PREHOD MED BAZAMI V KONČNIH OBSEGIH

Patrik Mlekuž

10. februar 2005

Kazalo

1 UVOD

1.1 Uporabnost abstraktnih teorij v vsakdanjem življenju

Nekatere matematične teorije, kot so abstraktna algebra, teorija števil in končni obsegi, so v zadnjem desetletju postale zelo pomembne tudi v vsakdanjem življenju. Glavni "krivec" za to je široka uporaba osebnih računalnikov, ki so med seboj povezani prek svetovnega spleta, zaradi česar se je pojavila velika potreba po varnosti podatkov ter prenosu informacij brez napak.

Le kako naj beremo informacije s CD-ja, ne da bi se pri tem kje zmotili? Podatke na CD-ju je potrebno zakodirati tako, da bomo naključno napako med branjem odkrili in jo znali tudi popraviti. Rešitev tega problema je recimo uporaba Reed-Solomonove kode, ki je najbolj pogost tip kod za popravljanje napak.

Tudi varovanje podatkov postaja vedno bolj pomembno. Dandanes praktično vsaka banka ponuja svoje storitve tudi preko interneta, zato je potrebno vzpostaviti varno povezavo med klientom in banko. To pomeni, da ni mogoče, da bi neka tretja oseba prisluškovala komunikaciji med njima. Še bolj pa je pomembno to, da se neka oseba ne bi mogla lažno identificirati in izvrševati nepooblaščenih transakcij. Tu pa za varnost poskrbi kriptografija.

Kje pa se vidi uporabnost končnih obsegov?

Končni obsegi so matematična teorija, ki je osnova za javno kriptografijo in teorijo kodiranja. Natančneje, z njimi se veliko ukvarjamo pri kriptosistemih z javnimi ključi, npr. kriptosistemi z eliptičnimi krivuljami, ki temeljijo na težavnosti reševanja problema diskretnega logaritma nad končnim obsegom $\mathbb{Z}_p$. S končnimi obsegi se srečamo tudi pri kodah za popravljanje naključnih napak, ki se pojavijo pri prenosu ali hranjenju podatkov.

1.2 Kratka vsebina poglavij

V drugem poglavju bomo definirali končne obsege. Povedali bomo, v kakšnem smislu nam nerazcepni polinom generira končen obseg. Nato si bomo pogledali tri predstavitve končnega obsega: eksponentno, polinomsko in vektorsko, ki si jo bomo najbolj podrobno ogledali. Za izpeljavo zadnje predstavitve bomo vpeljali pojem baza za končne obsege. Definirali bomo operaciji seštevanja in množenja med elementi vektorske predstavitve obsega. Na koncu drugega poglavja si bomo ogledali še zgled vseh treh predstavitev na obsegu $\mathbb{F}_{2^3}$. V drugem razdelku tega poglavja bomo definirali polinomske, normalne in dualne baze. Podrobneje bomo obravnavali normalne baze. Tretje poglavje bo posvečeno prehodu med bazama z matriko. Predstavili bomo algoritem za določitev elementov matrike in napravili analizo algoritma. Ugotovili bomo, da je prehod z matriko časovne in prostorske zahtevnosti $\mathcal{O}(n^2)$. Motiv četrtega poglavja bo izboljšava časovne in prostorske zahtevnosti prehoda med bazama. V prvem razdelku bomo vpeljali osnovne operacije aritmetike končnega obsega, s katerimi bomo v

petem poglavju izpeljali nove algoritme. Nato bomo definirali problema uvoza in izvoza. V drugem razdelku četrtega poglavja pa bomo predstavili še tri leme, ki nam bodo posredovale osnovne ideje za izpeljavo izboljšanih algoritmov za prehod med bazama, ki bodo časovne in prostorske zahtevnosti $\mathcal{O}(n)$. Izboljšane algoritme bomo predstavili v petem poglavju. V šestem poglavju bo sledila še analiza novih algoritmov za prehod med bazama.

2 KONČNI OBSEGI

2.1 Predstavitev

Končni obsegi so obsegi s končnim številom elementov. Število elementov končnega obsega je potenca praštevila (*Birkhoff and Mac Lane 1996*). Končni obseg s q elementi bomo označili z $\mathbb{F}_q$, kjer je $q = p^n$ potenca praštevila.

V primeru, ko je $n = 1$, lahko končni obseg $\mathbb{F}_p$ predstavimo kot kolobar ostankov pri deljenju s p, torej $\mathbb{Z}_p$, z operacijama seštevanja in množenja po modulu p.

Za $n > 1$ pa je predstavitev malo bolj kompleksna. Najprej definirajmo pojem *nerazcepen polinom*.

Definicija. Polinom f nad obsegom $\mathbb{F}$ je *nerazcepen*, če ne obstajata taka nekonstantna polinoma g in h nad istim obsegom, da bi veljalo $f = gh$.

Trditev. Za vsak n obstaja nerazcepen polinom stopnje n s koeficienti iz $\mathbb{F}_p$ (glej [7], str. 83).

□

Spodnja tabela prikazuje vse nerazcepne polinome s koeficienti iz $\mathbb{F}_2$ stopenj od 1 do 5.

n	*nerazcepni polinomi*
1	$1 + x,\ x$
2	$1 + x + x^2$
3	$1 + x + x^3,\ 1 + x^2 + x^3$
4	$1 + x + x^4,\ 1 + x + x^2 + x^3 + x^4,\ 1 + x^3 + x^4$
5	$1 + x^2 + x^5,\ 1 + x + x^2 + x^3 + x^5,\ 1 + x^3 + x^5,\ 1 + x + x^3 + x^4 + x^5,$ $1 + x^2 + x^3 + x^4 + x^5,\ 1 + x + x^2 + x^4 + x^5$

Tabela 1: *Nerazcepni polinomi nad* $\mathbb{F}_2$ *do stopnje* 5.

Za implementacijo kriptografskih aplikacij so najbolj pomembni nerazcepni polinomi nad $\mathbb{F}_2$, ki imajo čim manjše število členov. Kandidat za nerazcepen polinom nad $\mathbb{F}_2$ mora imeti prosti člen, sicer bi lahko izpostavili faktor x. Poleg tega mora imeti še sodo potenc, sicer bi bila 1 ničla in bi lahko izpostavili faktor

$x+1$. Tako da so najpreprostejši kandidati trinomi oblike x^n+x^k+1. Če nerazcepni trinom ne obstaja, so naslednji kandidati pentanomi. Tabelo nerazcepnih trinomov nad $\mathbb{F}_2$ do stopnje 1478 lahko najdeš v [7], str. 158, 159.

V kakšnem smislu nerazcepen polinom stopnje n nad $\mathbb{F}_p$ generira obseg $\mathbb{F}_q$?

