpha_k}{\alpha_1-\alpha_k} & \cdots & \prod_{k=2}^{n-1} \frac{\alpha_{2n-1}-\alpha_k}{\alpha_1-\alpha_k} \\ \vdots & & \vdots \\ \prod_{k=1}^{n-2} \frac{\alpha_{n+1}-\alpha_k}{\alpha_{n-1}-\alpha_k} & \cdots & \prod_{k=1}^{n-2} \frac{\alpha_{2n-1}-\alpha_k}{\alpha_{n-1}-\alpha_k} \end{vmatrix}$$

Druhá rovnost platí, protože vytčené zlomky mají v řádcích stejné jmenovatele a ve sloupcích stejné čitatele, takže je z determinantu můžeme vytknout. Všimněme si, že matice po vytknutí má stejný tvar jako matice M, akorát je řádu $n-1$. Jelikož jsme v tvrzení vybírali n libovolně ($p^d \geq 2n$ samozřejmě pro menší n stále platí) a vyjímečná posloupnost z definice zůstane výjimečnou posloupností i po odebrání libovolných prvků, je tato matice také invertibilní ze stejného důvodu jako M. Tím pádem je i její determinant invertibilní. Ten násobíme také invertibilním prvkem, protože $\alpha_i - \alpha_j$ je vždy invertibilní, takže i jejich součin je invertibilní. Z toho plyne, že i $det(N)$ je invertibilní, takže celkem je N invertibilní.

Stejným postupem a opakováním tohoto postupu dostaneme, že je matice získaná odebráním libovolných m řádků a m sloupců ($m \in \mathbb{N}, m \leq n$) stále invertibilní, což přesně odpovídá definici hyper-invertibility M. □

Definice Nechť $\underline{x} = (x_1,...,x_n) \in R^n$ a $I \subseteq \{1,...,n\}$. Pak $\underline{x_I} \in R^{|I|}$ bude značit vektor všech prvků z $\underline{x}$ s indexy z I (podobně jako značení pro matice z definice hyper-invertibilní matice) a $\overline{I}$ bude značit množinu $\{1,...,n\} \setminus I$.

Definice Pro vektory $x = (x_1,...,x_n) \in R^n, y = (y_1,...,y_m) \in R^m$ definujeme vektor $(x,y) = (x_1,...,x_n,y_1,...,y_m) \in R^{n+m}$.

Lemma 10 Nechť je $M \in R^{n\times n}$ hyper-invertibilní matice a $I,J \subseteq \{1,...,n\}$ jsou množiny indexů takové, že $|I|+|J| = n$. Potom existuje R-modulový izomorfismus $\varphi : R^n \to R^n$ takový, že pro každá $\underline{x},\underline{y} \in R^n$ taková, že $\underline{y} = \underline{x}M$, platí

$$\varphi((\underline{x_I}, \underline{y_J})) = (\underline{x_{\overline{I}}}, \underline{y_{\overline{J}}}).$$

Důkaz. Z $\underline{y} = \underline{x}M$ snadno vidíme, že $\underline{y_J} = \underline{x}M^J = \underline{x_I}M_I^J + \underline{x_{\overline{I}}}M_{\overline{I}}^J$. Protože $|I| + |J| = n$, musí platit $|\overline{I}| = |J|$. $M_{\overline{I}}^J$ je tedy čtvercová matice, takže je díky hyper-invertibilitě M zároveň i invertibilní. Můžeme tedy vyjádřit $\underline{x_{\overline{I}}}$ jako $\underline{x_{\overline{I}}} = (\underline{y_J} - \underline{x_I}M_I^J)(M_{\overline{I}}^J)^{-1}$. Analogicky dostaneme $\underline{y_{\overline{J}}} = \underline{x}M^{\overline{J}} = \underline{x_I}M_I^{\overline{J}} + \underline{x_{\overline{I}}}M_{\overline{I}}^{\overline{J}} = \underline{x_I}M_I^{\overline{J}} + (\underline{y_J} - \underline{x_I}M_I^J)(M_{\overline{I}}^J)^{-1}M_{\overline{I}}^{\overline{J}}$. Tyto dva vztahy nám přesně určují hledané zobrazení φ.

Že φ je R-modulovým homomorfismem ověříme přímým výpočtem. Mějme vektory $(\underline{a},\underline{b})$ a $(\underline{c},\underline{d})$ takové, že $\underline{b} = \underline{a}M$ a $\underline{d} = \underline{c}M$. Pak

$$\varphi((\underline{a},\underline{b}) + (\underline{c},\underline{d})) = \varphi((\underline{a}+\underline{c},\underline{b}+\underline{d})) =$$

$$((\underline{b_J}+\underline{d_J}-(\underline{a_I}+\underline{c_I})M_I^J)(M_{\overline{I}}^J)^{-1},(\underline{a_I}+\underline{c_I})M_I^{\overline{J}}+(\underline{b_J}+\underline{d_J}-(\underline{a_I}+\underline{c_I})M_I^J)(M_{\overline{I}}^J)^{-1}M_{\overline{I}}^{\overline{J}}) =$$

$$((\underline{b_J} - \underline{a_I}M_I^J)(M_{\overline{I}}^J)^{-1} + (\underline{d_J} - \underline{c_I}M_I^J)(M_{\overline{I}}^J)^{-1},$$

$$\underline{a_I}M_I^{\overline{J}} + (\underline{b_J} - \underline{a_I}M_I^J)(M_{\overline{I}}^J)^{-1}M_{\overline{I}}^{\overline{J}} + \underline{c_I}M_I^{\overline{J}} + (\underline{d_J} - \underline{c_I}M_I^J)(M_{\overline{I}}^J)^{-1}M_{\overline{I}}^{\overline{J}}) =$$

$$((\underline{b_J} - \underline{a_I}M_I^J)(M_{\overline{I}}^J)^{-1},\underline{a_I}M_I^{\overline{J}} + (\underline{b_J} - \underline{a_I}M_I^J)(M_{\overline{I}}^J)^{-1}M_{\overline{I}}^{\overline{J}})+$$

$$((\underline{d_J} - \underline{c_I}M_I^J)(M_{\overline{I}}^J)^{-1},\underline{c_I}M_I^{\overline{J}} + (\underline{d_J} - \underline{c_I}M_I^J)(M_{\overline{I}}^J)^{-1}M_{\overline{I}}^{\overline{J}}) = \varphi((\underline{a},\underline{b})) + \varphi((\underline{c},\underline{d})).$$

Že $r\varphi((\underline{a},\underline{b})) = \varphi(r(\underline{a},\underline{b}))$ pro $r \in R$ ověříme podobně. Zbývá už jen ukázat, že φ je bijekcí. To snadno vidíme z toho, že inverzní zobrazení k φ dostaneme stejným způsobem, jako jsme dostali φ, akorát I zaměníme za $\overline{I}$ a J zaměníme za $\overline{J}$. □

Předchozí lemma nám říká, že u zobrazení daných hyper-invertibilními maticemi můžeme libovolnou množinu n vstupů nebo výstupů dopočítat ze zbylých n vstupů a výstupů. Díky této vlastnosti budeme moci efektivně generovat náhodné části tajemství, aniž bychom riskovali, že nastane chyba.

