ŚREDNIA ZŁOŻONOŚĆ OBLICZENIOWA PROBABILISTYCZNEGO ALGORYTMU WYSZUKIWANIA PIERWIASTKÓW PIERWOTNYCH MODULO N

Tomasz Adamski

Instytut Systemów Elektronicznych;
Wydział Elektroniki i Technik Informacyjnych Politechniki Warszawskiej,
T.Adamski@ise.pw.edu.pl

Streszczenie. W pracy oszacowano średnią złożoność obliczeniową probabilistycznego algorytmu wyszukiwania pierwiastków pierwotnych modulo n. Uzyskany wynik może być w naturalny sposób uogólniony na przypadek algorytmu wyszukiwania generatorów dowolnej skończonej grupy cyklicznej jeśli znamy rozkład na czynniki pierwsze rzędu tej grupy.

Słowa kluczowe: algorytmy kryptograficzne, algorytmy probabilistyczne, średnia złożoność obliczeniowa, pierwiastki pierwotne modulo n generatory skończonych grup cyklicznych.

1. Wprowadzenie

Pierwiastek pierwotny z liczby n (lub modulo n) dla $n \in N, n \geq 2$ to inaczej generator grupy multiplikatywnej Z_n^*. Zatem algorytm wyznaczania pierwiastków pierwotnych z liczby n to inaczej algorytm wyszukiwania generatorów grupy multiplikatywnej Z_n^*. Ponieważ grupa multiplikatywna Z_n^* nie dla każdego $n \in N, n \geq 2$ jest cykliczna, pierwiastek pierwotny z liczby n istnieje lub nie istnieje. Sprawę istnienia generatora grupy Z_n^*, czyli sprawę istnienia pierwiastków pierwotnych z n wyjaśnia następujące znane twierdzenie.

Twierdzenie 1. *Grupa multiplikatywna Z_n^* dla $n \in N, n \geq 2$ jest grupą cykliczną wtedy i tylko wtedy, gdy $n = 2, 4, p^k, 2p^k$, gdzie p jest nieparzystą liczbą pierwszą oraz $k \in N$.*

Dowód. por. np. W. Narkiewicz [4] □

Przykład. Liczba 3 jest generatorem grupy multiplikatywnej Z_7^*. Łatwo to sprawdzić podnosząc 3 do kolejnych potęg 2,3,...,6 modulo 7 tzn. obliczając: $3^1 \pmod 7 = 3$, $3^2 \pmod 7 = 2$, $3^3 \pmod 7 = 6$, $3^4 \pmod 7 = 4$, $3^5 \pmod 7 = 5$, $3^6 \pmod 7 = 1$.

Znalezienie generatora grupy multiplikatywnej Z_n^* (gdzie n jest liczbą taką jak w twierdzeniu 1) jest szczególnie ważne dla algorytmów kryptograficznych, w których często wykorzystujemy generatory grupy Z_n^* jako dane wejściowe. Do takich algorytmów należą np. algorytm szyfrowania ElGamala, algorytmy podpisów cyfrowych ElGamala i Nyberga-Ruppela i bardzo popularny algorytm uzgadniania kluczy Diffiego-Hellmanna.

W dalszym ciągu dla uproszczenia zapisu przyjmujemy, że $n = p$, gdzie p jest liczbą pierwszą. Jeśli n jest dowolną liczbą naturalną dla, której istnieje pierwiastek pierwotny modulo n to w opisie algorytmu zamiast rozkładu na czynniki pierwsze liczby $p - 1$ używamy rozkładu na czynniki pierwsze liczby $\varphi(n)$, gdzie φ jest funkcją Eulera.

2. Algorytm wyszukiwania generatorów grupy multiplikatywnej

Nie są znane efektywne algorytmy obliczania generatorów grupy multiplikatywnej Z_p^* jeśli dane wejściowe ograniczają się tylko do wartości liczby pierwszej p. Jeśli natomiast znamy rozkład liczby $p-1$ na czynniki pierwsze tzn. wiemy, że

$$p - 1 = q_1^{k_1} q_2^{k_2} \dots q_r^{k_r}$$

gdzie $q_1, q_2, \dots, q_r$ są parami różnymi liczbami pierwszymi a $k_1, k_2, \dots, k_r$ liczbami naturalnymi, to istnieje prosty algorytm probabilistyczny opisany w języku publikacyjnym na rys.1 obliczający generator grupy Z_p^*. Ten algorytm będziemy analizować w dalszym ciągu. Algorytm ten ma jednak pewną wadę, wymaga bowiem znajomości rozkładu liczby $p-1$ na czynniki pierwsze. Z kolei uzyskanie rozkładu liczby całkowitej na czynniki pierwsze jest ogólnie rzecz biorąc problemem obliczeniowo trudnym. Konieczność znajomości rozkładu liczby $p - 1$ na czynniki pierwsze ogranicza użyteczność omawianego algorytmu ale jeśli konkretna wartość liczby pierwszej p nie jest specjalnie istotna to można postąpić tak:

1. Wygenerować parami różne liczby pierwsze $q_1, q_2, \dots, q_r$ przyjąć wykładniki $k_1, k_2, \dots, k_r \in N$ i obliczyć wartość $n = q_1^{k_1} \cdot q_2^{k_2} \cdot \dots \cdot q_r^{k_r}$

2. Przetestować pierwszość liczby $n + 1$ np. algorytmem Millera-Rabina. Jeśli liczba $n + 1$ jest pierwsza to można przyjąć $p = n + 1$ i znamy jednocześnie rozkład liczby $p - 1$ na czynniki pierwsze. Możemy więc zastosować omawiany algorytm.

3. Jeśli liczba $n+1$ nie jest pierwsza to szukamy nowego n jak w p. 1.

Algorytm: Probabilistyczny algorytm wyszukiwania generatorów grupy multiplikatywnej Z_p^*

Dane wejściowe: 1. liczba pierwsza p
2. rozkład na czynniki pierwsze liczby $p-1$ tzn. takie parami różne liczby pierwsze $q_1, q_2, \ldots, q_r$ i liczby naturalne $k_1, k_2, \ldots, k_r \in N$, że $p-1 = q_1^{k_1} q_2^{k_2} \ldots q_r^{k_r}$

Dane wyjściowe: $g \in Z_p^*$ generator grupy multiplikatywnej Z_p^*

$g := 1$;
for $i := 1$ **to** r **do begin**

repeat

wybierz losowo z rozkładem równomiernym liczbę $a \in Z_p^*$;
szukamy elementu rzędu $q_i^{k_i}$
$b := a^{\frac{p-1}{q_i}}$;

until $(b \neq 1)$;

$c := a^{\frac{p-1}{q_i^{k_i}}}$; obliczamy element rzędu $q_i^{k_i}$

$g := (g * c) (\mathrm{mod}\, p)$; obliczamy iloczyn elementów rzędu $q_1^{k_1}, q_2^{k_2}, \ldots, q_r^{k_r}$

end;
write("generator =", g);

Rysunek 1. Probabilistyczny algorytm obliczania generatorów grupy multiplikatywnej Z_p^*

Podany sposób doboru liczby pierwszej p do znanego rozkładu na czynniki pierwsze liczby $p-1$ uzupełniony naszym algorytmem okazuje się w praktyce skuteczny i prowadzi do znalezienia generatora „jakiejś tam" grupy multiplikatywnej Z_p^*. Jednak w wielu praktycznych zastosowaniach o to nam właśnie chodzi.

Zauważmy, że naturalny algorytm obliczania generatora Z_p^*, polegający na losowym wyborze elementu $a \in Z_p^*$ i sprawdzeniu czy kolejne potęgi $a^1, a^2, \ldots, a^{p-1}$ wyczerpują zbiór elementów grupy Z_p^*, ma wykładniczą złożoność obliczeniową. Nie jest to więc algorytm, który nadaje się dla dużych liczb pierwszych.

3. Uzasadnienie poprawności algorytmu

Kluczowe dla uzasadnienia poprawności algorytmu są następujące twierdzenia z teorii grup.

Twierdzenie 2. *Grupa multiplikatywna ciała skończonego $GF(p^k)$ jest grupą cykliczną rzędu $p^k - 1$.*

Dowód. por. np. N. Koblitz [2], C. Bagiński [3]. □

Twierdzenie 3. *Każda podgrupa grupy cyklicznej jest cykliczna*

Dowód. por. np. N. Koblitz [2], C. Bagiński [3]. □

Twierdzenie 4. *Jeśli liczba naturalna d jest dzielnikiem rzędu skończonej grupy cyklicznej G to istnieje dokładnie jedna podgrupa rzędu d grupy G.*

Dowód. por. np. N. Koblitz [2], C. Bagiński [3]. □

Prostym wnioskiem z twierdzeń 3 i 4 jest następujące twierdzenie 5

Twierdzenie 5. *Jeśli liczba naturalna d jest dzielnikiem rzędu skończonej grupy cyklicznej G to istnieje w grupie G element rzędu d.*

Zasadnicze dla poprawności algorytmu są następujące dwa twierdzenia: 6 i 7 o rzędzie elementu grupy G.