Denimo, da je α ničla nerazcepnega polinoma. Če računamo potence ničle, dobimo:

$$\begin{gathered}\alpha^0 = 1\\ \alpha^1\\ \vdots\\ \alpha^{p^n-2}\\ \alpha^{p^n-1} = 1.\end{gathered}$$

S potenciranjem ničle smo dobili $q-1$ različnih elementov. Torej je α primitiven element za multiplikativno grupo $\mathbb{F}_q$ brez ničle. Če dodamo še 0, dobimo q elementov. Dogovorimo se, da bomo ničlo označili z $\alpha^{-\infty}$. Dobili smo predstavitev $\mathbb{F}_q = \{\alpha^{-\infty}, \alpha^0, \alpha^1, \alpha^2, ..., \alpha^{q-2}\}$. Tej predstavitvi končnega obsega $\mathbb{F}_q$ pravimo *eksponentna predstavitev*. Množenje definiramo kot

$$\alpha^i \times \alpha^j = \alpha^{i+j \ (\bmod (q-1))}.$$

Za definicijo operacije seštevanja pa ta predstavitev ni praktična.

Zato si oglejmo raje drugo predstavitev $\mathbb{F}_q$ s polinomi stopnje manjše od n in koeficienti iz $\mathbb{F}_p$. Poljuben element obsega bo oblike $a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + ...+a_1x+a_0$. Definirajmo operaciji seštevanja in množenja med elementi obsega.

$$\sum_{i=0}^{n-1} a_i\ x^i + \sum_{i=0}^{n-1} b_i\ x^i = \sum_{i=0}^{n-1} (a_i + b_i)\ (\bmod\ p)\ x^i,$$

$$\sum_{i=0}^{n-1} a_i\ x^i \times \sum_{i=0}^{n-1} b_i\ x^i = \sum_{i=0}^{n-1} a_i\ x^i \sum_{i=0}^{n-1} b_i\ x^i\ (\bmod\ f(x)),$$

kjer je $f(x)$ nerazcepen polinom stopnje n nad $\mathbb{F}_p$. Ta predstavitev obsega $\mathbb{F}_q$ ustreza kvocientnemu prostoru

$$\mathbb{Z}_p[x]/f(x).$$

Definirali smo *polinomsko predstavitev* končnega obsega.

Oglejmo si še tretjo predstavitev. Na obseg $\mathbb{F}_q$ lahko gledamo tudi kot na vektorski prostor dimenzije n nad $\mathbb{F}_p$.

Definicija. Množica $\Omega = \{\omega_0, \omega_1, ..., \omega_{n-2}, \omega_{n-1}\}$ elementov iz $\mathbb{F}_q$ je *baza* za $\mathbb{F}_q$, če so elementi množice linearno neodvisni nad $\mathbb{F}_p$.

Tedaj lahko poljuben element obsega $\mathbb{F}_q$ enolično zapišemo kot linearno kombinacijo $\sum_{i=0}^{n-1} a_i\omega_i$, kjer so koeficienti a_i iz obsega $\mathbb{F}_p$. Identificirajmo $\sum_{i=0}^{n-1} a_i\omega_i$ z vektorjem $A = (a_0, a_1, ..., a_{n-1})$. Vektor A je *vektorska predstavitev* elementa glede na bazo Ω.

Definicija. Naj bo $\Omega = \{\omega_0, \omega_1, ..., \omega_{n-2}, \omega_{n-1}\}$ baza končnega obsega $\mathbb{F}_q$. *Multiplikacijska matrika* je matrika $T_k = (t_{ij}^{(k)})_{i,j=0}^{n-1}$ velikosti $n \times n$ nad $\mathbb{F}_p$, katere (i, j)-ti element je k-ti koeficient v razvoju produkta $\omega_i\omega_j$ v bazi Ω.

$$\omega_i\omega_j = \sum_{k=0}^{n-1} t_{ij}^{(k)}\ \omega_k$$

Množici $\{T_0, T_1, ..., T_{n-1}\}$ pravimo *multiplikacijska shema* za bazo Ω.

Naj bosta vektorja $A = (a_0, a_1, ..., a_{n-1})$ in $B = (b_0, b_1, ..., b_{n-1})$ vektorski predstavitvi elementov obsega $\mathbb{F}_q$. Izračunajmo njun produkt, ki ga označimo s $C = (c_0, c_1, ..., c_{n-1})$.

$$c_k = \left(\sum_{i=0}^{n-1} a_i\omega_i \times \sum_{j=0}^{n-1} b_j\omega_j\right)[k] = \sum_{i,j=0}^{n-1} a_i b_j(\omega_i\omega_j)\,[k] = \sum_{i,j=0}^{n-1} a_i t_{ij}^{(k)} b_j = AT_kB^T$$

Če poznamo multiplikacijsko shemo, lahko izračunamo vse komponente vektorja C. Vendar je za velike n ta shema nepraktična. Na srečo pa obstajajo baze za katere je multiplikacijska shema enostavnejša. To pomeni, da imajo matrike T_k manj neničelnih elementov ali pa ima shema druge regularnosti.

Sedaj si na enem primeru oglejmo še tri predstavitve obsega $\mathbb{F}_{2^3}$. Za nerazcepen polinom smo izbrali $f(x) = x^3 + x + 1$ iz *Tabele 1*, α pa je ničla tega polinoma.

eksponentna	*polinomska*	*vektorska v bazi* $\{\alpha^2, \alpha, 1\}$
$\alpha^{-\infty}$	0	$(0, 0, 0)$
α^0	1	$(0, 0, 1)$
α^1	α	$(0, 1, 0)$
α^2	α^2	$(1, 0, 0)$
α^3	$\alpha + 1$	$(0, 1, 1)$
α^4	$\alpha^2 + \alpha$	$(1, 1, 0)$
α^5	$\alpha^2 + \alpha + 1$	$(1, 1, 1)$
α^6	$\alpha^2 + 1$	$(1, 0, 1)$

Tabela 2: *Različne predstavitve obsega* $\mathbb{F}_{2^3}$.

2.2 Baze končnih obsegov

Če hočemo računati v končnih obsegih, potrebujemo aritmetiko končnega obsega. Najprej se moramo odločiti, kako bomo predstavili elemente obsega. To pomeni, da moramo izbrati bazo. Kot bomo v nadaljevanju videli, imamo na voljo več baz. Tu se takoj pojavi vprašanje, katero izbrati. Odločitev pa je odvisna od operacij, ki jih bomo izvedli. Kasneje bomo videli, da so lahko nekatere operacije v eni bazi veliko hitrejše kot v drugi.

Definicija. Naj bo α tak element iz $\mathbb{F}_q$, da je množica $\{\alpha^{n-1}, \alpha^{n-2}, ..., \alpha, 1\}$ baza za $\mathbb{F}_q$. Množici $\{\alpha^{n-1}, \alpha^{n-2}, ..., \alpha, 1\}$ pravimo *polinomska baza* za $\mathbb{F}_q$ nad $\mathbb{F}_p$.