3. Konkrétní protokol

3.1 Idea protokolu

Základ tohoto protokolu pochází z Beerliová-Trubíniová a Hirt (2008), kde protokol pracuje nad konečnými tělesy. Nad $\mathbb{Z}/p^k\mathbb{Z}$ jsme protokol převedli pomocí úprav z Abspoel a kol. (2019). V této práci slouží protokol především jako příklad využití vybudované teorie. Čtenář by měl tedy pouze pochopit k čemu jednotlivé části slouží a kde se vybudovaná teorie používá. Důkazy korektnosti, bezpečnosti a časové náročnosti jednotlivých algoritmů zde uvádět nebudeme. Tyto důkazy může čtenář najít v již zmíněných článcích Beerliová-Trubíniová a Hirt (2008); Abspoel a kol. (2019).

Nejdříve si protokol popíšeme zjednodušeně, ať máme představu, k čemu budou jednotlivé části protokolu sloužit, a zároveň abychom si vysvětlili základní značení. Začneme tím, co předpokládáme ještě předtím, než protokol začne. Máme množinu *účastníků* $\mathcal{P} = \{P_1,...,P_n\}$, $n \in \mathbb{N}$ a $t \in \mathbb{N} \cup \{0\}$ bude počet účastníků, kteří jsou zároveň *útočníky* (o účastnících, kteří nejsou útočníky, budeme říkat, že jsou *poctiví*). Pro tento protokol předpokládáme, že $t < \frac{n}{3}$. Všichni účastníci budou spolupracovat na výpočtu výstupu předem známé funkce $\mathcal{F}$, která se bude skládat ze součtů, součinů, vstupů, výstupů a výpočtů náhodných hodnot. Tyto operace nazveme *brány*, takže máme součtové, součinové, vstupní, výstupní a náhodné brány. Počet všech bran je předem známý, takže si jejich počty postupně označíme c_A, c_M, c_I, c_O, c_R (z anglického Addition, Multiplication, Input, Output, Random). Vstupy funkce budou zadávány *uživateli*, jejichž množinu označíme $\mathcal{U}$. Tito uživatelé pouze zadají vstup a očekávají výstup, na výpočtu se nepodílejí.

Dále budeme znát předem nalezenou výjimečnou posloupnost $\alpha_0,\alpha_1,...,\alpha_n \in R$ nad zvoleným Galoisovým okruhem $R = GR(p^k,d)$. Hodnoty $\alpha_1,...,\alpha_n$ budou přiřazeny jednotlivým účastníkům $P_1,...,P_n$ a α_0 bude přiřazeno sdílenému tajemství. To znamená, že pro nějaký polynom f bude účastník P_i znát hodnotu $f(\alpha_i) \in R, i \in \{1,...,n\}$ a $f(\alpha_0) \in R$ bude sdíleným tajemstvím, které účastníci neznají, a budeme ho ve většině případů značit x_0 (podobně jsme prvky vyjímečné posloupnosti přiřazovali v Algoritmu 1). Sdíleným tajemstvím nebude vždy pouze utajený vstup od uživatelů, ale může to být jakákoliv hodnota, kterou v průběhu protokolu potřebujeme tajně sdílet mezi účastníky. Všechny prvky výjimečné posloupnosti a komu náleží budou veřejné informace, takže je zná každý z $\mathcal{P} \cup \mathcal{U}$. Dále $C_m = \{(f(\alpha_0),...,f(\alpha_n)) \mid f \in R[x]_{\leq m}\}$ pro $m \in \mathbb{N}, m \leq n$ je R-modul vektorů částí tajemství, takže *vektor částí tajemství* je libovolný prvek C_m.

Protokol se dělí na Offline a Online fázi. Offline fáze se provádí dříve, než začnou uživatelé zadávat vstupy. Slouží především k předpočítaní některých pomocných hodnot, pro které není potřeba znát vstupy, a k vyřazení účastníků tak, aby se snížil poměr útočníků k celkovému počtu účastníků. Fáze začne tím, že každý účastník se stane *spokojeným*. Že je účastník spokojený znamená, že zatím nedetekoval žádnou chybnou informaci od útočníků. Jakmile chybnou informaci účastník detekuje, stane se *nespokojeným*. Pokud se kdykoliv v průběhu protokolu stane některý z účastníků nespokojeným, nemůže výpočet pokračovat, dokud nespokojenost nevyřešíme. Dalším krokem Offline fáze je generování pomocných

hodnot pro každou součinovou, vstupní a náhodnou bránu (pro náhodné brány přímo spočítáme náhodné hodnoty, které poté v Online fázi použijeme). Pokud při generování těchto hodnot nastane chyba, detekujeme ji a poté zjistíme, kde nastala. Díky tomu najdeme dvojici účastníků, kde alespoň jeden účastník je zároveň útočníkem, a tuto dvojici vyřadíme. Toto je výhodné, jelikož $t < \frac{n}{3}$, takže se nám poměr útočníků k poctivým účastníkům vždy zlepší. Celý tento postup (i s nastavením spokojenosti všech účastníků) několikrát opakujeme a na konci tedy máme vygenerované všechny pomocné hodnoty a také máme zredukovanou množinu účastníků, kterou označíme $\mathcal{P}' = \{P_{i_1},...,P_{i_{n'}}\} \subseteq P, n' \in \mathbb{N}$. Zároveň se nám tedy počet útočníků sníží na $t' \in \mathbb{N} \cup \{0\}$. Abychom měli jistotu, že útočníci nemohou změnit tolik hodnot, aby tajemství sdílené pomocí polynomu stupně t změnili (taková budeme používat nejčastěji), musí platit $t < n' - 2t'$. Toto za našeho předpokladu $t < \frac{n}{3}$ platí, protože $n - n' \leq 2(t - t') < \frac{2n}{3} - 2t'$, takže $n' - 2t' > \frac{n}{3} > t$. Navíc si označíme $T = n' - 2t'$.

V Online fázi řešíme, co se má dít pro jednotlivé brány zvlášť. Na vstupní bráně chce některý z uživatelů tajně sdílet jeho vstup mezi všechny účastníky z $\mathcal{P}'$. Uživatel obdrží od každého účastníka jeho část (tajemství) pomocné hodnoty, kterou pro danou vstupní bránu každý účastník získal v Offline fázi. Pomocí těchto částí uživatel zrekonstruuje samotnou pomocnou hodnotu, kterou odečte od svého vstupu a výsledek sdělí všem účastníkům. Ti pak samostatně k výsledku přičtou svou část pomocné hodnoty, čímž každý získá vlastní část tajemství (vstupu od uživatele). Na součtové bráně chceme sečíst nějaké dvě hodnoty, které jsou sdíleny mezi účastníky. Jelikož ale pracujeme s polynomy a hodnotami polynomů v určitých bodech, stačí, aby každý účastník sečetl svoje části tajemství těchto dvou hodnot a dostaneme správný výsledek. Na náhodné bráně jednoduše vezmeme náhodnou hodnotu, kterou jsme pro danou bránu spočítali v Offline fázi. Součinová brána je ze všech nejsložitější, protože součin dvou polynomů stupně t nám může dát polynom stupně až $2t$, ale my chceme, aby pro interpolaci stačilo stále pouze t hodnot. K tomu, aby toto bylo možné, se využijí předpočítané pomocné hodnoty z Offline fáze. Na výstupní bráně chceme poslat určitý výstup konkrétnímu uživateli, což se provede jednoduše tak, že mu každý účastník pošle svou část tajemství tohoto výstupu a uživatel si pomocí nich výstup zrekonstruuje.

