INSTYTUT BADAŃ SYSTEMOWYCH
POLSKIEJ AKADEMII NAUK

ANALIZA SYSTEMOWA W FINANSACH I ZARZĄDZANIU

Wybrane problemy
Tom 11

Pod redakcją
Jerzego HOŁUBCA

Warszawa 2009

INSTYTUT BADAŃ SYSTEMOWYCH
POLSKIEJ AKADEMII NAUK

ANALIZA SYSTEMOWA W FINANSACH I ZARZĄDZANIU

Wybrane problemy
Tom 11

Pod redakcją
Jerzego HOŁUBCA

Warszawa 2009

Wykaz opiniodawców artykułów zamieszczonych w niniejszym tomie:

prof. dr hab. inż. Jerzy HOŁUBIEC
dr inż. Lech KRUŚ
doc. dr hab. inż. Wiesław KRAJEWSKI
doc. dr hab. Jacek MALINOWSKI
dr inż. Edward MICHALEWSKI
prof. dr Adam SKOREK
dr hab. Ryszard SMARZEWSKI
prof. dr hab. inż. Andrzej STRASZAK
dr Dominik ŚLĘZAK
prof. dr hab. inż. Stanisław WALUKIEWICZ
doc. dr hab. Sławomir ZADROŻNY

ISBN 9788389475220

Druk: Zakład Poligraficzny Jerzy Kosiński, Warszawa

SZYBKIE ALGORYTMY TRANSFORMACJI WIELOMIANOWYCH I ICH ZASTOSOWANIA KRYPTOGRAFICZNE

Joanna Kapusta
Studia Doktoranckie IBS PAN

The paper presents new algorithms for fast computing polynomial transformations and theirs applications in data encryption, secret sharing and e-voting. These algorithms use, in the essential way, algorithms for computing multidimensional convolutions and deconvolutions based on the FFT algorithm over finite fields.

***Key words**: multivariate polynomial interpolation and evaluation, encryption, secret sharing, e-voting, convolution, deconvolution, computational complexity, information security.*

Wstęp

Do ważniejszych zadań w kryptografii należy dzielenie sekretu oraz szyfrowanie danych. Określenia te pojawiają się przy rozpatrywaniu zagadnień związanych z bezpieczeństwem informacji. Podział sekretu polega na rozdzieleniu pewnej informacji, pomiędzy członków ustalonej grupy osób w taki sposób, aby tylko cała grupa, bądź uprawniona podgrupa mogła tę informację odzyskać. Natomiast szyfrowanie ma na celu zapewnienie poufności informacji przesyłanych niezabezpieczonym kanałem lub gromadzonych na różnych nośnikach danych. Wyróżnia się przy tym dwa systemy szyfrowania: symetryczne i asymetryczne [14], [15]. W kryptosystemach symetrycznych (np. DES, IDEA itp.) posługujemy się jednym kluczem prywatnym, który wykorzystujemy do szyfrowania i deszyfrowania informacji. W kryptosystemach asymetrycznych mamy dwa różne klucze: prywatny i publiczny. Szyfrowanie odbywa się przy pomocy klucza publicznego, natomiast deszyfrowanie przy pomocy klucza prywatnego.

W artykule proponuje się model szyfrowania symetrycznego w oparciu o szybkie algorytmy zmiany specjalnie wybranych wielowymiarowych baz wielomianowych, które uogólniają analogiczne jednowymiarowe wyniki omawiane w pracach [10] i [17]. W tym modelu zakłada się, że dana wiadomość cyfrowa

została podzielona na bloki $M=(a_\alpha)_{\alpha\in Q_n}$, gdzie $n=(n_1,n_2,..,n_d)$ i Q_n, jest zbiorem wielowskaźników $\alpha=(\alpha_1,\alpha_2,..,\alpha_d)$ takich, że $0\le\alpha_i<n_i$ dla każdego i. Następnie ustala się dwie bazy $\{L_\alpha(x)\}$ i $\{B_\alpha(x)\}$ w przestrzeni $P_n^d(K)$ wielomianów nad ciałem K, zmiennej $x=(x_1,x_2,...,x_d)\in K^d$. Jeżeli teraz dla danej wiadomości $M=(a_\alpha)_{\alpha\in Q_n}$ przyjmiemy

$$p(x)=\sum_{\alpha\in Q_n}a_\alpha L_\alpha(x) \tag{1}$$

to kryptogram $C=(c_\alpha)_{\alpha\in Q_n}$ tworzą współczynniki rozwinięcia tego wielomianu względem bazy $\{B_\alpha(x)\}$, tzn.

$$p(x)=\sum_{\alpha\in Q_n}c_\alpha B_\alpha(x). \tag{2}$$

Dla takiego modelu istotna jest znajomość dwóch szybkich algorytmów obliczania transformacji wielomianowych wyznaczających współczynniki rozwinięcia wielomianu $p(x)$ względem jednej bazy, gdy znane są współczynniki rozwinięcia tego wielomianu względem drugiej bazy. Należy zauważyć, że w literaturze jest dostępna szeroka gama baz wielomianowych: bazy złożone z wielomianów ortogonalnych względem dowolnie ustalonej funkcji wagowej, wielomiany fundamentalne Lagrange'a, wielomiany bazowe Newtona, Maclaurina, Bernsteina i inne.