Trditev. Za vsak končen obseg obstaja vsaj ena polinomska baza. Element α generira polinomsko bazo za $\mathbb{F}_q$ natanko takrat, ko je α ničla nerazcepnega polinoma stopnje n s koeficienti iz $\mathbb{F}_p$ (glej [8], str. 7).

□

Definicija. Naj bo β tak element $\mathbb{F}_q$, da je množica $\{\beta, \beta^p, \beta^{p^2}, ..., \beta^{p^{n-2}}, \beta^{p^{n-1}}\}$ baza za $\mathbb{F}_q$. Množici $\{\beta, \beta^p, \beta^{p^2}, ..., \beta^{p^{n-2}}, \beta^{p^{n-1}}\}$ pravimo *normalna baza* za $\mathbb{F}_q$ nad $\mathbb{F}_p$. Elementu β, ki to bazo generira, pa *normalni element*.

Definicija. Nerazcepnemu polinomu, ki ima za ničlo normalni element, pravimo *normalni polinom*.

Trditev. Za vsak končen obseg obstaja vsaj ena normalna baza. Elementi množice $\{\beta, \beta^p, \beta^{p^2}, ..., \beta^{p^{n-2}}, \beta^{p^{n-1}}\}$ so ničle normalnega nerazcepnega polinoma stopnje n s koeficienti iz $\mathbb{F}_p$ (glej [8], str. 8).

□

Ničla nerazcepnega polinoma stopnje n nad $\mathbb{F}_p$ nam vedno generira polinomsko bazo za $\mathbb{F}_q$. V posebnem primeru je lahko nerazcepni polinom tudi normalni polinom in je ničla normalni element, ki nam generira tudi normalno bazo. Tako da bom odslej v obeh primerih govoril o *polinomu obsega*.

Med vsemi bazami so najbolj pomembne tiste, pri katerih je množenje najbolj učinkovito. Za množenje v normalnih bazah ne potrebujemo celotne multiplikacijske sheme, temveč samo multiplikacijsko matriko T_0 (glej [9], str. 6 in [1], str. 107-108). Da bo množenje še hitrejše mora imeti multiplikacijska matrika čim manj neničelnih elementov. V ta namen bomo vpeljali pojem kompleksnost normalne baze. Manjša kompleksnost pomeni več ničel v multiplikacijski matriki in s tem hitrejše množenje v tej bazi.

Definicija. Označimo s C_N *kompleksnost normalne baze* N oziroma število neničelnih elementov v multiplikacijski matriki T_0 za normalno bazo N.

Trditev. Za vsako normalno bazo N velja, da je $C_N \geq 2n - 1$ (glej [9], str. 7).

□

Definicija. Normalna baza N je *optimalna*, če je $C_N = 2n - 1$. Optimalne normalne baze bom včasih označeval z ONB.

Trditev. Naj bo $n+1$ praštevilo in p primitiven v $\mathbb{Z}_{n+1}$. Potem so vsi $(n+1)$-vi koreni enote razen enote same linearno neodvisni in tvorijo ONB za $\mathbb{F}_{p^n}$ nad $\mathbb{F}_p$. Bazi, generirani na ta način, pravimo *ONB tipa 1.* Normalni polinom pa je

$$f(x) = \sum_{i=0}^{n-1} x^i$$

(glej [9], str. 65-67).

□

Trditev. Naj bo $2n + 1$ praštevilo in naj velja ena od točk:

(1) 2 je primitiven v $\mathbb{Z}_{2n+1}$ ali

(2) $2n + 1 \equiv 3 \pmod 4$ in 2 generira kvadratne ostanke v $\mathbb{Z}_{2n+1}$.

Potem element $\beta = \gamma + \gamma^{-1}$ generira ONB za $\mathbb{F}_{2^n}$ nad $\mathbb{F}_2$, kjer je γ primitiven $(2n + 1)$-vi koren enote. Bazi, generirani na ta način, pravimo *ONB tipa 2.* Normalni polinom je

$$f(x) = \prod_{j=1}^{n} (x - \gamma^j - \gamma^{-j})$$

(glej [9], str. 65-67).

□

Naj omenim še, da poznamo tudi družino *Gaussovih normalnih baz*, ki jih označimo z GNB. GNB tipa 1 je ravno ONB tipa 1, podobno je GNB tipa 2 ONB tipa 2. ONB so podmnožica GNB ([1], str. 92, 110).

Definicija. Naj bo množica $\{\omega_0, \omega_1, ..., \omega_{n-2}, \omega_{n-1}\}$ baza za $\mathbb{F}_q$ in $h : \mathbb{F}_q \to \mathbb{F}_p$ linearna neničelna preslikava. *Dualna baza* glede na h je $\{\psi_0, \psi_1, ..., \psi_{n-2}, \psi_{n-1}\}$, tako da velja

$$h(\omega_i \psi_j) = \begin{cases} 1 & ; \quad i = j \\ 0 & ; \quad i \neq j \end{cases}.$$

3 PREHOD MED BAZAMA Z MATRIKO

3.1 Prehodna matrika

Za kriptografske aplikacije, ki temeljijo na aritmetiki končnega obsega, je pomembna izbira aritmetike, saj hočemo doseči, da bo aplikacija delovala čim hitreje. To pa je odvisno od algoritmov, ki jih aplikacija uporablja in izbora baze. Nekateri algoritmi so namreč hitrejši, če imamo elemente obsega predstavljene v eni bazi, za druge algoritme pa je boljša druga baza. Tu se pojavi problem, kako element končnega obsega, ki je predstavljen glede na eno bazo, predstaviti v drugi bazi. Vektorju s komponentami glede na eno bazo je treba prirediti vektor s komponentami glede na drugo bazo. Iščemo izomorfizem med dvema končnima vektorskima prostoroma. Torej bomo prehod opisali z matriko.

Denimo, da imamo vektorsko predstvitev $B = (B_0, B_1, ..., B_{n-1})$ nekega elementa glede na bazo $\Omega = \{\omega_0, \omega_1, ..., \omega_{n-1}\}$. Radi pa bi imeli vektorsko predstavitev v bazi $\Psi = \{\psi_0, \psi_1, ..., \psi_{n-1}\}$ in sicer vektor $A = (A_0, A_1, ..., A_{n-1})$. Če A in B predstavljata isti element, mora veljati

$$\sum_{i=0}^{n-1} A_i \ \psi_i = \sum_{i=0}^{n-1} B_i \ \omega_i.$$

Poiskati pa je treba prehodno matriko $M : \Omega \to \Psi$, tako da bo veljalo

$$A = BM,$$

kjer vektorje pišemo kot vrstice. Potem je

$$A_i = \sum_{j=0}^{n-1} B_j \ M_{ji}.$$

Sledi

$$\sum_{i=0}^{n-1} A_i \ \psi_i = \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} B_j \ M_{ji} \ \psi_i = \sum_{j=0}^{n-1} B_j \ \sum_{i=0}^{n-1} M_{ji} \ \psi_i.$$

Od tod sledi, da je

$$\omega_j = \sum_{i=0}^{n-1} M_{ji} \ \psi_i.$$

Torej je j-ta vrstica matrike M razvoj elementa ω_j glede na bazo Ψ. V naslednjem razdelku bomo predstavili algoritem za določitev elementov matrike M pri prehodu med bazama za $\mathbb{F}_q$.