3.2 Rekonstrukce tajemství

Definice Pro Galoisův okruh $R = GR(p^k,d)$, $\mathcal{P}'$ a t' (definovány jako výše) řekneme, že jsme tajemství x_0 schopni zrekonstruovat *robustně*, pokud existuje právě jeden polynom $f \in R[x]_{\leq t}$ takový, že pro vektor částí tajemství $x = (x_0,x_{i_1},...,x_{i_{n'}})$ platí $|\{P_{i_j} \in \mathcal{P}' \mid f(\alpha_{i_j}) = x_{i_j}, j \in \{1,...,n'\}\}| \geq n' - t'$ a zároveň $f(\alpha_0) = x_0$.

Aby se dalo tajemství zrekonstruovat robustně, je pro náš protokol samozřejmě nutné. Pokud to ale možné je, jak to provést? Chceme algoritmus, který jako vstup dostane vektor $(x_1,...,x_n) \in (R \cup \{\perp\})^n$ ($\perp$ bude značit, že na dané pozici jsme našli chybu), pro který algoritmus najde $f \in R[x]_{\leq t}$ takový, že $x_i = f(\alpha_i)$ pro $i = 1,...,n$ až na maximálně $\lfloor \frac{n-t-1}{2} \rfloor$ pozic. Pokud by R bylo konečné těleso, byl by toto přesně problém dekódování lineárních kódů, kde jako kód bychom

měli R-modul C_m. Přesněji by C_m byl v takovém případě takzvaným *zobecněným Reedovým-Solomoným kódem*.

Dekódování zobecněných Reedových-Solomonových kódů je sám o sobě problém hodný práce většího rozsahu, než by měla být tato. Budeme tedy předpokládat, že již máme algoritmus pro dékodování Reedových-Solomonových kódů nad $\mathbb{F}_{p^d}$ a pomocí něj získáme algoritmus pro C_m nad Galoisovým okruhem $R = GR(p^k,d)$.

Z Poznámky 8 víme, že všechna $a \in R$ lze jednoznačně psát jako

$$a = a_0 + a_1 p + a_2 p^2 + ... + a_{k-1} p^{k-1},$$

kde $a_0,...,a_n \in \mathcal{I} = \{0,1,\xi,...,\xi^{p^d-2}\}$. Z toho plyne, že všechny $f(x) \in R[x]_{\leq t}$ můžeme také jednoznačně psát jako $f(x) = f_0(x) + pf_1(x) + ... + p^{k-1}f_{k-1}(x)$, kde $f_0(x),...,f_{k-1}(x) \in \mathcal{I}[x]_{\leq t}$. Navíc pro všechna $j = 1,...,k-1$ máme

$$f(\alpha_i) \equiv \sum_{\ell=0}^{j-1} p^\ell f_\ell(\alpha_i) \pmod{p^j}.$$

Dále nechť $\beta_i = \pi(\alpha_i) \in \mathbb{F}_{p^d}$ pro $i = 1,...,n$, kde π je redukce modulo p z Poznámky 7. Jelikož je π homomorfismus a prvky α_i tvoří výjimečnou posloupnost, budou β_i po dvou různá. Všimněme si navíc, že zúžení $\pi \restriction_{\mathcal{I}}$ nám dává bijekci mezi $\mathcal{I}$ a $\mathbb{F}_{p^d}$, takže π^{-1} můžeme chápat jako dobře definované zobrazení do $\mathcal{I}$.

Algoritmus 2 (Dekódování R.-S. kódů nad Galoisovým okruhem R)

- *Vstup:* $\underline{x} = (x_1,...,x_n) \in (R \cup \{\perp\})^n$.
- $\underline{y} \leftarrow \underline{x}$ a poté pro každé $i = 0,...,k-1$ provedeme:
  1. $\underline{y} \leftarrow \pi(\underline{y}/p^i)$ (po prvcích $\underline{y}$).
  2. Použijeme dekódovací algoritmus pro tělesa na vstup $\underline{y}$ a označíme si jeho výstup $\overline{f_i}(x)$. Nechť $f_i(x) = \pi^{-1}(\overline{f_i}(x)) \in \mathcal{I}[x]_{\leq t}$.
  3. Nechť $t_j = \sum_{\ell=0}^{i} p^\ell f_\ell(\alpha_j)$ pro $j = 1,...,n$ a $\underline{y} \leftarrow (x_1 - t_1,...,x_n - t_n)$.
  4. Pokud pro některé j není $x_j - t_j$ dělitelné p^{i+1}, nastavíme j-tý prvek $\underline{y}$ na $\perp$.
- *Výstup:* $f(x) = f_0(x) + pf_1(x) + ... + p^{k-1}f_{k-1}(x)$.

Abychom mohli s prvky, které jsme nastavili na $\perp$ pracovat, budeme za ně jednoduše dosazovat náhodné prvky. Protože $\perp$ značí chybný prvek, nezáleží na tom, jestli za něj budeme brát jiný (náhodný) chybný prvek, nebo původní chybný prvek.

Tvrzení 11 (Abspoel a kol. (2019) Theorem 4) Algoritmus 2 opraví až $\lfloor \frac{n-t-1}{2} \rfloor$ chyb s k voláními dekódovacího algoritmu nad $\mathbb{F}_{p^d}$.

Definice Řekneme, že pro $D \in \mathbb{N} \cup \{0\}$ je hodnota *D-sdílena* mezi účastníky z $\mathcal{P}'$, pokud každý účastník $P_{i_j} \in \mathcal{P}'$ drží část tajemství x_{i_j} tajemství x_0 takovou, že existuje polynom f stupně D, pro který platí $f(\alpha_0) = x_0$ a $f(\alpha_{i_j}) = x_{i_j}$ pro všechny $P_{i_j} \in \mathcal{P}'$. Vektor částí tajemství $(x_0,x_{i_1},...,x_{i_{n'}})$ pro takový polynom f se nazývá *vektor D-sdílení* x_0 a značí se $[x_0]_D$.

V následujících algoritmech pro privátní a veřejnou rekonstrukci budeme pro nalezení konkrétních polynomů využívat Algoritmus 2. Pro privátní rekonstrukci je navíc podstatné, který účastník nebo uživatel ji provádí, protože on jediný bude znát zrekonstruovanou hodnotu. Proto je uveden jako jeden ze vstupů algoritmu.

Algoritmus 3 (Privátní rekonstrukce)

- *Vstup:* $X \in (\mathcal{P} \cup \mathcal{U})$,$D$,$[s_0]_D$.
- Každý účastník z $\mathcal{P}'$ pošle svou část tajemství s_i z tajemství s_0 účastníkovi/uživateli X.
- Pokud existuje polynom $f(x) \in R[x]_{\leq D}$ takový, že alespoň $D + t' + 1$ z obdržených částí tajemství splňuje $f(\alpha_i) = s_i$, pak X spočítá tajemství $s_0 = f(\alpha_0)$. Jinak se stane nespokojeným.
- *Výstup:* X získá prvek s_0 (pokud nenastane chyba) a poté veřejně oznámí, jestli je spokojený nebo nespokojený.

Tvrzení 12 (Beerliová-Trubíniová a Hirt (2008) Lemma 3) Pro $D < n' - 2t'$ Algortimus 3 robustně zrekonstruuje $[s_0]_D$ vůči X.