Ważnym czynnikiem przy szyfrowaniu wydaje się być możliwość wyboru stosunkowo krótkiego klucza, gwarantującego jednak odpowiednio wysokie bezpieczeństwo. Z tego względu najbardziej obiecujące wydają się być algorytmy wykorzystujące bazy Lagrange'a, Newtona, Maclaurina, Bernsteina i bazy wielomianów ortogonalnych. W przypadku jednowymiarowym szybkie algorytmy transformacji pomiędzy bazami Lagrange'a i Newtona zostały zaprezentowane w pracy [18]. Wspomniana praca zasługuje na szczególną uwagę ze względu na specjalny dobór konfiguracji punktów, obejmujących - jako przypadki szczególne, wszystkie uprzednio znane konfiguracje. W przypadku d - wymiarowym punkty

$$x_\alpha=(x_{1,\alpha_1},x_{2,\alpha_2},...,x_{d,\alpha_d})\in K^d,\quad \alpha\in Q_n, \tag{3}$$

są generowane przez następujące wzory rekurencyjne

$$x_{i,j}=\lambda_i x_{i,j-1}+\delta_i,\quad i=1,2,..,d,\quad j=1,2,...,n_i-1, \tag{4}$$

gdzie λ_i, $x_{i,0} = \beta_i$, δ_i są dowolnie ustalone w K. W szczególnym przypadku możemy założyć, że te parametry nie zależą od i oraz przyjąć, że kluczem przy szyfrowaniu będzie $(d, n, \lambda, \beta, \delta)$.

Badania w zakresie szybkiego obliczania transformacji wielomianowych dla specjalnych konfiguracji punktów były prowadzone przez wiele lat. Szczególnym przypadkiem jest słynny algorytm FFT [8] obliczania dyskretnej transformacji Fouriera oraz algorytm transformacji do niej odwrotnej. Algorytmy te pozwalają na szybkie przechodzenie pomiędzy reprezentacją wielomianu $p(x)$ względem bazy Lagrange'a z węzłami interpolacji $1, \omega, \omega^2, ..., \omega^{n-1}$, gdzie ω jest pierwiastkiem pierwotnym z jedności stopnia $n = 2^k$, a współczynnikami rozwinięcia tego wielomianu względem bazy potęgowej. Ich złożoność obliczeniowa jest równa jedynie $O(n \log n)$, podczas gdy złożoność obliczeniowa innych, klasycznych algorytmów jest równa $O(n^2)$ lub w najlepszym przypadku $O(n \log^2 n)$ [1], [4]. Następnymi istotnymi wynikami były zaprezentowane w pracach [2] i [6] algorytmy rzędu $O(n \log n)$ dla ewaluacji wielomianów w n punktach tworzących ciągi geometryczne albo arytmetyczne. Dobre obszerne omówienie literatury związanej z tą tematyką można znaleźć w monografiach: A.V. Aho, J.E. Hopcroft i J.D. Ullman [1], D. Bini, V. Pan [3], D.E. Knuth [13] oraz artykule A.Bostan i E.Schost [6]. Po 2005 roku istotnym wynikiem w tym zakresie wydaje się być jednowymiarowe uogólnienie algorytmów rzędu $O(n \log n)$ obliczania transformacji wielomianowych na punkty generowane przez równanie rekurencyjne pierwszego rzędu, które zostało zaprezentowane w pracy [18]. Przypadek wielowymiarowy zostanie krótko opisany poniżej.

1. Szybkie algorytmy konwolucyjne i dekonwolucyjne

Podobną rolę jak algorytmy zmiany bazy mogą oczywiście pełnić w kryptografii znacznie prostsze, szybkie algorytmy obliczania zwiniętych i cyklicznych konwolucji (splotów) oraz dekonwolucji, których rząd jest tak jak poprzednio liniowo-logarytmiczny. Różnią się one jednak tym od szybkich algorytmów interpolacyjnych, że są lepsze od klasycznych algorytmów już dla stosunkowo małych wartości n. Przypomnijmy, że zwiniętym splotem wektorów $a = (a_i)_{i=0}^{n-1}$ i $b = (b_i)_{i=0}^{n-1}$ w K^n jest wektor $c = a \otimes_n b \in K^n$ o współrzędnych

$$c_i = \sum_{k=0}^{i} a_k b_{i-k}, \quad i = 0,1,...,n-1.$$

Jeśli w ciele K istnieje pierwiastek pierwotny ψ z jedności stopnia $2n = 2^{k+1}$, to splot może być obliczony następującym szybkim algorytmem:

$$a \otimes_n b = \left\{F_\omega^{-1}\left[F_\omega(a)\cdot F_\omega(b)\right] + F_\omega^{-1}\left[F_\omega(\psi\cdot a)\cdot F_\omega(\psi\cdot b)\right]/\Psi\right\}/2,$$

gdzie $\omega = \psi^2$, $\Psi = \left(1,\psi,...,\psi^{n-1}\right)$, $F_\omega : K^n \to K^n$ jest dyskretną transformacją Fouriera zaś mnożenie i dzielnic oznaczają mnożenie i dziclenie odpowiednich współrzędnych; zob. [1] i [18]. Ważnym zastosowaniem zwiniętych splotów jest algorytm Schönhage – Strassena binarnego mnożenia liczb całkowitych [16].

Wiadomo, że operacja dekonwolucji $a = c \otimes_n b^{-1}$ jest określona dla wektorów c, $b \in K^n$, o ile $b_0 \neq 0$. Niewiadomy wektor $d = b^{-1}$ zaleca się obliczać metodą Newtona [5], co wymaga $O(n \log n)$ operacji. Mianowicie, przyjmując

$$d = \left(\frac{1}{b_0}, -\frac{b_1}{b_0^2}, 0, 0, .., 0\right),$$

kolejne współrzędne wektora $d = b^{-1}$ są generowane tak, jak w Algorytmie 1.

Input: n, wektory $c = (c_0, c_1, \ldots, c_{n-1})$ i $b = (b_0, b_1, \ldots, b_{n-1})$, gdzie $b_0 \neq 0$.

Output: Wektor $a = c \otimes_n b^{-1} \in K^n$.

1. Oblicz $d = (1/b_0, -b_1/b_0^2, 0, 0, .., 0) \in K^n$.

2. Dla i od 2 do $\lceil \log_2 n \rceil$:

   2.1. Oblicz $d = 2 \cdot d - d \otimes_{2^i} d \otimes_{2^i} b$.

3. Oblicz $a = c \otimes_n d$.

Algorytm 1. Dekonwolucja metodą Newtona.