3.2 Algoritem za izračun elementov prehodne matrike

vhodni podatki:

(1) $p_0(u)$: polinom, ki generira bazo Ω za $\mathbb{F}_q$

(2) $p_1(t)$: polinom, ki generira bazo Ψ za $\mathbb{F}_q$

Opomba: V primeru polinomske baze je to nerazcepen polinom stopnje n nad $\mathbb{F}_p$, v primeru normalne baze pa normalni polinom stopnje n nad $\mathbb{F}_p$, katerega ničla je normalni element.

izhodni podatki:

prehodna matrika M med bazama Ω in Ψ

algoritem "Prehodna matrika"

1. Naj bo u ničla polinoma $p_0(u)$ glede na bazo Ψ, ki jo dobimo s probabilističnim algoritmom (glej [1], str. 103, 104).

2. Izračunaj elemente m_{ij} matrike M za $0 \leq i \leq n-1$ in $0 \leq j \leq n-1$:

2.1 Če sta Ω in Ψ polinomski bazi:

$$1 = \sum_{j=0}^{n-1} m_{n-1,j}\ t^{n-1-j}$$

$$u = \sum_{j=0}^{n-1} m_{n-2,j}\ t^{n-1-j}$$

$$\vdots$$

$$u^{n-2} = \sum_{j=0}^{n-1} m_{1,j}\ t^{n-1-j}$$

$$u^{n-1} = \sum_{j=0}^{n-1} m_{0,j}\ t^{n-1-j}.$$

2.2 *Če je Ω polinomska in Ψ normalna baza:*

$$1 = \sum_{j=0}^{n-1} m_{n-1,j}\ t^{p^j}$$

$$u = \sum_{j=0}^{n-1} m_{n-2,j}\ t^{p^j}$$

$$\vdots$$

$$u^{n-2} = \sum_{j=0}^{n-1} m_{1,j}\ t^{p^j}$$

$$u^{n-1} = \sum_{j=0}^{n-1} m_{0,j}\ t^{p^j}.$$

2.3 *Če sta* Ω *in* Ψ *normalni bazi:*

$$u = \sum_{j=0}^{n-1} m_{0,j}\ t^{p^j}.$$

Tako smo dobili 0-to vrstico, $(m_{0,0}, m_{0,1}, ..., m_{0,n-1})$, matrike M. Vsako naslednjo vrstico dobimo s cikličnim pomikom prejšnje v desno.

2.4 *Če je* Ω *normalna in* Ψ *polinomska baza:*

$$u = \sum_{j=0}^{n-1} m_{0,j}\ t^{n-1-j}$$

$$u^p = \sum_{j=0}^{n-1} m_{1,j}\ t^{n-1-j}$$

$$\vdots$$

$$u^{p^{n-2}} = \sum_{j=0}^{n-1} m_{n-2,j}\ t^{n-1-j}$$

$$u^{p^{n-1}} = \sum_{j=0}^{n-1} m_{n-1,j}\ t^{n-1-j}.$$

3. Izhod

$$M \leftarrow \begin{pmatrix} m_{0,0} & m_{0,1} & \dots & m_{0,n-1} \\ m_{1,0} & m_{1,1} & \dots & m_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ m_{n-1,0} & m_{n-1,1} & \dots & m_{n-1,n-1} \end{pmatrix}.$$

3.3 Zgledi prehodov med bazama

Zgled 2.1: polinomska baza → polinomska baza

Naj bo Ω polinomska baza s polinomom obsega $p_0(u) = u^5 + u^4 + u^2 + u + 1$ za $\mathbb{F}_{2^5}$. Za Ψ pa vzemimo polinomsko bazo s polinomom obsega $p_1(t) = t^5 + t^2 + 1$. Ničla polinoma p_0 glede na bazo Ψ pa je $u = t + 1$. Računamo:

$1 = 1$

$u = t + 1$

$u^2 = (t + 1)^2 = t^2 + 2t + 1 \equiv t^2 + 1$

$u^3 = (t^2 + 1)(t + 1) = t^3 + t^2 + t + 1$

$u^4 = (t^3 + t^2 + t + 1)(t + 1) = t^4 + 2t^3 + 2t^2 + 2t + 1 \equiv t^4 + 1.$

Na vsakem koraku je treba množiti z u in po potrebi rezultat oklestiti modulo polinom p_1. Iskana prehodna matrika je

$$M = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

Zgled 2.2: polinomska baza → normalna baza

Naj bo Ω polinomska baza s polinomom obsega $p_0(u) = u^5 + u^2 + 1$ in Ψ ONB tipa 2 za $\mathbb{F}_{2^5}$. Torej $p_1(t) = t^5 + t^4 + t^2 + t + 1$. Ničla polinoma p_0 je $u = t^2 + t^4 + t^8 + t^{16}$. Računamo:

$1 = t + t^2 + t^4 + t^8 + t^{16}$

$u = t^2 + t^4 + t^8 + t^{16}$

$u^2 = (t^2 + t^4 + t^8 + t^{16})^2 = t^4 + t^8 + t^{16} + t^{32} \equiv t + t^4 + t^8 + t^{16}$

$u^3 = (t + t^4 + t^8 + t^{16})(t^2 + t^4 + t^8 + t^{16}) \equiv t + t^4 + t^{16}$

$u^4 = (t + t^4 + t^{16})(t^2 + t^4 + t^8 + t^{16}) \equiv t + t^2 + t^8 + t^{16}.$

Na vsakem koraku smo množili z elementom u. Uporabili smo algoritem za

množenje v normalnih bazah, ki je opisan v [1], str. 107, 108.

$$M = \begin{pmatrix} 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

Zgled 2.3: normalna baza $\rightarrow$ normalna baza

Naj bo Ω GNB tipa 3 za $\mathbb{F}_{2^4}$. Torej $p_0(u) = u^4 + u^3 + 1$. Za Ψ pa vzemimo ONB tipa 1. Kar pomeni $p_1(t) = t^4 + t^3 + t^2 + t + 1$. Ničla polinoma p_0 je $u = t + t^4 + t^8$. Takoj lahko zapišemo prehodno matriko

$$M = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \end{pmatrix}.$$

Zgled 2.4: normalna baza $\rightarrow$ polinomska baza

Naj bo Ω ONB tipa 1 za $\mathbb{F}_{2^4}$. To pomeni, da smo za polinom obsega izbrali $p_0(u) = u^4 + u^3 + u^2 + u + 1$. Za Ψ pa vzemimo polinomsko bazo s polinomom obsega $p_1(t) = t^4 + t + 1$. Ničla polinoma p_0 glede na bazo Ψ je $u = t^3 + t^2$.