Algoritmus 4 (Veřejná rekonstrukce)

- *Vstup:* $D, [s_1]_D,...,[s_T]_D$ (*Připomenutí:* $T = n' - 2t'$).
- Pro každé $j = 1,...,n'$ spočítají účastníci z $\mathcal{P}'$ lokálně (tím myslíme, že každý účastník zde provádí výpočet pouze s jím drženými hodnotami a nic mezi sebou účastníci nesdílejí) $[u_j]_D$ jako

$$[u_j]_D = [s_1]_D + \beta_j[s_2]_D + \beta_j^2[s_3]_D... + \beta_j^{T-1}[s_T]_D.$$

  Zde $\beta_1,...,\beta_{n'}$ je výjimečná posloupnost a uvažujeme $s_1,...,s_T$ jako koeficienty polynomu stupně $T-1$. Hodnoty $u_1,...,u_{n'}$ jsou tedy vyhodnoceními tohoto polynomu v bodech $\beta_1,...,\beta_{n'}$.
- Pro každého $P_{i_j} \in \mathcal{P}'$ zrekonstruujeme $[u_{i_j}]_D$ vůči P_{i_j} pomocí Algoritmu 3 (vstup je tedy P_{i_j},D,$[u_{i_j}]_D$).
- Každý $P_{i_j} \in \mathcal{P}'$ pošle u_{i_j} všem ostatním účastníkům z $\mathcal{P}'$, nebo oznámí, že je nespokojený, pokud v předchozím kroku detekoval chybu.
- Pro každého $P_{i_j} \in \mathcal{P}'$: Pokud P_{i_j} obdržel z předchozího kroku alespoň $T+t'$ hodnot, které všechny leží na nějakém polynomu stupně $T-1$, pak pomocí libovolných T z nich spočítá $s_1,...,s_T$. Jinak se stane nespokojeným.
- *Výstup:* Každý účastník získá prvky $s_1,...,s_T$ (pokud nenastane chyba) a poté veřejně oznámí, jestli je spokojený nebo nespokojený.

Tvrzení 13 (Beerliová-Trubíniová a Hirt (2008) Lemma 4) Pro $D < n' - 2t'$ Algortimus 4 robustně zrekonstruuje $[s_1]_D,...,[s_T]_D$ vůči všem účastníkům z $\mathcal{P}'$.

3.3 Sdílení a generování sdílení

Následující jednoduchý algoritmus pro sdílení prvku využívá Algoritmus 1. Stejně jako u privátní rekonstrukce je i zde podstatné, který účastník nebo uživatel prvek sdílí, protože on jediný bude původní prvek znát, ostatní znají jen obdrženou část tajemství.

Algoritmus 5 (Sdílení)

- *Vstup:* $X \in (\mathcal{P} \cup \mathcal{U})$,$s_0$,$D$.
- X zvolí náhodný polynom $f(x) \in R[x]_{\leq D}$ takový, že $s_0 = f(\alpha_0)$ a pošle $s_{i_j} = f(\alpha_{i_j})$ každému $P_{i_j} \in \mathcal{P}'$.
- *Výstup:* Vektor D-sdílení $[s_0]_D$.

Definice Řekneme, že pro D,$D' \in \mathbb{N} \cup \{0\}$ je hodnota (D,D')-*sdílena* mezi účastníky z $\mathcal{P}'$, pokud je zároveň D-sdílena i D'-sdílena. *Vektor* (D,D')-*sdílení* x_0 budeme značit $[x_0]_{D,D'}$ a bude to dvojice vektorů $((x_0,x_{i_1},...,x_{i_{n'}}),(y_0,y_{i_1},...,y_{i_{n'}}))$, kde $(x_0,x_{i_1},...,x_{i_{n'}})$ je vektor D-sdílení x_0, $(y_0,y_{i_1},...,y_{i_{n'}})$ je vektor D'-sdílení y_0 a $x_0 = y_0$.

Následující algoritmus pro generování náhodných vektorů (D,D')-sdílení buď vygeneruje T nezávislých náhodných hodnot $r_1,...,r_T$, které budou (D,D')-sdíleny mezi účastníky z P', nebo selže, protože bude alespoň jeden poctivý účastník nespokojený. S (D,D')-sdílenými hodnotami potřebujeme pracovat především kvůli součinu a generování pomocných hodnot pro součin.

Zde zároveň využijeme hyper-invertibilní matice, konkrétně předem zvolenou hyper-invertibilní matici M typu $n' \times n'$. Samozřejmě n' předem neznáme a může se nám dokonce změnit mezi jednotlivými voláními tohoto algoritmu. Můžeme si ale předem zvolit hyper-invertibilní matici typu $n \times n$, ze které poté vybíráme podmatice. Z definice hyper-invertibilní matice ihned vidíme, že i tyto podmatice budou hyper-invertibilní. Tuto matici zkonstruujeme podle Tvrzení 9.

V algortimu každý účastník dvakrát sdílí jím vybranou náhodnou hodnotu s_j, takže hodnota s_j tedy bude (D,D')-sdílena (v celé této části $j = 1,...,n'$). Poté účastníci spočítají vektory (D,D')-sdílení r_j jako

$$([r_1]_{D,D'},...,[r_{n'}]_{D,D'}) = ([s_1]_{D,D'},...,[s_{n'}]_{D,D'})M.$$

Tento zápis ale není přesný. My sice chceme v podstatě spočítat $(r_1,...,r_{n'}) = (s_1,...,s_{n'})M$, ale žádný účastník nezná všechna s_j, takže toto nikdo spočítat nemůže. Počítáme tedy s vektory (D,D')-sdílení s_j. To znamená, že pro každé s_j máme vektor D-sdílení $(s_j^0,s_j^1,...,s_j^{n'})$ a vektor D'-sdílení $(\tilde{s}_j^0,\tilde{s}_j^1,...,\tilde{s}_j^{n'})$, ze kterých každý účastník $P_{i_j} \in \mathcal{P}'$ zná $s_1^j, s_2^j,...,s_{n'}^j, \tilde{s}_1^j, \tilde{s}_2^j,...,\tilde{s}_{n'}^j$ a navíc zná $s_j^0 = \tilde{s}_j^0 = s_j$, protože s_j vybíral a sdílel ostatním, ale s touto hodnotou žádný účastník dál pracovat nebude, takže na ni můžeme zapomenout (případně ji samozřejmě můžeme zrekonstruovat, pokud by to byla potřeba). Pomocí ostatních hodnot účastník P_{i_j} spočítá $(r_1^j,...,r_{n'}^j) = (s_1^j,...,s_{n'}^j)M$ a $(\tilde{r}_1^j,...,\tilde{r}_{n'}^j) = (\tilde{s}_1^j,...,\tilde{s}_{n'}^j)M$. Jakmile toto provede každý účastník, získáme pro každé j celkem $r_j^1,...,r_j^{n'},\tilde{r}_j^1,...,\tilde{r}_j^{n'}$. Toto jsou přesně části dvou tajemství r_j^0 a $\tilde{r}_j^0$, která sice neznáme, ale jsme je schopni zrekonstruovat, pokud kdokoliv obdrží od všech účastníků jejich části tajemství. Celkem

tedy opět pro každé j dostaneme $(r_j^0,r_j^1,...,r_j^{n'})$,$(\tilde{r}_j^0,\tilde{r}_j^1,...,\tilde{r}_j^{n'})$, pro které musí platit $r_j^0 = \tilde{r}_j^0 = r_j$ a navíc $(r_j^0,r_j^1,...,r_j^{n'})$ musí být vektor D-sdílení r_j a $(\tilde{r}_j^0,\tilde{r}_j^1,...,\tilde{r}_j^{n'})$ musí být vektor D'-sdílení r_j. Tím jsme přesně získali hledané vektory (D,D')-sdílení $[r_1]_{D,D'},...,[r_{n'}]_{D,D'}$.