Zwróćmy uwagę, że każdy cykl pętli w Algorytmie 1 podwaja liczbę obliczonych współrzędnych wektora d.

Przykład 1. Załóżmy, że zostało wybrane ciało skończone Z_{239} reszt modulo 239. Jeżeli blok wiadomości cyfrowej ma następującą postać

$$a = [2 \quad 14 \quad 234 \quad 56 \quad 78 \quad 3 \quad 123 \quad 2],$$

zaś kluczem jest

$$k = [3 \quad 64 \quad 12 \quad 197 \quad 12 \quad 9 \quad 34 \quad 75],$$

to blokiem wiadomości zaszyfrowanej będzie

$$c = a \otimes_8 k = [6 \quad 170 \quad 188 \quad 171 \quad 87 \quad 94 \quad 235 \quad 155].$$

W praktycznych zastosowaniach, w przypadku długich wiadomości, kolejne bloki nie muszą być szyfrowane przy pomocy tego samego klucza k, ale np. przy pomocy poprzedzającego bloku kryptogramu c. Różne sposoby szyfrowania wiadomości o dowolnej długości określane są mianem trybów pracy algorytmów szyfrujących. Szczegółowe omówienie trybów pracy ze wskazaniem ich wad oraz zalet można znaleźć w monografii [14].

Deszyfrowanie wiadomości c polega na obliczeniu dekonwolucji $a = c \otimes_8 d$, gdzie $d = k^{-1}$. Zgodnie z Algorytmem 1 będzie ona najpierw wymagała obliczenia wektora d w następujący sposób

$$d = [80 \quad 46 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0],$$
$$d = [80 \quad 46 \quad 53 \quad 124 \quad 0 \quad 0 \quad 0 \quad 0],$$
$$d = [80 \quad 46 \quad 53 \quad 124 \quad 16 \quad 118 \quad 49 \quad 151],$$

a następnie obliczenia

$$a = c \otimes_8 d = [2 \quad 14 \quad 234 \quad 56 \quad 78 \quad 3 \quad 123 \quad 2].$$

Uogólnienie splotów jednowymiarowych do wielowymiarowych wymaga jedynie podania definicji i-tego cząstkowego splotu tensorowego

$$w = (w_\alpha)_{\alpha \in Q_n} = x \otimes_i y_i \in K^{n_1 \times n_2 \times \ldots \times n_d}, 1 \le i \le d,$$

hipermacierzy $x = (x_\alpha)_{\alpha \in Q_n}$ i wektora $y = (y_i)_{i=1}^d$, gdzie $y_i = (y_{i,0}, y_{i,1}, \ldots, y_{i,n_i-1})$ oraz $n = (n_1, n_2, \ldots, n_d)$. W tym celu przyjmuje się z definicji, że każda kolumna

$$w_{\beta_1,\ldots,\beta_{i-1},\bullet,\beta_{i+1},\ldots,\beta_d} = x_{\beta_1,\ldots,\beta_{i-1},\bullet,\beta_{i+1},\ldots,\beta_d} \otimes y_i,$$
$$0 \le \beta_j < n_j, \; j = 1,2,\ldots,i-1,i+1,\ldots,d,$$

hipermacierzy w jest równa jednowymiarowemu splotowi wektora y_i i kolumny

$$x_{\beta_1,\ldots,\beta_{i-1},\bullet,\beta_{i+1},\ldots,\beta_d} = \left(x_{\beta_1,\ldots,\beta_{i-1},j,\beta_{i+1},\ldots,\beta_d}\right)_{j=0}^{n_i-1}.$$

Input: Hipermacierz $x = (x_\alpha)_{\alpha \in Q_n}$ i tablica $y = (y_i)_{i=1}^{d}$,

gdzie $y_i = (y_{i,0}, y_{i,1}, \ldots, y_{i,n_i-1})$, $n = (n_1, n_2, \ldots, n_d)$, d.

Output: Wielowymiarowy splot tensorowy $z = (z_\alpha)_{\alpha \in Q_n}$.

1. Dla i od 1 do d:

    1.1. Oblicz $x = x \otimes_i y_i$.

2. Podstaw $z = x$.

Algorytm 2. Wielowymiarowy splot tensorowy $z = x \otimes y$.

Korzystając z tej definicji można obliczać wielowymiarowy tensorowy splot $z = x \otimes y$ Algorytmem 2, w którym hipermacierz wejściowa $x = (x_\alpha)_{\alpha \in Q_n}$ zmienia się d razy poprzez cząstkowe splatanie z wektorem y_i, dla $i = 1,2,\ldots,d$.

Algorytm dekonwolucji wielowymiarowej można skonstruować uogólniając idee podane w Algorytmie 1:

$$x = z \otimes_d y_d^{-1} \otimes_{d-1} y_{d-1}^{-1} \ldots \otimes_1 y_1^{-1}.$$

Ponieważ obliczenie i-tego splotu cząstkowego sprowadza się do obliczenia N/n_i ($N = n_1 n_2 \ldots n_d$) splotów jednowymiarowych, więc można stwierdzić że złożoność obliczeniowa Algorytmu 2 jest równa

$$\frac{N}{n_1} O(n_1 \log n_1) + \ldots + \frac{N}{n_d} O(n_d \log n_d) = O(N \log N).$$

Oczywiście rząd algorytmu obliczającego wielowymiarową dekonwolucję jest identyczny do powyższego.

Przykład 2. Załóżmy, że $n = (4,8)$ i $k = (k_1, k_2)$, gdzie

$k_1 = [2 \quad 12 \quad 7 \quad 15]$ i $k_2 = [11 \quad 67 \quad 4 \quad 12 \quad 152 \quad 7 \quad 10 \quad 23]$.