Twierdzenie 6. *Niech G będzie grupą i niech a będzie elementem grupy G. Niech ponadto dla pewnej liczby pierwszej p i pewnej liczby $e \in N$ mamy*

$$a^{p^e} = 1 \tag{1}$$

$$a^{p^{(e-1)}} \neq 1 \tag{2}$$

wówczas element a ma rząd p^e.

Dowód. Jeśli m jest rzędem elementu $a \in G$ to ponieważ, $a^{p^e} = 1$, zatem $m|p^e$. Zatem $m = p^f$ dla pewnego $f \in < 0, e >$. Jeśli $f < e$ to $a^{p^f} = 1$ i również $a^{p^{e-1}} = 1$ co jest sprzeczne z założeniem, że $a^{p^e-1} \neq 1$. Zatem $f = e$ i element a ma rząd p^e. □

Twierdzenie 7. *Niech G będzie grupą abelową. Jeśli $g_1, g_2, \ldots g_n \in G$, oraz s_i jest rzędem elementu g_i dla $i = 1, 2, \ldots n$ oraz $NWD(s_i, s_j) = 1$ dla każdego $i, j \in N$, $i \neq j$, (mówimy, że liczby $s_1, s_2, \ldots s_n$ są parami względnie pierwsze) to rząd iloczynu $g_1 \cdot g_2 \cdot \ldots \cdot g_n$ jest równy $s_1 \cdot s_2 \cdot \ldots \cdot s_n$.*

Dowód. Wystarczy twierdzenie wykazać dla $n = 2$ i następnie zastosować indukcję względem n. Niech 1 będzie jednością grupy a k rzędem elementu $g_1 g_2$. Jeśli $(g_1 g_2)^k = 1$ to $(g_1 g_2)^(k \cdot s_1) = 1$ dla $i = 1, 2$ skąd mamy $g_1^{k \cdot s_2} = 1$ oraz $g_2^{k \cdot s_1} = 1$. Zatem $k \cdot s_2$ jest podzielne przez s_1 oraz $k \cdot s_1$ jest podzielne przez s_2.

Ponieważ $NWD(s_1, s_2) = 1$, zatem k musi być podzielne przez s_1 i s_2, a zatem również przez iloczyn $s_1 \cdot s_2$. Zatem rząd iloczynu $g_1 \cdot g_2$ czyli k dzieli się przez $s_1 \cdot s_2$. Mamy jednak $(g_1 g_2)^{s_1 \cdot s_2} = 1$ skąd wynika, że $s_1 \cdot s_2$ dzieli się przez k, a więc rząd iloczynu $g_1 g_2$ jest równy $s_1 \cdot s_2$. □

Z twierdzenia 2 wynika, że grupa multiplikatywna Z_p^* jest grupą cykliczną rzędu $p - 1$. Z kolei z twierdzenia 5 będącego bezpośrednim wnioskiem z twierdzeń 3 i 4 wynika, że w grupie Z_p^* dla każdego dzielnika d liczby $p - 1 = q_1^{k_1} \cdot q_2^{k_2} \ldots q_r^{k_r}$ istnieje element rzędu d a w szczególności element rzędu $q_i^{k_i}$.

W pętli **repeat**... **until** szukamy dla każdego $i = 1, 2, \ldots r$ elementu rzędu $q_i^{k_i}$ (stosując kryterium oparte na twierdzeniu 6). Taki element $a_i \in Z_p^*$ w grupie multiplikatywnej Z_p^* dla każdego $i = 1, 2, \ldots r$ na pewno istnieje ponieważ $q_i^{k_i}$ jest dzielnikiem rzędu grupy Z_p^* czyli liczby $p - 1 = q_1^{k_1} \cdot q_2^{k_2} \cdot \ldots q_r^{k_r}$. Każda pętla **repeat**... **until** wykrywająca element rzędu $q_i^{k_i}$ ma więc szanse by się skończyć.