$u = t^3 + t^2$

$u^2 = (t^3 + t^2)^2 = t^6 + t^4 \equiv t^3 + t^2 + t + 1 \pmod{p_1}$

$u^4 = (t^3 + t^2 + t + 1)^2 = t^6 + t^4 + t^2 + 1 \equiv t^3 + t \pmod{p_1}$

$u^8 = (t^3 + t)^2 = t^6 + t^2 \equiv t^3 \pmod{p_1}$.

Na vsakem koraku smo kvadrirali in rezultat oklestili modulo polinom p_1. Tako smo dobili

$$M = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}.$$

Opomba. Prehodne matrike M v vseh zgornjih primerih sem dobil s funkcijo *transitonMatrix* in ustreznimi argumenti, ki sem jo sprogramiral v *Mathematici* in se nahaja v priloženi datoteki (glej [10]).

3.4 Analiza prehoda med bazama z matriko

Algoritem *Prehodna matrika* nam vrne prehodno matriko M velikosti $n \times n$, s katero znamo pretvarjati med bazama Ω in Ψ po formuli $A = BM$. Za prehod v drugi smeri pa rabimo inverzno matriko M^{-1} in računamo po formuli $B = AM^{-1}$. Torej, za prehod med bazama v obe smeri bi morali imeti poleg matrike M tudi njen inverz, kar bi nas stalo še enkrat več pomnilnika. Matriko M^{-1} bi lahko tudi izračunali iz matrike M, kar pa bi bila potrata časa. Prostorska zahtevnost prehoda je $\mathcal{O}(n^2)$ koeficientov iz $\mathbb{F}_p$. To lahko postane nezanemarljivo veliko že vprimeru, ko je $n = 160$ za $\mathbb{F}_{2^n}$. Potrebovali bi 3.2 kb. Druga slabost metode prehoda med bazama z matriko pa je, da ni nujno, da je mogoče implementirati produkt med matriko in vektorjem s koeficienti iz $\mathbb{F}_p$ z aritmetiko izbrane baze v poceni strojni opremi. Časovna zahtevnost prehoda med bazama z matriko je $\mathcal{O}(n^2)$ operacij v $\mathbb{F}_p$. Izračunati je potrebno n skalarnih produktov vektorjev dolžine n s komponentami iz $\mathbb{F}_p$.

4 IZBOLJŠAVA ČASOVNE IN PROSTORSKE ZAHTEVNOSTI PREHODA MED BAZAMA

Sedaj vemo, da lahko prehod med bazama izvršimo s pomočjo matrike. Eden izmed drugih pristopov reševanja problema prehoda med bazama je s pomočjo principa množenja z elementi iz dualne baze, vendar bi morali shraniti celotno dualno bazo, kar bi spet zahtevalo več pomnilnika. Ta rešitev je podrobno opisana v [4]. Naš cilj je izboljšati tako časovno kot prostorsko zahtevnost iz prehoda med bazama z matriko. V naslednjem razdelku bomo navedli orodja, ki jih imamo na voljo, ter definirali probleme, ki jih želimo rešiti.

4.1 Problem uvoza in izvoza

Želimo izvesti prehod med bazama, ne da bi pri tem porabili veliko pomnilnika ali potrebovali preveliko število operacij. Radi bi izkoristili učinkovitost operacij končnega obsega ene baze kot pa implementirali nove operacije, npr. množenje matrik. Pri prehodu z matriko smo namreč morali znati izračunati produkt BM, kjer je bil B vrstični vektor in M prehodna matrika. To sicer ni zapletena operacija, vendar ni nujno, da jo vsaka aritmetika končnega obsega podpira, saj ni osnovna operacija. Naj bo *notranja baza* tista baza, katere aritmetiko poznamo. Na naslednji strani so navedene operacije, ki jih aritmetika podpira.

Naj bosta A in $\widetilde{A}$ vektorski predstavitvi obsega $\mathbb{F}_q$. Element s naj bo skalar iz $\mathbb{F}_p$. Aritmetika notranje baze pozna naslednje operacije.

seštevanje oziroma odštevanje :	$A \pm \widetilde{A}$
množenje :	$A \times \widetilde{A}$
potenciranje :	A^i
množenje s skalarjem iz $\mathbb{F}_p$:	$s \times A$
izbira i-tega koeficienta :	$A\,[i]$

Drugo bazo pa bomo imenovali *zunanja baza*. Operacijo prehoda med zunanjo in notranjo bazo bomo imenovali *uvoz*. Operaciji v drugi smeri pa bomo rekli *izvoz*. Sledi podrobnejši opis dveh problemov.

- *problem uvoza*
Podani imamo notranjo in zunanjo bazo za $\mathbb{F}_q$ in vektor B, ki je vektorska predstavitev elementa obsega $\mathbb{F}_q$ glede na zunanjo bazo. Določiti pa je treba ustrezen vektor A, natančneje vektorsko predstavitev istega elementa v notranji bazi primarno z uporabo aritmetike notranje baze in čim manjšo prostorsko zahtevnostjo.

- *problem izvoza*
Podani imamo notranjo in zunanjo bazo za $\mathbb{F}_q$ in vektor A, ki je vektorska predstavitev elementa obsega $\mathbb{F}_q$ glede na notranjo bazo. Določiti pa je treba ustrezen vektor B, natančneje vektorsko predstavitev istega elementa v zunanji bazi primarno z uporabo aritmetike notranje baze in čim manjšo prostorsko zahtevnostjo.

Splošnejši problem prehoda med dvema zunanjima bazama Ω in $\Omega^{'}$ bomo rešili tako, da bomo najprej element iz baze Ω uvozili v notranjo bazo Ψ, nato pa izvozili v zunanjo bazo $\Omega^{'}$.

Kasneje bomo izpeljali štiri algoritme za prehod med bazama, po dva za problem uvoza in dva za problem izvoza. Privzeli bomo, da sta obe bazi definirani nad istim obsegom $\mathbb{F}_p$ ter da so koeficienti iz $\mathbb{F}_p$ predstavljeni na enak način v obeh bazah. Zahtevali bomo, da je zunanja baza polinomska ali pa normalna. To pomeni, da je element ϵ iz obsega $\mathbb{F}_q$ razvit po zunanji bazi oblike

$$\epsilon = \sum_{i=0}^{n-1} B\,[i]\,\gamma^i \tag{1}$$

ali pa

$$\epsilon = \sum_{i=0}^{n-1} B\left[i\right] \gamma^{p^i}, \tag{2}$$

kjer je γ generator zunanje baze in so $B\left[0\right], ..., B\left[n-1\right]$ iz $\mathbb{F}_p$ komponente vektorske predstavitve B v zunanji bazi. Privzeli bomo tudi, da je podana notranja predstavitev G elementa γ, sicer bi ga morali dodatno izračunati. Algoritmi za prehod bodo neodvisni od notranje baze. Lahko je polinomska, normalna ali pa celo kakšnega drugega tipa. Znali bomo pretvarjati med polinomsko in normalno bazo, med normalno in polinomsko, med polinomsko z enim generatorjem v polinomsko z drugim generatorjem ter med normalno z enim generatorjem v normalno z drugim generatorjem. Privzeli bomo, da imamo na voljo učinkovito aritmetiko notranje baze. Z drugimi besedami povedano, operacije, ki smo jih na prejšnji strani opisali, morajo biti kolikor je mogoče hitre.