Następnie w zbiorze Z_{256} reszt modulo 256 ustalmy dwuwymiarową wiadomość (np. obraz) mającą postać cyfrową

$$x=\begin{bmatrix} 123 & 124 & 23 & 25 & 67 & 28 & 128 & 129 \\ 212 & 211 & 201 & 23 & 25 & 27 & 127 & 134 \\ 231 & 145 & 123 & 146 & 167 & 234 & 235 & 246 \\ 229 & 218 & 208 & 143 & 167 & 123 & 23 & 234 \end{bmatrix}.$$

Wtedy dwuwymiarowy, konwolucyjny Algorytm 2 generuje następujący kryptogram

$$c=x\otimes k=\begin{bmatrix} 146 & 10 & 186 & 152 & 64 & 100 & 182 & 116 \\ 164 & 86 & 180 & 56 & 128 & 120 & 210 & 208 \\ 41 & 159 & 75 & 162 & 230 & 142 & 57 & 60 \\ 213 & 128 & 235 & 230 & 152 & 216 & 72 & 120 \end{bmatrix}.$$

Po zastosowaniu algorytmu dekonwolucji

$$x=c\otimes_2 k_2^{-1}\otimes_1 k_1^{-1}$$

otrzymujemy wiadomość wyjściową x.

2. Efektywne algorytmy interpolacyjne i ewaluacyjne

W tym rozdziale zakładamy, że bazy $\{L_\alpha(x)\}$ i $\{B_\alpha(x)\}$ we wzorach (1) i (2) są odpowiednio interpolacyjnymi wielowymiarowymi wielomianami fundamentalnymi Lagrange'a i Newtona, tzn.

$$L_\alpha(x)=\prod_{i=1}^{d}\prod_{\substack{j=0\\ j\neq\alpha_i}}^{n_i-1}\frac{x_i-x_{i,j}}{x_{i,\alpha_i}-x_{i,j}} \text{ i } B_\alpha(x)=\prod_{i=1}^{d}\prod_{j=0}^{\alpha_i-1}(x_i-x_{i,j}),$$

gdzie węzły $(x_\alpha)_{\alpha\in Q_n}$ są takie jak w (3) i (4).

Okazuje się, że wielowymiarowa transformacja Lagrange'a-Newtona $T:(a_\alpha)_{\alpha\in Q_n}\to(c_\alpha)_{\alpha\in Q_n}$, opisująca przejście od rozwinięcia (1) względem bazy Lagrange'a $\{L_\alpha(x)\}$ do rozwinięcia (2) względem bazy Newtona $\{B_\alpha(x)\}$, i transformacja odwrotna T^{-1} mogą być obliczone przy pomocy szybkich Algorytmów 3 i 4 o złożoności obliczeniowej

$$O(N\log N),\ N=n_1n_2...n_d,$$

które uogólniają jednowymiarowe Algorytmy 1 i 2 z pracy [18]. Szczegóły związane z wyprowadzeniem tych algorytmów pomijamy w tej przeglądowej pracy. Zostaną one wkrótce opublikowane.

Input: $\lambda=(\lambda_1,\lambda_2,...,\lambda_d)$, $\delta=(\delta_1,\delta_2,...,\delta_d)$, $\beta=(\beta_1,\beta_2,...,\beta_d)$, $a=(a_\alpha)_{\alpha\in Q_n}$, d i $n=(n_1,n_2,...,n_d)$.

Output: $c=(c_\alpha)_{\alpha\in Q_n}$ - hipermacierz ilorazów różnicowych.

1. Oblicz q_α, r_α dla $\alpha\in Q_n$ ze wzorów

$$q_\alpha=\prod_{i=1}^{d}a_{\alpha_i}\left(\prod_{k=0}^{\alpha_i-1}\sum_{r=0}^{k}\lambda_i^r\right)^{-1},\quad r_\alpha=\prod_{i=1}^{d}\prod_{k=0}^{\alpha_i-1}\lambda_i^k(\beta_i(\lambda_i-1)+\delta_i)^j.$$

2. Dla $i=1,2,...,d$ oblicz $b_i=(b_{i,0},b_{i,1},...,b_{i,n_i-1})$, gdzie

$$b_{i,j}=(-1)^j\prod_{k=0}^{j-1}\lambda_i^k\Big/\prod_{k=0}^{j-1}\sum_{r=0}^{k}\lambda_i^r.$$

3. Oblicz splot wielowymiarowy $q=q\otimes b$.

4. Wykonaj dzielenie $c=q/r$ po współrzędnych.

Algorytm 3. Wielowymiarowa transformacja Lagrange'a-Newtona T.

Zwróćmy uwagę, że na sumaryczny koszt obliczenia wielowymiarowej transformacji Lagrange'a-Newtona T składają się:

— koszt obliczenia współrzędnych tablicy $b=(b_i)_{i=1}^{d}$ oraz elementów hipermacierzy $q=(q_\alpha)_{\alpha\in Q_n}$ i $r=(r_\alpha)_{\alpha\in Q_n}$ - $O(dN)$,

— koszt obliczenia splotu wielowymiarowego - $O(N\log N)$,

— koszt dzielenia odpowiednich współrzędnych hipermacierzy q i r - $O(dN)$

Zatem, przy założeniu, że $n_i\geq 2$ $(i=1,2,...,d)$, sumaryczny koszt obliczenia transformacji Lagrange'a-Newtona wynosi $O(N\log N)$, gdyż $\log N\geq d$. To samo jest prawdą dla złożoności obliczeniowej Algorytmu 4 obliczającego transformację odwrotną T^{-1}: $(c_\alpha)_{\alpha\in Q_n}\to(a_\alpha)_{\alpha\in Q_n}$.