Příklad Pro lepší představu si ještě ukážeme, jak předchozí výpočet vypadá z pohledu jednoho z účastníků. Buďme například účastníkem P_{i_3}. Na začátku náhodně zvolíme s_3, najdeme polynomy $f(x) \in R[x]_{\leq D}, g(x) \in R[x]_{\leq D'}$ takové, že $s_3 = f(\alpha_0) = g(\alpha_0)$ a každému účastníkovi $P_{i_j}, j = 1,...,n'$ pošleme $f(\alpha_j) = s_3^j$ a $g(\alpha_j) = \tilde{s}_3^j$ (pro $j = 3$ „posíláme“ sobě). Od všech účastníků (včetně od sebe z předchozího kroku) obdržíme celkem $s_1^3,...,s_{n'}^3, \tilde{s}_1^3,...,\tilde{s}_{n'}^3$ a spočítáme $(r_1^3,...,r_{n'}^3) = (s_1^3,...,s_{n'}^3)M$ a $(\tilde{r}_1^3,...,\tilde{r}_{n'}^3) = (\tilde{s}_1^3,...,\tilde{s}_{n'}^3)M$. Tím jsme hotovi. Pak už jen posíláme konkrétní r_k^3 a $\tilde{r}_k^3$, když je potřeba zrekonstruovat r_k pro nějaké k (například pro kontrolu, která po tomto kroku algoritmu následuje).

Z hodnot r_j se jich $2t'$ zrekonstruuje vůči různým účastníkům, kteří je ověří. Zbylých $n' - 2t' = T$ hodnot se odešle na výstup. Tímto zajistíme alespoň n' správných (D,D')-sdílených hodnot, protože máme $n' - t'$ správných hodnot s_j od poctivých účastníků a alespoň t' hodnot r_j poctiví účastníci ověří (pokud by některá byla chybná, daný poctivý účastník by se stal nespokojeným a algoritmus by selhal), protože z $2t'$ ověřovaných hodnot jich může být ověřováno útočníky nejvýše t' (počet útočníků). Díky hyper-invertibilitě M tedy můžeme podle Lemma 10 zbylých n' (D,D')-sdílených hodnot jednoznačně dopočítat, takže získáme všech $2n'$ správných (D,D')-sdílených hodnot.

Algoritmus 6 (Náhodné vektory (D,D')-sdílení)

- *Vstup:* D,D'.
- *Sdílení:* Každý účastník $P_{i_j} \in \mathcal{P}'$ vybere náhodně s_j a (paralelně) použije Algoritmus 5 na vstupy P_{i_j},s_j,D a P_{i_j},s_j,D', čímž získáme $[s_j]_{D,D'}$.
- *Použití hyper-invertibilní matice:* Účastníci z $\mathcal{P}'$ lokálně spočítají

  $$([r_1]_{D,D'},...,[r_{n'}]_{D,D'}) = ([s_1]_{D,D'},...,[s_{n'}]_{D,D'})M$$

  (tak jak bylo popsáno výše).
- *Kontrola:* Pro $j = T+1,...,n'$ pošle účastníkovi P_{i_j} každý účastník z $\mathcal{P}'$ vlastní část tajemství $[r_j]_{D,D'}$. P_{i_j} pak zkontroluje, že pomocí n' obdržených částí tajemství opravdu získá vektor (D,D')-sdílení. Neboli že opravdu existuje polynom $g(x)$ stupně D, na kterém leží všechny hodnoty pro vektor D-sdílení, že existuje polynom $g'(x)$ stupně D', na kterém leží všechny hodnoty pro vektor D'-sdílení a že $g(\alpha_0) = g'(\alpha_0)$. Pokud kterákoliv z těchto tří podmínek neplatí, P_{i_j} bude nespokojený a algoritmus selže.
- *Výstup:* Zbylých T vektorů (D,D')-sdílení, neboli $[r_1]_{D,D'},...,[r_T]_{D,D'}$.

Tvrzení 14 (Beerliová-Trubíniová a Hirt (2008) Lemma 5) Pokud Algoritmus 6 neselže (neboli všichni poctiví účastníci zůstanou spokojeni), získáme na výstupu $T = n' - 2t'$ korektních náhodných (D,D')-sdílených hodnot, které útočníci nemohou znát.

Nyní pomocí předchozího algoritmu získáme algoritmus pro generování trojic pomocných hodnot. Pomocné hodnoty potřebujeme pro každou součinovou, vstupní a náhodnou bránu. Trojice jsou to pouze kvůli součinovým branám, pro ostatní nám stačí jedna pomocná hodnota, takže u vstupních a náhodných bran z trojice vezmeme jen tu první a zbylé dvě ignorujeme.

Idea algoritmu je taková, že pomocí tří volání Algortimu 6 vygenerujeme $[a_1]_{t,t'},...,[a_T]_{t,t'},[b_1]_{t,t'},...,[b_T]_{t,t'}$ a $[r_1]_{t,2t'},...,[r_T]_{t,2t'}$ a poté pro každý pár a_k,b_k spočítáme vektor t-sdílení součinu $c_k = a_k \cdot b_k$ tak, že lokálně spočítaný vektor $2t'$ sdílení $[c_k]_{2t'} = [a_k]_{t'} \cdot [b_k]_{t'}$ zredukujeme na vektor t-sdílení $[c_k]_t$ pomocí $[r_k]_t$ a $[r_k]_{2t'}$. A to tak, že spočítáme $[d_k]_{2t'} = [c_k]_{2t'} - [r_k]_{2t'}$ a veřejně zrekonstruujeme všechna d_k, což můžeme, protože díky tomu, že jsou r_k náhodná, účastníci pomocí d_k o c_k nic nezjistí. Pak už si pomocí d_k a své části $[r_k]_t$ může lokálně každý účastník spočítat vlastní část $[c_k]_t$.

Algoritmus 7 (Generování trojic)

- *Generování sdílení:* Paralelně třikrát použijeme Algoritmus 6 na vstupy (t,t'),(t,t') a $(t,2t')$, čímž získáme $[a_1]_{t,t'},...,[a_T]_{t,t'},[b_1]_{t,t'},...,[b_T]_{t,t'}$ a $[r_1]_{t,2t'},...,[r_T]_{t,2t'}$.
- *Násobení:*
  1. Pro $k = 1,...,T$ účastníci z $\mathcal{P}'$ lokálně spočítají $2t'$-sdílení $[c_k]_{2t'}$ součinu $c_k = a_k b_k$ jako $[c_k]_{2t'} = [a_k]_{t'}[b_k]_{t'}$ (neboli každý účastník spočítá součin jím držených částí).
  2. Pro $k = 1,...,T$ účastníci z $\mathcal{P}'$ lokálně spočítají $2t'$-sdílení rozdílu $[d_k]_{2t'} = [c_k]_{2t'} - [r_k]_{2t'}$.
  3. Zavoláme Algoritmus 4 na vstup $2t'$,$[d_1]_{2t'},...,[d_T]_{2t'}$, abychom zrekonstruovali $d_1,...,d_T$ vůči všem účastníkům z $\mathcal{P}'$.
  4. Pro $k = 1,...,T$ účastníci z $\mathcal{P}'$ lokálně spočítají t-sdílení $[c_k]_t = [r_k]_t + [d_k]_0$, kde $[d_k]_0$ značí sdílení $[d_k]_0 = (d_k,...,d_k)$.
- *Výstup:* t-sdílené trojice $([a_1]_t,[b_1]_t,[c_1]_t),...,([a_T]_t,[b_T]_t,[c_T]_t)$.