4.2 Teoretično ozadje za izpeljavo novih algoritmov

Algoritme za uvoz iz zunanje baze lahko skonstruiramo na osnovi direktnega računanja enačb (1) in (2). Ker poznamo notranjo predstavitev G generatorja γ, lahko enostavno pretvorimo vsak element iz zunanje baze v predstavitev v notranji bazi z uporabo operacij v notranji bazi. Algoritmov za izvoz v zunanjo bazo pa ne moremo skonstruirati na enak način, saj ne vemo, kako pretvoriti vsak element v notranji bazi v predstavitev glede na zunanjo bazo z uporabo operacij v notranji bazi, pa četudi bi poznali zunanjo predstavitev generatorja. Zato bomo uporabili druge pristope za rešitev tega problema, ki jih bomo opisali v naslednjih treh lemah.

Lema 1. Denimo, da je zunanja baza polinomska z generatorjem γ. Naj bo B vektorska predstavitev elementa ϵ iz obsega $\mathbb{F}_{q}$ v zunanji bazi in $B^{'}$ vektorska predstavitev elementa $\epsilon\gamma^{-1}$ v zunanji bazi. Če je $B\left[0\right] = 0$, potem za $0 \leq i < n-1$ velja

$$B^{'}\left[i\right] = B\left[i+1\right].$$

Dokaz.

$$\epsilon = \sum_{i=0}^{n-1} B\left[i\right] \gamma^{i} = \sum_{i=1}^{n-1} B\left[i\right] \gamma^{i}$$

$$\epsilon\gamma^{-1} = \sum_{i=1}^{n-1} B\left[i\right] \gamma^{i-1} = \sum_{i=0}^{n-2} B\left[i+1\right] \gamma^{i}$$

Po drugi strani pa je

$$\epsilon\gamma^{-1} = \sum_{i=0}^{n-1} B^{'}\left[i\right] \gamma^{i}.$$

□

Če je zunanja baza polinomska, potem je množenje z γ^{-1} pomik koeficientov v levo, pri predpostavki, da je $B[0] = 0$. *Lemo 1* bomo uporabili pri algoritmu za izvoz v polinomsko bazo. Najprej bomo izračunali koeficient $B[0]$, nato odšteli $B[0]$, množili z G^{-1} in to proceduro ponavljali dokler ne bomo dobili vseh koeficientov $B[i]$.

Lema 2. Denimo, da je zunanja baza normalna. Naj bo B vektorska predstavitev elementa ϵ iz obsega $\mathbb{F}_q$ v zunanji bazi in $B^{'}$ vektorska predstavitev elementa ϵ^p v zunanji bazi. Potem velja za indekse $1 \leq i \leq n-1$ zveza

$$B^{'}[i] = B[i-1]$$

in $B^{'}[0] = B[n-1]$.

Dokaz.

Najprej se spomnimo, da v končnih obsegih karakteristike p velja enačba

$$(x+y)^p = x^p + y^p.$$

Z indukcijo se da hitro pokazati, da velja tudi enačba

$$\left(\sum_{i=0}^{n-1} x_i\right)^p = \sum_{i=0}^{n-1} x_i^p.$$

Računajmo

$$\epsilon^p = \left(\sum_{i=0}^{n-1} B[i]\,\gamma^{p^i}\right)^p = \sum_{i=0}^{n-1} B[i]^p\,\gamma^{p^{i+1}} = \sum_{i=0}^{n-1} B[i]\,\gamma^{p^{i+1}} =$$

$$B[0]\,\gamma^p + B[1]\,\gamma^{p^2} + ... + B[n-2]\,\gamma^{p^{n-1}} + B[n-1]\,\gamma^q =$$

$$B[0]\,\gamma^p + B[1]\,\gamma^{p^2} + ... + B[n-2]\,\gamma^{p^{n-1}} + B[n-1]\,\gamma.$$

Po drugi strani pa je

$$\epsilon^p = B^{'}[0]\,\gamma + B^{'}[1]\,\gamma^p + B^{'}[2]\,\gamma^{p^2} + ... + B^{'}[n-1]\,\gamma^{p^{n-1}}.$$

□

Drugače povedano, če je zunanja baza normalna, potem je p-ta potenca cikličen pomik koeficientov v desno. *Lemo 2* bomo uporabili pri algoritmu za izvoz v normalno bazo. Najprej bomo izračunali koeficient $B[n-1]$, nato izračunali p-to potenco in proceduro ponavljali, dokler ne dobimo vseh koeficientov.

Ostane nam samo še vprašanje, kako izračunati koeficienta $B[0]$ in $B[n-1]$. Spomnimo se prehoda med bazama z matriko M. Velja $B = AM^{-1}$. Koeficient $B[i]$ dobimo z naslednjo linearno kombinacijo

$$B[i] = \sum_{j=0}^{n-1} A[j]\;\; M^{-1}[j][i],$$

kjer so elementi $M^{-1}[j][i]$ iz obsega $\mathbb{F}_p$. Koeficiente $B[i]$ lahko izračunamo z operacijama '+' in '×' nad $\mathbb{F}_p$. V naslednji lemi bomo videli, da lahko koeficiente izračunamo samo z operacijami aritmetike notranje baze.

Lema 3. Naj bo množica $\{\omega_0, \omega_1, ..., \omega_{n-2}, \omega_{n-1}\}$ notranja baza, T_0 pa multiplikacijska matrika za notranjo bazo, ki smo jo definirali v prvem razdelku drugega poglavja. Naj bodo $s_0, s_1, ..., s_{n-1}$ elementi iz $\mathbb{F}_p$. Potem za katero koli predstavitev A v notranji bazi velja

$$\sum_{j=0}^{n-1} s_j A[j] = (A \times V)[0],$$

kjer je $V^T = (T_0)^{-1}(s_0, s_1, ..., s_{n-1})^T$.

Dokaz.

Ker je matrika T_0 obrnljiva, vektor V obstaja. Vemo tudi, da je $(A \times V)[0] = AT_0V^T$. Upoštevamo definicijo za V^T in lema sledi.

□

Lema 3 trdi, da lahko poljubno linearno kombinacijo komponent vektorske predstavitve v notranji bazi izračunamo z operacijami v notranji bazi. Označimo z V_i tisti vektor, za katerega velja

$$B[i] = (A \times V_i)[0],$$

to je takrat, ko so vrednosti $s_0, s_1, ..., s_{n-1}$ ravno vrednosti i-tega stolpca matrike M^{-1}.