Input: $\lambda=(\lambda_1,\lambda_2,...,\lambda_d)$, $\delta=(\delta_1,\delta_2,...,\delta_d)$, $\chi=(\chi_1,\chi_2,...,\chi_d)$, $c=(c_\alpha)_{\alpha\in Q_n}$, $n=(n_1,n_2,...,n_d)$.

Output: $a=(a_\alpha)_{\alpha\in Q_n}$ - wartości wielomianu w punktach x_α.

1. Oblicz g_α, z_α dla $\alpha\in Q_n$ ze wzorów

$$g_\alpha=\prod_{i=1}^{d} c_{\alpha_i}\prod_{k=0}^{\alpha_i-1}\lambda_i^k(\beta_i(\lambda_i-1)+\delta_i)^l,\quad z_\alpha=\prod_{i=1}^{d}\prod_{k=0}^{\alpha_i-1}\sum_{l=0}^{k}\lambda_i^l.$$

2. Dla $i=1,2,...,d$ oblicz $h_i=(h_{i,0},h_{i,1},...,h_{i,n_i-1})$, gdzie

$$h_{i,j}=\left(\prod_{k=0}^{\alpha_i-1}\sum_{l=0}^{k}\lambda_i^l\right)^{-1}.$$

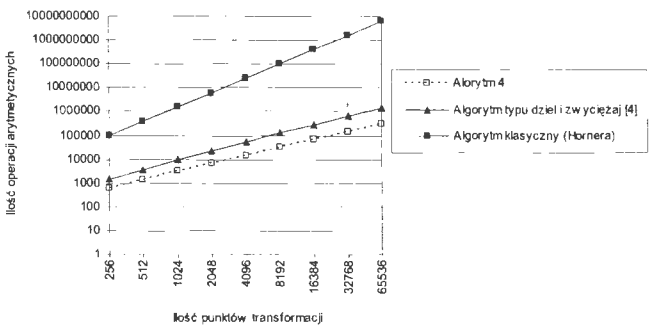
3. Oblicz splot wielowymiarowy $g=g\otimes h$.

4. Wykonaj mnożenie $a=g\cdot z$ po współrzędnych.

Algorytm 4. Wielowymiarowa transformacja Newtona-Lagrange'a T^{-1}.

Porównanie złożoności obliczeniowej przedstawionego algorytmu obliczania transformacji wielomianowej Newtona-Lagrange'a T^{-1} z klasycznymi, wspomnianymi w Rozdziale 1 algorytmami rzędu $O(N^2)$ i $O(N\log^2 N)$, podajemy na Rysunku 1 w przypadku jednowymiarowym. Ze względu na dużą różnicę pomiędzy liczbą operacji porównywanych algorytmów została zastosowana na wykresie skala logarytmiczna.

Podamy teraz przykład szyfrowania symetrycznego wykorzystującego Algorytmy 3 i 4. Dla prostoty rozważymy tym razem jedynie przypadek jednowymiarowy, co biorąc pod uwagę Przykłady 1 i 2, nie zmniejsza w sposób istotny ogólności rozważań.



Rys. 1. Porównanie ilości operacji arytmetycznych dla transformacji Newtona-Lagrange'a.

Przykład 3. Przypuśćmy, że $K = Z_{251265551}$ jest ciałem reszt modulo 251265551, zaś kluczem jest $(d, n, \lambda, \beta, \gamma) = (1{,}4096{,}3{,}1{,}3)$. Ponadto załóżmy, że szyfrowany tekst – fragment książki M. Twaina pt. „Adventures of Tom Sawyer", zawierający 12021 znaków, jest poprzedzony trzema znakami „XYZ". Te początkowe 3 znaki można wykorzystać do przesłania informacji, które nie muszą być zaszyfrowane. Początkowy, 180-cio znakowy fragment tego uzupełnionego tekstu ma postać:

XYZMonday morning found Tom Sawyer miserable. Monday morning always found him so - because it began another week's slow suffering in school. He generally began that day with wishing

Postać cyfrową uzupełnionego tekstu otrzymujemy zapisując go w postaci ciągu liczb 3-bajtowych. Zatem składa się on z 4008 liczb, które zapisujemy w tablicy $a = (a_i)_{i=0}^{4007}$. Początkowe 60 elementów tej tablicy, odpowiadających przytoczonemu tekstowi, wygląda następująco:

005921112 007237453 007954788 007302432 006909554 002123630 007696230
002122862 007171924 006378272 006650231 007151730 006648681 006447474
003040620 007294240 006382702 007151737 007238255 006778473 007102752
007954807 006692979 007239023 006824036 002125161 002125683 002108717
006514018 007566689 006889573 006430836 006383461 006365294 007630702
007497064 006649632 002583397 007544947 007827308 007697184 006645350
007235954 006889575 007544942 007301219 003042415 006637600 006645536
002108782 006386277 007957612 006644256 007233895 006845472 002126945
007954788 006911776 002123892 007563639

Stosując teraz szybki interpolacyjny Algorytm 3 dla $d=1$, $\lambda=3$, $\beta=1$, $\delta=3$ i $n=4096$ otrzymujemy tablicę $c=(c_i)_{i=0}^{4007}$ ilorazów różnicowych stanowiących kryptogram. Jego 60 pierwszych liczb ma postać:

005921112 215558521 111098319 249726932 143081377 131313827 199541408
166071050 143165337 157137598 114055741 174244462 095704099 025984029
164740457 022274275 117642376 078794317 162569913 193635551 081757934
208608640 060870243 135583787 073522998 004121278 041977994 149636178
043628983 219967444 043965721 185036348 247504193 014531128 109688713
249095312 202235910 162877733 211996576 136126691 068263952 107639739
018023332 009525745 055296678 016799116 091477580 133386479 039749225
045168252 120132734 114364426 185149176 029082409 234996561 092169528
097326209 145861281 157564277 120479061

Z kolei po zastosowaniu Algorytmu 4 do tablicy $c=(c_i)_{i=0}^{4007}$ z parametrami $d=1$, $\lambda=3$, $\beta=1$, $\delta=3$ ·i $n=4096$ otrzymujemy pierwotną wiadomość $a=(a_i)_{i=0}^{4007}$ w postaci cyfrowej, którą następnie zamieniamy na wiadomość alfanumeryczną.