Tvrzení 15 (Beerliová-Trubíniová a Hirt (2008) Lemma 6) Pokud Algoritmus 7 neselže (neboli všichni poctiví účastníci zůstanou spokojeni), získáme na výstupu nezávislé náhodné vektory t-sdílení T trojic $([a_1]_t,[b_1]_t,[c_1]_t),...,([a_T]_t,[b_T]_t,[c_T]_t)$, kde a_k,b_k jsou nezávislé náhodné hodnoty a $c_k = a_k \cdot b_k$ pro $k = 1,...,T$.

3.4 Problém byzantských generálů

Problém byzantských generálů (Byzantine agreement/fault) je známý problém, který se nejlépe popisuje následujícím příkladem: Máme určitý počet generálů, kteří útočí na pevnost. Pokud zaútočí se svými armádami všichni generálové najednou, mají jisté vítězství. Pokud ale někteří nezaútočí, mají jistou prohru. Musí se tedy shodnout, jestli všichni zaútočí, nebo se všichni stáhnou. Problém je v tom, že někteří generálové mohou lhát. To znamená, že máme-li například celkem devět generálů, ze kterých chtějí čtyři zaútočit a čtyři se stáhnout, může devátý generál sdělit generálům, co chtějí zaútočit, že zaútočí, a generálům, co

se chtějí stáhnout, že se také stáhne. Ve výsledku si pak tedy část bude myslet, že většina volila útok, takže zaútočí, a jiná část, že většina volila proti útoku, takže se stáhne. Přesně tomuto ale chceme zabránit. Druhou částí problému je, že generálové jsou fyzicky odděleni, takže si posílají zprávy o tom, pro co volí, skrze posly. Ti mohou zprávu ztratit nebo také pozměnit.

V našem protokolu budeme předpokládat, že oba problémy již umíme vyřešit (například řešení z Berman a kol. (1992); Coan a Welch (1992)). Řešení prvního problému budeme chápat jako algoritmus a budeme mu říkat *protokol souhlasu*. Protokol souhlasu nám v naší situaci zajistí, že pokud jsou někteří účastníci spokojeni a někteří nespokojeni, dojdeme ke shodě. Neboli po spuštění protokolu souhlasu budou všichni účastníci jako celek spokojeni, nebo budou všichni účastníci nespokojeni. Řešení druhého problému zajistí, že pokud některý z účastníků má odeslat hodnotu všem ostatním účastníkům, obdrží všichni stejnou hodnotu. Neboli i kdyby účastník, který hodnotu odesílá, byl útočníkem, nemůže odeslat různým účastníkům různé hodnoty.

3.5 Offline fáze

Jak již bylo zmíněno, Offline fáze může teoreticky probíhat libovolně dlouhou dobu před Online fází, jelikož při ní využíváme pouze předem známé informace a nepotřebujeme zde znát vstupy, které nám budou později zadávat uživatelé. Hlavním cílem Offline fáze je vypočítat pomocné náhodné hodnoty, které potřebujeme pro náhodné brány a pro utajení výpočtu u vstupních a součinových bran. Cílem Offline fáze je také eliminovat účastníky tak, aby se zlepšil poměr útočníků k poctivým účastníkům.

Náhodné hodnoty generujeme pomocí Algoritmu 7. Po tom, co algoritmus proběhne, chceme dojít ke shodě, jestli jsou vygenerované hodnoty korektní. To znamená, že pokud některý účastník narazil při výpočtu na chybu, stal se nespokojeným, a to nyní oznámí ostatním účastníkům. Tímto se účastníci dozví o chybě, takže by se měli také stát nespokojenými. Nakonec chceme dojít k tomu, že jsou všichni účastníci buď nespokojení a jistě nastala chyba, nebo jsou všichni účastníci spokojení a jistě můžeme vygenerované hodnoty použít. Toto je přesně výše uvedený Problém byzantských generálů, kde se generálové musí shodnout, jestli všichni najednou zaútočí, nebo se stáhnou. Zde tedy využijeme protokol souhlasu, čímž dojdeme k požadované shodě, jestli jsou vygenerované hodnoty korektní, nebo ne.

V případě, že nastala chyba, chceme eliminovat účastníky a výpočet hodnot opakovat. Pro eliminaci účastníků musíme zjistit, kde chyba nastala. To provedeme tak, že zvolíme účastníka P_r, který od všech účastníků obdrží všechny hodnoty z předchozího výpočtu, takže může spočítat vše, co bylo během výpočtu mezi účastníky přeposíláno. Díky tomu najde zprávu, kde nastala chyba, a účastníky, kteří si tuto zprávu předali. Označme je P_i,P_j. Tito účastníci mohou s P_r nesouhlasit. Pokud nesouhlasí P_i, eliminujeme P_r a P_i. Pokud nesouhlasí P_j, eliminujeme P_r a P_j. Jinak eliminujeme P_i a P_j. Toto zajistí, že alespoň jeden ze dvou eliminovaných účastníků je útočník, protože pokud jsou P_r a P_j poctiví a P_i je útočník, ať P_i souhlasí nebo nesouhlasí, vždy ho eliminujeme, protože P_j bude s P_r jistě souhlasit, jelikož jsou oba poctiví (pokud je útočníkem pouze P_j, stejnou

úvahou zjistíme, že P_j bude vždy eliminován). Pokud je P_r útočníkem a P_i a P_j jsou poctiví (takže si P_r chybu vymyslel), P_i nebo P_j bude jistě nesouhlasit, takže vždy eliminujeme P_r. Pokud je v trojici více než jeden útočník, očividně vždy alespoň jednoho útočníka eliminujeme, i kdybychom dvojici vybrali náhodně. Pokud by byli všichni tři účastníci poctiví, tak tato situace nemůže nastat, protože by mezi P_i a P_j nemohla nastat chyba a P_r by si chybu nevymyslel.

Nyní už tedy popíšeme přímo algoritmus Offline fáze. Připomeňme si ještě, že $T = n' - 2t'$ a c_A, c_M, c_I, c_O, c_R značí postupně počet součtových, součinových, vstupních, výstupních a náhodných bran. Rozdělíme generování $c_M + c_R + c_I$ trojic pomocných hodnot na t segmentů délky $\ell = \lceil \frac{c_M + c_R + c_I}{t} \rceil$ a pro každý segment $k = 1,...,t$ provedeme:

0. Každý účastník $P_i \in \mathcal{P}'$ se stane spokojeným.
1. *Generování trojic:* Paralelně zavoláme $\lceil \frac{\ell}{T} \rceil$-krát Algoritmus 7.
2. *Detekce chyby:* Chceme dojít ke shodě, jestli je některý z účastníků nespokojený, nebo ne.
   - Každý $P_i \in \mathcal{P}'$ oznámí, jestli je spokojený nebo nespokojený, každému $P_j \in \mathcal{P}'$, který se stane nespokojeným, pokud alespoň jeden P_i tvrdí, že je nespokojený:
   - Použijeme protokol souhlasu na množině účastníků $\mathcal{P}'$. Pokud po průběhu protokolu souhlasu jsou všichni účastníci spokojeni, jsou vygenerované trojice korektní, jsou poslány na výstup a segment je hotový. Jinak nastala chyba a pokračujeme krokem 3.
3. *Lokalizace chyby:* Chceme najít $E \subseteq \mathcal{P}', |E| = 2$ takové, že alespoň jeden účastník z E je lhář:
   - Zvolíme účastníka $P_r \in \mathcal{P}'$ (například takového, který má nejmenší index r a zároveň ještě nebyl zvolen v žádném z předchozích segmentů).
   - Každý $P_i \in \mathcal{P}'$ pošle zvolenému P_r všechny jím doposud vybrané náhodné hodnoty a vše, co během výpočtu pro tento segment obdržel od ostatních účastníků.
   - Ze získaných hodnot může P_r spočítat všechny zprávy, které měly být odeslány. Ty pak porovná se zprávami, které účastníci údajně obdrželi. Poté P_r sdělí všem (l,i,j,x,x'), kde l je index zprávy, ve které měl P_i poslat x účastníkovi P_j, ale P_j tvrdí, že obdržel $x' \neq x$.
   - Podezřelí účastníci oznámí, jestli s P_r souhlasí, nebo ne. Pokud P_i nesouhlasí, zvolíme $E = \{P_r, P_i\}$, pokud P_j nesouhlasí, zvolíme $E = \{P_r, P_j\}$, jinak $E = \{P_i, P_j\}$.
4. *Eliminace účastníků:* Nastavíme $\mathcal{P}' \leftarrow \mathcal{P}' \setminus E$, $n' \leftarrow n' - 2$, $t' \leftarrow t' - 1$ a vrátíme se na krok 0 (segment opakujeme).