5 IZBOLJŠANI ALGORITMI ZA PREHOD MED BAZAMA

V naslednjih algoritmih bomo z vektorjem $B = (B_0, B_1, ..., B_{n-1})$ označili vektorsko predstavitev elementa obsega $\mathbb{F}_q$ v zunanji bazi, ki je bodisi polinomska bodisi normalna. Vektor $A = (A_0, A_1, ..., A_{n-1})$ pa bo vektorska predstavitev elementa obsega $\mathbb{F}_q$ v notranji bazi. Oznaka G bo vektorska predstavitev generatorja zunanje baze v notranji bazi, oznaka G^{-1} pa vektorska predstavitev inverza generatorja zunanje baze glede na notranjo bazo. Vektorsko predstavitev enote za množenje v notranji bazi bomo označili z I.

5.1 Uvoz iz polinomske baze

Algoritem *UvoziPolinomsko* pretvori elemente obsega $\mathbb{F}_q$ iz vektorske predstavitve v polinomski bazi v vektorsko predstavitev v notranji bazi z uporabo

operacij v notranji bazi.

VHOD:	B, vektorska predstavitev elementa obsega $\mathbb{F}_q$ v polinomski bazi
IZHOD:	A, vektorska predstavitev istega elementa v notranji bazi
KONSTANTE:	G

proc $UvoziPolinomsko$
$A \leftarrow 0$
for $i \leftarrow\ n-1$ **down to** 0 **do**
    $A \leftarrow A \times G$
    $A \leftarrow A + B\,[i] \times I$
endfor

To je *Hornerjev* algoritem. V algoritmu *UvoziPolinomsko* imamo n množenj med elementi obsega ter n množenj s skalarjem. Potrebujemo pa tudi prostor za eno konstanto G. Algoritem nam vrne

$$A = \sum_{i=0}^{n-1} B\,[i]\,G^i.$$

To lahko hitro preverimo upoštevajoč dejstvo, da lahko A zapišemo na drugačen način kot

$$B\,[0] \times I + G \times (B\,[1] \times I + G \times (B\,[2] \times I + ... + G \times (B\,[n-1] \times I)...)).$$

Ali lahko število ponovitev zanke zmanjšamo? Odgovor je da, in sicer v primeru, ko je n popoln kvadrat, torej $n = k^2$ za neko naravno število k. Tedaj lahko v eni ponovitvi zanke sprocesiramo k koeficientov. Za motivacijo si poglejmo naslednje vrstice, kjer smo vse sumande zložili v k stolpcev. Vsak stolpec vsebuje k elementov.

$$\begin{array}{ccccccccc}
B\,[0]\,I & + & B\,[k]\,G^k & + & B\,[2k]\,G^{2k} & + & ... & + & B\,[(k-1)k]\,G^{(k-1)k} \\
B\,[1]\,G & + & B\,[k+1]\,G^{k+1} & + & B\,[2k+1]\,G^{2k+1} & + & ... & + & B\,[(k-1)k+1]\,G^{(k-1)k+1} \\
\vdots & & \vdots & & \vdots & & & & \vdots \\
B\,[k-1]\,G^{k-1} & + & B\,[2k-1]\,G^{2k-1} & + & B\,[3k-1]\,G^{3k-1} & + & ... & + & B\,[n-1]\,G^{n-1}
\end{array}$$

V drugi vrstici lahko izpostavimo G, v tretji G^2 in tako naprej do zadnje vrstice, kjer lahko izpostavimo G^{k-1}. Sedaj lahko zapišemo spremenjeno zanko.

for $i \leftarrow\ k-1$ **down to** 0 **do**
    $A \leftarrow A \times G$
    $A \leftarrow A + B\,[i + (k-1)k]\,G^{(k-1)k} + B\,[i + (k-2)k]\,G^{(k-2)k} + ... + B\,[i] \times I$
endfor

Število množenj s skalarjem ostane enako, medtem ko smo število množenj med elementi obsega zmanjšali kar za k-krat. Vendar pa potrebujemo prostor za dodatnih $k-1$ konstant, elementov množice $\{G^{(k-1)k}, G^{(k-2)k}, ..., G^k\}$. Teh k sumandov iz zanke bi lahko sprocesirali vzporedno, če bi imeli na voljo k procesorjev.

5.2 Uvoz iz normalne baze

Algoritem *UvoziNormalno* pretvori elemente obsega $\mathbb{F}_q$ iz vektorske predstavitve v normalni bazi v vektorsko predstavitev v notranji bazi z uporabo operacij v notranji bazi.

VHOD:	B, vektorska predstavitev elementa obsega $\mathbb{F}_q$ v normalni bazi
IZHOD:	A, vektorska predstavitev istega elementa v notranji bazi
KONSTANTE:	G

proc $UvoziNormalno$
$A \leftarrow 0$
for $i \leftarrow\ n-1$ **down to** 0 **do**
    $A \leftarrow A^p$
    $A \leftarrow A + B[i] \times G$
endfor

V algoritmu *UvoziNormalno* imamo n potenciranj na potenco p ter n množenj s skalarjem. Potrebujemo pa tudi prostor za eno konstanto G ter dodatno še za vmesni rezultat potenciranja. V algoritmu smo upoštevali dejstvo, da je $(A + B[i] \times G)^p = A^p + B[i] \times G^p$. Algoritem nam vrne

$$A = \sum_{i=0}^{n-1} B[i]\, G^{p^i}.$$

To se lahko hitro prepričamo, če upoštevamo dejstvo, da lahko A zapišemo na drugačen način kot

$$B[0] \times G + \big(B[1] \times G + ... + (B[n-2] \times G + (B[n-1] \times G)^p)^p ...\big)^p.$$

Algoritem *UvoziNormalno* lahko v primeru $n = k^2$ optimiziramo podobno kot prej s procesiranjem k koeficientov v eni ponovitvi zanke.

5.3 Izvoz v polinomsko bazo

Algoritem *IzvoziPolinomsko* pretvori elemente obsega $\mathbb{F}_q$ iz vektorske predstavitve v notranji bazi v vektorsko predstavitev v zunanji polinomski bazi, z uporabo operacij v notranji bazi.

VHOD:	A, vektorska predstavitev elementa obsega $\mathbb{F}_q$ v notranji bazi
IZHOD:	B, vektorska predstavitev istega elementa v polinomski bazi
KONSTANTE:	G^{-1}, V_0
	V_0 takšen, da velja $(A \times V_0)[0] = B[0]$ po *Lemi 3*

proc $IzvoziPolinomsko$
$A \leftarrow A \times V_0$
for $i \leftarrow 0$ **to** $n-1$ **do**
    $B[i] \leftarrow A[0]$
    $A \leftarrow A - B[i] \times V_0$
    $A \leftarrow A \times G^{-1}$
endfor

V algoritmu *IzvoziPolinomsko* imamo $n+1$ množenj med elementi obsega ter n množenj s skalarjem. Potrebujemo pa tudi prostor za dve konstanti. Algoritem *IzvoziPolinomsko* lahko v primeru $n = k^2$ podobno optimiziramo kot prej s procesiranjem k koeficientov v eni ponoviti zanke.