Program do powyższego przykładu został zaimplementowany z użyciem biblioteki NTL (www.shoup.net/ntl).

3. Hierarchiczne dzielenie sekretu

Zadanie dzielenia sekretu można klasyfikować ze względu na wiele różnych kryteriów np. rodzaj uzyskiwanych udziałów lub wzajemne relacje uczestników podziału. Najbardziej znany jest schemat progowy (t z n), w którym każda grupa uczestników podziału sekretu o liczebności uczestników przekraczającej bądź równej wartości progowej t może odzyskać klucz [14], [15].

Interesujące hierarchiczne modele dzielenia sekretu uzyskujemy poprzez zastosowanie interpolacji Hermite'a [17] oraz Hermite'a-Birkoffa [19]. Poza szczególnymi przypadkami opisanymi w [11] i [12] nie są znane szybkie algorytmy zmiany bazy dla interpolacji Hermite'a. Z drugiej strony, klasyczne algorytmy rzędu $O(n^2)$ wydają się być wystarczające w tym przypadku, gdyż odpowiednia organizacja obliczeń [17] i niezbyt duże rozmiary rozważanych zadań redukują zyski z ewentualnego przyśpieszania algorytmów do minimum.

We wprowadzonym w pracy [17] hierarchicznym modelu podziału sekretu z losowo wybranym sekretem $s=a_0\in K$, dealer (zaufany podmiot dzielący se-

kret) wybiera losowo nickoniecznie parami różne argumenty $x_0, x_1, ..., x_{n-1} \in K \setminus \{0\}$ i współczynniki $a_1, a_2, .., a_{m-1} \in K$ wielomianu

$$w(x) = \sum_{k=0}^{m-1} a_k x^k, \ m \le n.$$

Następnie uogólnionym schematem Hornera oblicza wartości

$$y_i = \frac{w^{(k_i)}(x_i)}{k_i!}, \ i = 0,1,..,n-1,$$

gdzie

$$k_i = \max\{k : x_{i-k} = x_i\}$$

jest lewostronną krotnością węzła x_i. Przy tych oznaczeniach n uczestników podziału sekretu otrzymuje udziały zdefiniowane w postaci trójek

$$U_i = (k_i, x_i, y_i), \ i = 0,1,..,n-1,$$

przy tym wektory $k = (k_i)_{i=0}^{n-1}$ i $x = (x_i)_{i=0}^{n-1}$ mogą zostać upublicznione.

Do odtworzenia sekretu $s = a_0 = w(0)$, combiner (zaufany podmiot, ktory w imieniu grupy uczestników odzyskuje sekret) musi otrzymać r udziałów, powiedzmy

$$U_{i_j} = (k_{i_j}, x_{i_j}, y_{i_j}), j = 0,1,...,r-1,$$

gdzie $m \le r \le n$ oraz $0 \le i_0 < i_1 < ... < i_r < n$. Ponadto te udziały muszą spełniać warunki typowe dla interpolacji Hermite'a

$$w^{(k_i)}(x_i) = k_i! y_i, \quad x_i \in K, \ i = 0,1,..,r-1, \tag{5}$$

tzn. musi być $k_{i_0} = 0$ oraz

$$i_{j-1} = i_j - 1, i_{j-2} = i_j - 2,...,i_{j-k_{i_j}} = i_j - k_{i_j},$$

w przypadku, gdy $k_{i_j} > 0$. Wynika stąd, że proponowany podział sekretu jest hierarchiczny, gdyż posiadacz udziału $U_{i_j} = (k_{i_j}, x_{i_j}, y_{i_j})$, $j > 0$ nie może partycypować w odtworzeniu sekretu, o ile jego kolejni zwierzchnicy, posiadający udziały

$$U_{i_j-1}, U_{i_j-2}, \ldots, U_{i_j-k_j},$$

nie zrobią tego.

Bez zmniejszenia ogólności rozważań, będziemy dalej zakładać, że indeksy $i_0, i_1, \ldots, i_{r-1}$ są równe odpowiednio indeksom $0,1,\ldots,r-1$. Odtworzenie sekretu $s = w(0)$ wymaga wówczas obliczenia uogólnionych ilorazów różnicowych $(c_k)_{k=0}^{r-1}$, dla których zachodzi

$$w(x) = c_0 + c_1(x - x_0) + \ldots + c_{r-1}(x - x_0)(x - x_1)\ldots(x - x_{r-2}).$$

Przypomnijmy, że uogólnione ilorazy różnicowe

$$c_k = \langle y_0, y_1, \ldots, y_k \rangle, \quad k = 0,1,\ldots,r-1,$$

są zdefiniowane następującymi wzorami rekurencyjnymi

$$\langle y_i, \ldots, y_{i+k} \rangle = \begin{cases} y_{i+k-k_i}, & gdy \quad x_i = x_{i+k}, \\ \dfrac{\langle y_{i+1}, \ldots, y_{i+k} \rangle - \langle y_i, \ldots, y_{i+k-1} \rangle}{x_{i+k} - x_i}, & gdy \quad x_i \neq x_{i+k}, \end{cases}$$

w których $0 \leq i,\ i + k < r$ [9]. Oczywiście musi być

$$c_k = 0,\ k = m, m+1, \ldots, r-1.$$

Powyższe identyczności mogą być wykorzystywane do weryfikacji udziałów, czyli do sprawdzenia, czy wśród otrzymanych udziałów $U_0, U_1, \ldots, U_{r-1}$ nie ma udziałów niedopuszczalnych tzn. takich, w których trzecia współrzędna nie należy do zbioru wartości wielomianu $w(x)$ lub jego pochodnych.