Tvrzení 16 (Beerliová-Trubíniová a Hirt (2008) Lemma 7) Offline fáze vygeneruje nezávislé náhodné vektory t-sdílení $c_M + c_R + c_I$ utajených trojic (a_k,b_k,c_k), kde a_k,b_k jsou nezávislé náhodné hodnoty a $c_k = a_k \cdot b_k$ pro $k = 1,...,c_M + c_R + c_I$. Navíc bude po Offline fázi poměr útočníků k poctivým účastníkům takový, že Online fáze proběhne vždy bezpečně.

3.6 Online fáze

Při Online fázi již máme připravené vše z Offline fáze a uživatelé nám začínají posílat vstupy, se kterými máme provést dané operace. Při zadávání vstupu (neboli na vstupních branách) zrekonstruujeme $[r]_t$ vůči danému uživateli, kde $[r]_t$ je první prvek z trojice pomocných náhodných hodnot, které jsme si pro každou vstupní bránu připravili v Offline fázi. Uživatel pak pomocí zrekonstruovaného r utají svůj vstup s tak, že sdělí $d = s - r$ všem účastníkům. Žádný z účastníků nemůže s zjistit, protože r je náhodné, nikdo ho nezná a každý zná pouze svoji část tajemství $[r]_t$ (zde ji značíme r_i pro účastníka P_i). Každý účastník P_i tedy může z d spočítat pouze svou část tajemství $[s]_t$ jako $s_i = d + r_i$.

U součtových bran jednoduše každý účastník sečte své části tajemství dvou hodnot, které se mají sečíst. Náhodné brány také nejsou složité. Zde pouze vezmeme první prvek z trojice pomocných náhodných hodnot, které jsme si pro každou náhodnou bránu připravili v Offline fázi. Jelikož součinové brány jsou trochu složitější, popíšeme je v dalším odstavci a zde přeskočíme k výstupním branám. U těch uživateli každý účastník pošle svou část tajemství daného výstupu, který si pak pomocí nich uživatel privátně zrekonstruuje, takže se ho nedozví nikdo jiný než on.

Díky tomu, že nám Algoritmus 4 umožňuje veřejně rekonstruovat až T hodnot najednou (kde $T = n - 2t$), součinových bran můžeme počítat až $\lfloor T/2 \rfloor$ najednou. To pro vysvětlení principu výpočtu ale není příliš důležité, takže si vysvětlíme výpočet jedné součinové brány. Máme tedy dva t-sdílené činitele $[x]_t$ a $[y]_t$ a chceme získat vektor t-sdílení $[z]_t$, kde $z = x \cdot y$. Nejdříve pomocí prvních dvou hodnot z předpřipravené trojice z Offline fáze utajíme $[x]_t$ a $[y]_t$ tak, že spočítáme $[d]_t = [x]_t - [a]_t$ a $[e]_t = [y]_t - [b]_t$ (neboli každý účastník P_i spočítá $d_i = x_i - a_i$ a $e_i = y_i - b_i$, kde tyto hodnoty značí jím držené části tajemství x,y,a,b,d,e). Protože a i b jsou náhodné, můžeme $[d]_t$ i $[e]_t$ zrekonstruovat veřejně, aniž by kdokoliv zjistil x nebo y. Nyní tedy každý účastník P_i zná konkrétní d a e, své části tajemství $[a]_t$,$[b]_t$ (opět části budeme značit a_i,b_i) a navíc svou část tajemství $[c]_t$ (označíme tuto část c_i), což je třetí hodnota z trojice pomocných hodnot z Offline fáze, o které víme, že $c = a \cdot b$. S tímto už můžeme spočítat vektor t-sdílení $[z]_t$ jako $[z]_t = [de]_0 + d[b]_t + e[a]_t + [c]_t$, kde vektor 0-sdílení $[de]_0$ je vektor (de,...,de). Předchozí výpočet bude tedy z pohledu konkrétního účastníka P_i vypadat jako $z_i = d \cdot e + d \cdot b_i + e \cdot a_i + c_i$. Že předchozí rovnost platí vidíme z následujícího výpočtu a z faktu, že pracujeme s modulovými homomorfismy

$$z = de + db + ea + c = (x - a)(y - b) + (x - a)b + (y - b)a + ab =$$

$$= xy - ay - bx + ab + xb - ab + ya - ba + ab = xy.$$

Algoritmus Online fáze pak vypadá pro jednotlivé brány následovně:

- *Vstupní brána* (Uživatel U pošle vstup s):

  1. Zrekonstruujeme vektor t-sdílení $[r]_t$ vůči U pomocí Algoritmu 3 se vstupem U,t,$[r]_t$. Toto je robustní, protože $t < n' - 2t'$.
  2. Uživatel U spočítá a sdělí všem $d = s - r$.
  3. Každý $P_i \in \mathcal{P}'$ lokálně spočítá jeho část tajemství jako $s_i = d + r_i$.

- *Součtová brána*: Každý $P_i \in \mathcal{P}'$ sečte své části tajemství hodnot, které na této konkrétní bráně máme sečíst.
- *Náhodná brána*: Vezmeme náhodný vektor t-sdílení $[r]_t$, který jsme si pro danou bránu připravili v Offline fázi.
- *Součinová brána*: Až $\lfloor T/2 \rfloor$součinových bran se vyhodnocuje najednou. Označme vektory t-sdílení činitelů jako $([x_1]_t,[y_1]_t),...,([x_{T/2}]_t,[y_{T/2}]_t)$ a k ním příslušné trojice pomocných náhodných hodnot z Offline fáze jako $([a_1]_t,[b_1]_t,[c_1]_t),...,([a_{T/2}]_t,[b_{T/2}]_t,[c_{T/2}]_t)$. Součiny $[z_1]_t,...,[z_{T/2}]_t$ jsou pak spočteny následovně:
  1. Pro $k = 1,...,T/2$ účastníci spočítají $[d_k]_t = [x_k]_t - [a_k]_t$ a $[e_k]_t = [y_k]_t - [b_k]_t$.
  2. Použijeme Algoritmus 4 na vstup $t,[d_1]_t,...,[d_{T/2}]_t,[e_1]_t,...,[e_{T/2}]_t$, abychom veřejně zrekonstruovali všech T t-sdílených hodnot $(d_1,e_1),...,(d_{T/2},e_{T/2})$. Toto je robustní, protože $t < n' - 2t'$.
  3. Pro $k = 1,...,T/2$ účastníci spočítají vektor t-sdílení součinu $[z_k]_t = [d_k e_k]_0 + d_k[b_k]_t + e_k[a_k]_t + [c_k]_t$, kde $[d_k e_k]_0$ značí vektor 0-sdílení $d_k e_k$ definované jako $[d_k e_k]_0 = (d_k e_k,...,d_k e_k)$.
- *Výstupní brána* (Výstup s uživateli U): Zavoláme Algoritmus 3 na vstup $U,t,[s]_t$.