5.4 Izvoz v normalno bazo

Algoritem *IzvoziNormalno* pretvori elemente obsega $\mathbb{F}_q$ iz vektorske predstavitve v notranji bazi v vektorsko predstavitev v zunanji normalni bazi z uporabo operacij v notranji bazi.

VHOD:	A, vektorska predstavitev elementa obsega $\mathbb{F}_q$ v notranji bazi
IZHOD:	B, vektorska predstavitev istega elementa v normalni bazi
KONSTANTE:	V_{n-1}
	V_{n-1} takšen, da velja $(A \times V_{n-1})[0] = B[n-1]$ po *Lemi 3*

proc $IzvoziNormalno$
for $i \leftarrow n-1$ **down to** 0 **do**
    $T \leftarrow A \times V_{n-1}$
    $B[i] \leftarrow T[0]$
    $A \leftarrow A^p$
endfor

V algoritmu *IzvoziNormalno* imamo n potenciranj na potenco p ter n množenj med elementi obsega. Potrebujemo pa tudi prostor za konstanto ter vmesni rezultat potenciranja. Velja opomniti, da se vhodni podatek A kljub modifikacijam vrne na začetno vrednost. Algoritem *IzvoziNormalno* lahko v primeru $n = k^2$ podobno optimiziramo kot prej s procesiranjem k koeficientov v eni ponoviti zanke.

6 ZAKLJUČEK

Opisali smo štiri algoritme: uvoz polinomske baze, uvoz normalne baze, izvoz v polinomsko bazo in izvoz v normalno bazo. Z novimi algoritmi lahko izvedemo štiri vrste prehodov med bazama. Vsak prehod med bazama opravimo v dveh korakih: uvoz + izvoz.

vrsta prehoda	*opis prehoda*	*algoritma za prehod*
pp	polinomska baza → polinomska baza	*UvoziPolinomsko+IzvoziPolinomsko*
pn	polinomska baza → normalna baza	*UvoziPolinomsko+IzvoziNormalno*
nn	normalna baza → normalna baza	*UvoziNormalno+IzvoziNormalno*
np	normalna baza → polinomska baza	*UvoziNormalno+IzvoziPolinomsko*

Tabela 3: *Vrste prehodov med bazama in algoritmi.*

Naslednja tabela prikazuje analizo prehoda med bazama z novimi algoritmi. Analizirali smo število operacij ter prostorsko zahtevnost. Nismo upoštevali števila seštevanj med elementi obsega, ker je ta operacija zelo hitra. Prav tako smo zanemarili množenja s skalarjem iz $\mathbb{F}_p$. Navedli smo samo število množenj med elementi obsega $\mathbb{F}_p$ ter število potenciranj na potenco p. Število vhodnih konstant, ki so potrebne za prehod med bazama, pa določa prostorsko zahtevnost prehoda.

vrsta prehoda	*št. množenj*	*št. potenciranj na p*	*vhodne konstante*
pp	$2n+1$	-	$G,\ G^{-1},\ V_0$
pn	$2n$	n	$G,\ V_{n-1}$
nn	n	$2n$	$G,\ V_{n-1}$
np	$n+1$	n	$G,\ G^{-1},\ V_0$

Tabela 4: *Analiza zahtevnosti prehodov med bazama.*

Za prehod med bazama potrebujemo dodatno še dve ali tri konstante, kjer je vsaka od njih vektor dolžine n s komponentami iz $\mathbb{F}_p$. Torej je prostorska zahtevnost prehoda med bazama $\mathcal{O}(n)$ koeficientov iz $\mathbb{F}_p$, medtem ko smo pri prehodu z matriko potrebovali n^2 koeficientov iz $\mathbb{F}_p$ za prehodno matriko.

Potenciranje na potenco p lahko učinkovito izvršimo z algoritmom *Kvadriraj in zmnoži*, kar nas stane $\mathcal{O}(\log p)$ množenj med elementi obsega. Časovna zahtevnost prehoda med bazama je $\mathcal{O}(n)$ oziroma natančneje $\mathcal{O}(n \log p)$ operacij obsega $\mathbb{F}_q$. Naj še opozorim, da $\mathcal{O}(n^2)$ operacij nad $\mathbb{F}_p$ pri prehodu z matriko lahko vzame več časa kot $\mathcal{O}(n)$ operacij nad obsegom $\mathbb{F}_q$, če je aritmetika obsega $\mathbb{F}_q$ učinkovito implementirana. Tu mislim predvsem na izbiro ustrezne baze kot so naprimer ONB, ter učinkovita implementacija množenja in potenciranja med elementi obsega $\mathbb{F}_q$. Potenciranje na potenco p elementa obsega $\mathbb{F}_q$, kjer smo uporabili vektorsko predstavitev v normalni bazi, je kar ciklična rotacija vektorja v desno.

Literatura

[1] *Standard Specifications for Public Key Cryptography, Draft version 13, Annex A*. IEEE P1363, November 12, 1999. *http://grouper.ieee.org/groups/1363/P1363/draft.html*

[2] SAC '98: *Storage-Efficient Finite Fields Basis Conversion*. Burton S. Kaliski Jr. and Yiqun Lisa Yin, 1998.

[3] *"Methods and Apparatus for Efficient Finite Field Basis Conversion"*. Patent, U.S. Serial No. *5,854,759*. Burton S. Kaliski Jr. and Moses Liskov, November 1998.

[4] *"Efficient Finite Field Basis Conversion Involving A Dual Basis"*. Patent, U.S. Application Serial No. *09/195,346*. Burton S. Kaliski Jr. and Moses Liskov, November 1998.

[5] *Computation in Finite Fields*. John Kerl, Arizona State University and Avnet Design Services, April 22, 2004.

[6] *Finite Fields*. R. Lidl and H. Niederreiter, Cambridge University Press, 1996.

[7] *Handbook of Applied Cryptography*. Alfred J. Menezes, Paul C. van Oorschot and Scott A. Vanstone, CRC Press, October 1996.

[8] *Computational Complexity of Finite Field Multiplication*. Nils-Hassan Quttineh, August 2003. *http://www.ep.liu.se/exjobb/isy/2003/3402/exjobb.pdf*

[9] *Normal bases over Finite Fields*. Shuhong Gao. Waterloo, Ontario, Canada 1993. *http://www.math.clemson.edu/faculty/Gao/papers/thesis.pdf*

[10] *"Finite Fields Algorithms.nb"*. Patrik Mlekuž, January 2005.