Przykład 4. Załóżmy, że $K = Z_{37}$ jest ciałem reszt modulo 37 oraz, że dealer wybrał sekret $s = 27$, wielomian

$$w(x) = 2x^3 + 11x + 27,$$

węzły interpolacji

$$x_0 = x_1 = x_2 = 9,\ x_3 = x_4 = x_5 = 32$$

i liczbę $r = 5$, a następnie obliczył uogólnionym schematem Hornera składowe udziałów

$$y_0 = 30, y_1 = 16, y_2 = 17, y_3 = 18, y_4 = 13, y_5 = 7$$

i w końcu wręczył uczestnikom podziału sekretu udziały:

$$U_0 = (0,9,30), \quad U_1 = (1,9,16), \quad U_2 = (2,9,17),$$
$$U_3 = (0,32,18), \quad U_4 = (1,32,13), \quad U_5 = (2,32,7).$$

Zauważmy, że w tym przypadku struktura dostępu, czyli zbiór wszystkich podzbiorów zbioru udziałów pozwalających na odtworzenie sekretu bez sprawdzenia poprawności udziałów ma następującą postać

$$\{U_0, U_1, U_2, U_3\}, \{U_0, U_1, U_3, U_4\}, \{U_0, U_3, U_4, U_5\}$$

Combiner po otrzymaniu odpowiednich udziałów oblicza wektor uogólnionych ilorazów różnicowych, a następnie oblicza sekret będący wartością wielomianu w punkcie zero, $w(0) = 27$.

W przypadku, gdy combiner otrzyma $r = 5$ udziałów może weryfikować ich autentyczność. Istotnie, załóżmy, że combiner otrzymał udziały U_0, U_1, U_3, U_4, U_5, a następnie obliczył wektor uogólnionych ilorazów różnicowych $c = [30,16,26,2,0]$. Ponieważ ostatni iloraz różnicowy jest równy 0, więc wymienione udziały są autentyczne; gdyby ostatnia współrzędna była różna od zera to nie wszystkie udziały byłyby poprawne.

Rozważmy teraz zastosowanie wyżej opisanego modelu dzielenia sekretu przy przeprowadzaniu wyborów elektronicznych przez Internet. Załóżmy, że chcemy przeprowadzić głosowanie wśród dużej społeczności np. członków Amerykańskiego Towarzystwa Matematycznego, których liczba na całym świecie wynosi ok. 30 tyś. W tym celu spośród władz tej organizacji wybieramy komisję; dla uproszczenia przyjmijmy, że składa się ona z przewodniczącego i dwóch zastępców, bez których przeprowadzenie głosowania oraz podliczenie głosów nie jest możliwe. Przewodniczący otrzymuje udział postaci

$$U_0 = (0, x_0, y_0),$$

natomiast jego zastępcy udziały

$$U_1 = (1, x_1, y_1) \text{ i } U_2 = (2, x_2, y_2), \text{ gdzie } x_2 = x_1 = x_0.$$

Ponadto ustalamy administratora głosowania, który uzyskuje $c-3$, $c \le m$, udziałów

$$U_i = (k_i, x_i, y_i), \; i = 3,4,..,c-1,$$

takich, że $k_i = i$, $x_i = x_0$, co gwarantuje, że administrator nie może samodzielnie przeprowadzić podliczania głosów. Z kolei głosujący otrzymują udziały

$$U_i = (0, x_i, y_i),\ i = c, c+1, ..., n-1,$$

gdzie zakłada się, że $n - c < m$. Ostatnie założenie gwarantuje, że zebranie przez administratora wszystkich udziałów od głosujących nie umożliwia mu samodzielnego podliczenia głosów połączonego z ich weryfikacją.

Zauważmy, że w przypadku dużej liczby głosujących (rzędu kilku tysięcy) warto dobrać węzły interpolacji tak, aby spełniały równanie rekurencyjne pierwszego rzędu i wykorzystać szybki Algorytm 4 do obliczenia udziałów poszczególnych głosujących.

Rozpoczęcie podliczania głosów w tym modelu jest możliwe wówczas, gdy oprócz przewodniczącego, zastępców i administratora, co najmniej $r - c$ głosujących prześle swoje udziały wraz z dokonanym wyborem. Proces podliczania głosów inicjalizuje przewodniczący wraz z zastępcami składając swoje udziały administratorowi, który tworzy wektor zmodyfikowanych ilorazów różnicowych

$$b_k = \langle y_k, y_{k+1}, ..., y_{c-1} \rangle = y_{c-k-1},\ k = 0,1,...,c-1.$$

Administrator używa tego wektora wielokrotnie, stosując Algorytm 5 w celu weryfikowania autentyczności napływających głosów.

Input: id - indeks, y_{id}, $x = (x_0, x_1, ..., x_{id-1})$, $k = (k_0, k_1, ..., k_{id-1})$, $b = (b_0, b_1, ..., b_{id-1})$.

Output: $b = (b_0, b_1, ..., b_{id})$ - wektor ilorazów różnicowych.

1. Podstaw $b_{id} = y_{id-k_i}$.
2. Dla $i = id - 1$ do $id - k_{id}$ podstaw $b_i = y_{2id-k_{id}-i}$.
3. Dla $i = id - k_{id} - 1$ do 0 oblicz $b_i = (b_{i+1} - b_i)/(x_{id} - x_i)$.

Algorytm 5. Obliczania ilorazów różnicowych.