Tvrzení 17 (Beerliová-Trubíniová a Hirt (2008) Lemma 8) Online fáze bezpečně vyhodnotí funkci s c_I vstupními, c_R náhodnými, c_M součinovými, c_A součtovými a c_O výstupními branami, pokud má k dispozici $c_M + c_R + c_I$ trojic (a_k,b_k,c_k), kde a_k,b_k jsou nezávislé náhodné hodnoty a $c_k = a_k \cdot b_k$ pro $k = 1,...,c_M + c_R + c_I$.

Celkem jsme tedy získali MPC protokol nad R, což můžeme shrnout následujícím tvrzením:

Tvrzení 18 (Abspoel a kol. (2019) Theorem 5) Existuje MPC protokol pro n účastníků nad Galoisovým okruhem $R = GR(p^k,d)$, kde $p^d \geq 2n$, který je bezpečný, pokud počet útočníků je menší než $\frac{n}{3}$.

Že musí platit $p^d \geq 2n$, znamená, že počet účastníků je ve skutečnosti omezen polovinou Lenstrovy konstanty okruhu R. Že toto musí platit vidíme z předpokladů pro konstrukci hyper-invertibilní matice (Tvrzení 9). Ve skutečnosti tedy musíme znát výjimečnou posloupnost o alespoň $2n$ prvcích. Ale prvky, které nejsou přiřazeny účastníkům nebo sdílenému tajemství, jinde než při konstrukci této matice nevyužíváme, takže pro pochopení protokolu příliš zásadní nejsou.

Získali jsme MPC protokol nad Galoisovým okruhem R, ale chceme MPC protokol nad $\mathbb{Z}/p^k\mathbb{Z}$. Jelikož je ale $\mathbb{Z}/p^k\mathbb{Z}$ podokruhem použitého Galoisova okruhu, je zde $\mathbb{Z}/p^k\mathbb{Z}$ uzavřené na prováděné operace. To znamená, že pokud jsou vstupy ze $\mathbb{Z}/p^k\mathbb{Z}$ a provedeme na nich operace nad R, dostaneme na výstupu opět prvky ze $\mathbb{Z}/p^k\mathbb{Z}$. Z bezpečnostích důvodů bychom tedy mohli ještě požadovat kontrolu, že jsou všechny vstupy opravdu ze $\mathbb{Z}/p^k\mathbb{Z}$. Takovou kontrolu provést lze, ale popsat ji by bylo dosti technické a pro tuto práci nepodstatné, takže čtenáře odkážeme na Abspoel a kol. (2019), pokud by se chtěl o kontrole dozvědět více. Touto úvahou a předchozím tvrzením jsme tedy získali následující důsledek:

Důsledek 19 (Abspoel a kol. (2019) Theorem 6) Existuje MPC protokol pro n účastníků nad $\mathbb{Z}/p^k\mathbb{Z}$, který je bezpečný, pokud počet útočníků je menší než $\frac{n}{3}$.

Závěr

Hlavním cílem práce bylo vysvětlit teoretické pozadí, na kterém stojí protokol popsaný v poslední části práce, jelikož v původních článcích se autoři zaměřují spíše na důkazy bezpečnosti a časové náročnosti a pro ostatní teorii uvádějí spíše jen idey důkazů. Tím jsme se zabývali v první části práce, kde jsme dokázali, že některá známá tvrzení, která zde potřebujeme, jako je například Čínská zbytková věta nebo věta o interpolaci, platí pro určité předpoklady i nad komutativními okruhy.

Hlavním přínosem druhé části práce byl konstruktivní důkaz existence hyperinvertibilních matic, které protokol využívá. Tento důkaz využívá většinu teorie z první části práce a původní články obsahují také spíše jen návod, jak tvrzení dokázat, než přímo formální důkaz, podobně jako tomu bylo u některých důkazů v první části.

Poslední část práce popisuje konkrétní protokol, který představuje úpravu protokolu z Beerliová-Trubíniová a Hirt (2008) nad $\mathbb{Z}/p^k\mathbb{Z}$ pomocí úvah z Abspoel a kol. (2019). Sjednocení těchto dvou článků je tedy rovněž jeden z přínosů této práce. Účelem poslední části práce je spíše vysvětlit, jak samotný protokol pracuje a kde je použita vybudovaná teorie. Přesný důkaz správného fungování protokolu by byl nad rámec zadání práce. Navíc je v této práci rozšířené značení, které bylo v původních článcích místy zavádějící nebo neúplné.

Práci by bylo možné rozšířit například o některé nástroje, pro které jsme se odkázali na externí zdroje, o popis a formální důkazy korektnosti, bezpečnosti a časové náročnosti částí uvedeného protokolu nebo o implementaci protokolu, na které by bylo možné teorii prakticky ověřit. Práce by ale se zmíněnými rozšířeními značně přesáhla maximální rozsah ze zadání.

Seznam použité literatury

Abspoel, M., Cramer, R., Damgård, I., Escudero, D. a Yuan, C. (2019). Efficient information-theoretic secure multiparty computation over $\mathbb{Z}/p^k\mathbb{Z}$ via Galois Rings. Cryptology ePrint Archive, Paper 2019/872. URL https://eprint.iacr.org/2019/872.

Barto, L. a Tůma, J. (2023). *Lineární algebra*. Vznikající elektronická skripta. URL https://www2.karlin.mff.cuni.cz/~stanovsk/vyuka/LinAlg/skripta22.pdf. z 17. února 2023.

Beerliová-Trubíniová, Z. a Hirt, M. (2008). Perfectly-secure mpc with linear communication complexity. In Canetti, R., editor, *Theory of Cryptography*, pages 213–230, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg. ISBN 978-3-540-78524-8.

Berman, P., Garay, J. A. a Perry, K. J. (1992). *Bit Optimal Distributed Consensus*, pages 313–321. Springer US, Boston, MA. ISBN 978-1-4615-3422-8. doi: 10.1007/978-1-4615-3422-8_27. URL https://doi.org/10.1007/978-1-4615-3422-8_27.

Coan, B. A. a Welch, J. L. (1992). Modular construction of a byzantine agreement protocol with optimal message bit complexity. *Information and Computation*, **97**(1), 61–85. ISSN 0890-5401. doi: https://doi.org/10.1016/0890-5401(92)90004-Y. URL https://www.sciencedirect.com/science/article/pii/089054019290004Y.

Kala, V. (2023). *Úvod do komutativní algebry*. Elektronická skripta. URL http://karlin.mff.cuni.cz/~kala/files/UKA22.pdf. z 5. ledna 2023.

Stanovský, D. (2022). *Algebra*. Elektronická skripta. URL https://www2.karlin.mff.cuni.cz/~stanovsk/vyuka/2122/algebra22.pdf. z 17. června 2022.

Stanovský, D. a Barto, L. (2017). *Počítačová algebra*. Druhé upravené vydání. Matfyzpress, Praha. ISBN 978-80-7378-340-2.

Wan, Z. (2003). *Lectures on Finite Fields and Galois Rings*. Lectures on Finite Fields and Galois Rings. World Scientific. ISBN 9789812385703. URL https://books.google.cz/books?id=uCSVbYMljNIC.