Przykład 5. Przypuśćmy, że podmiot przygotowujący wybory wybrał ciało $K = Z_{62533}$, a następnie ustalił parametry $n = 60057$, $m = c = 30057$, $r = 50057$ i wielomian

$$w(x) = 4123 + 2343(x-156) + 87(x-156)^2 + 35786(x-156)^{30056}$$

oraz punkty

$$x_0 = x_1 = ... = x_{30056} = 156,$$
$$x_i = 123x_{i-1} + 3456,1\ \ (i = 3057,...,60056;\ x_{3056} = 41).$$

Następnie wręczył udziały: przewodniczącemu $U_0 = (0,156,4123)$, pierwszemu zastępcy $U_1 = (1,156,2343)$, drugiemu zastępcy $U_2 = (2,156,87)$, administratorowi $U_{30056} = (30056,156,35786)$ wraz z informacją, że wszystkie pozostałe wartości pochodnych rzędu niższego od 30056 są równe zero tzn. $U_i = (i,156,0)$ dla $i = 3,4,...,30055$.

Ostatnią czynnością podmiotu przygotowującego wybory jest zastosowanie Algorytmu 4 do obliczenia 30000 udziałów

$$U_i = (0, x_i, w(x_i)), \ (i = 30057,30058,..,60056),$$

które otrzymują głosujący. Poniżej prezentujemy kilka przykładowych udziałów:

$$U_{30057} = (0,8499,6201), \ U_{30058} = (0,48305,6201),$$

...

$$U_{60055} = (0,44038,47086), U_{60056} = (0,42292,58254).$$

Administrator, po otrzymaniu udziałów U_0, U_1 i U_2 przewodniczącego i zastępców oraz wykorzystując swoje udziały, inicjalizuje proces głosowania budując wektor

$$b = [35786,0,0,...,0,0,87,2343,4123].$$

Po zakończeniu głosowania administrator zlicza liczbę oddanych głosów ważnych i podaje wyniki głosowania. W tym celu dla każdego udziału wykonuje Algorytm 5. Oczywiście, głosy ważne to te, dla których Algorytm 5 wygeneruje $b_0 = 0$.

W celu zapewnienia anonimowości głosujących możemy na etapie głosowania ustalić tzw. zaufanego mieszacza tak, jak w protokole wyborczym zaproponowanym przez D. Chauma [7]. Zwróćmy uwagę, że odnotowywanie przez administratora wartości x_i, dla których już zostały oddane głosy zapewnia niepowtarzalność oddawanych głosów tzn. żaden głosujący nie może głosować więcej niż jeden raz. Ponadto opublikowanie, po przeprowadzeniu wyborów, listy tych wartości wraz ze związanymi z nimi głosami umożliwia sprawdzenie przez głosujących czy ich głosy zostały poprawnie policzone. Podkreślmy również, że nieuczciwy głosujący (nieposiadający uprawnień do głosowania) nie ma wpływu na wynik wyborów - może przesłać niepoprawny głos, ale zostanie to wykryte w fazie zliczania głosów.

Literatura

[1]. A. V. Aho, J. E. Hopcroft, J. D. Ullman (2003): *Projektowanie i analiza algorytmów.* Helion, Gliwice.

[2]. A. V. Aho, K. Steiglitz, J. D. Ullman (1975): Evaluating polynomials at fixed sets of points. *SIAM J. Comput.* 4, 533 – 539.

[3]. D. Bini, V. Pan (1994): *Polynomial and Matrix Computation 1.* Boston.

[4]. A. Borodin, I. Munro (1971): Evaluating polynomials at many points. *Inform. Process. Lett.* 1 (2), 66 – 68.

[5]. J. M. Borwein, P. B. Borwein, Pi and the AGM (1987): *A study in analytic number theory and computational complexity.* Canadian Mathematical Society Series of Monographs and Advanced Texts, John Wiley and Sons, New York.

[6]. A. Bostan, E. Schost (2005): Polynomial evaluation and interpolation on special sets of points. *J. Complexity* 21, 420 – 446.

[7]. D. Chaum (1981): Untraceable electronic mail, return addresses and digital pseudonyms. *Communications of the ACM* 24, 84-88.

[8]. J. M. Coley, W. J. Tukey (1965): An algorithm for the machine calculation of complex Fourier series. *Match. Comp.* 19, 297 – 301.

[9]. D. Kincaid, W. Cheney (2006): Analiza numeryczna. WNT, Warszawa.

[10]. J. Kapusta, R. Smarzewski (2006): Fast interpolating algorithms in cryptography. *Annales UMCS Informatica AI*,5, 37 – 45.

[11]. J. Kapusta, R. Smarzewski (2006): Fast Hermite interpolating polynomial algorithm for a special configuration of knots. *Academic Research*, 19, no. 3, 67 – 74.

[12]. J. Kapusta, R. Smarzewski (2007): Fast algorithms for polynomial evaluation and differentiation at special knots. *Annales UMCS Informatica AI*, 6, 65 – 102.

[13]. D. E. Knuth (2002): *Sztuka Programowania. Algorytmy seminumeryczne.* WNT, Warszawa.

[14]. A. J. Menezes, P. C. van Oorschot, S. A. Vanstone (2005): *Kryptografia stosowana.* WNT, Warszawa.

[15]. J. Pieprzyk, T. Hardjono, J. Seberry (2003): *Teoria bezpieczeństwa systemów komputerowych.* Helion, Gliwice.

[16]. A. Schönhage, V. Strassen (1971): Schnelle multiplikation großer zahlen. *Computing,* 7, 281 – 292.

[17]. R. Smarzewski, J. Kapusta (2005): Algorithms for multi-secret hierarchical sharing schemes of Shamir type. *Annales UMCS Informatica,* AI 3, 65 – 91.

[18]. R. Smarzewski, J. Kapusta (2007): Fast Lagrange-Newton transformations. *J. Complexity*, 23, 336 – 345.

[19]. T. Tassa (2007): Hierarchical Threshold Secret Sharing. *J. Cryptology*, 20, 237 – 264.

ISBN 9788389475220