Jeśli $b_i = a_i^{(p-1)/q_i^{k_i}} (\mathrm{mod}\, p) \neq 1$ to element $b_i = a_i^{(p-1)/q_i^{k_i}} (\mathrm{mod}\, p) \neq 1$ ma na mocy twierdzenia 6 rząd równy $q_i^{k_i}$.

Ponieważ liczby $q_1^{k_1}, q_2^{k_2}, q_r^{k_r}$ są parami względnie pierwsze, na mocy twierdzenia 7, dostajemy, że element (iloczyn) będący wynikiem obliczeń algorytmu czyli

$$\prod_{i=1}^{r} a_i^{(p-1)/q_i^{k_i}} (\mathrm{mod}\, p)$$

ma rząd równy $q_1^{k_1} \cdot q_2^{k_2} \cdot \ldots \cdot q_r^{k_r}$ co dowodzi poprawności algorytmu.

Liczbę generatorów skończonej grupy cyklicznej łatwo obliczyć korzystając z następującego twierdzenia.

Twierdzenie 8. *Jeśli G jest skończoną grupą cykliczną rzędu n i g jest jej generatorem tej grupy to dla każdego $k \in < 1, \#G - 1 >$:*

g^k jest generatorem grupy G wtedy i tylko wtedy $NWD(k, n) = 1$

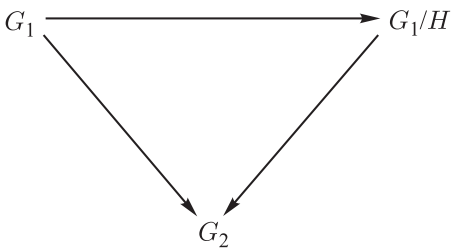
Dowód. por. C. Bagiński [3]. □

Zatem liczba wszystkich generatorów skończonej grupy cyklicznej jest równa $\varphi(\#G)$, gdzie φ jest funkcją Eulera. W szczególności jeśli grupa multiplikatywna Z_n jest cykliczna, to ma $\varphi(\varphi(n))$ generatorów. Grupa multiplikatywna Z_p^* jako grupa cykliczna o $\varphi(p) = p - 1$ elementach ma więc dokładnie $\varphi(\varphi(p)) = \varphi(p-1)$ generatorów. Iloraz $\frac{\varphi(p-1)}{p-1}$ pozwala ocenić prawdopodobieństwo wylosowania generatora przy losowaniu elementu z Z_p^* z rozkładem równomiernym.

4. Ocena średniej złożoności obliczeniowej algorytmu

Dla praktycznej użyteczności algorytmu ważna jest jego średnia złożoność obliczeniowa. Ocenimy średnią złożoność obliczeniową algorytmu z rys. 1 przy założeniu, że zmienna losowa dokonująca wyboru elementu z grupy multiplikatywnej $^*_p$ (wewnątrz pętli **repeat**... **until**) ma rozkład równomierny na Z_p^*.

Twierdzenie 9. (o homomorfizmie grup) *Niech będą dane 2 grupy G_1 i G_2, niech ponadto H będzie dzielnikiem normalnym grupy G_1 a $f : G_1 \longrightarrow G_2$ homomorfizmem grupy G_1 na grupę G_2 przy czym $H \subseteq \ker f$. Wówczas istnieje dokładnie jeden taki homomorfizm $f_0 : G_1/H \longrightarrow G_2$, że $f_0 \circ \kappa = f$, gdzie κ jest homomorfizmem kanonicznym G_1 na G_1/H. Innymi słowy istnieje dokładnie jeden homomorfizm $f_0 : G_1/H \longrightarrow G_2$, taki że diagram*



jest przemienny. W przypadku, gdy $h = \ker f$ homomorfizm f_0 jest izomorfizmem.

Dowód. por. np. A. Białynicki [9], C. Bagiński [3]. □

Z powyższego twierdzenia wynika jako prosty wniosek twierdzenie następujące.

Twierdzenie 10. *Niech grupa G_1 będzie skończona i niech $H \subseteq G_1$ będzie dzielnikiem normalnym grupy G_1. Jeśli zmienna losowa X określona na przestrzeni probabilistycznej $(\Omega, \mathfrak{M}, P)$ i o wartościach w przestrzeni mierzalnej $(G_1, 2^{G_1})$ ma rozkład równomierny na G_1 to*

1. *Zmienna losowa $\kappa \circ X$ o wartościach w przestrzeni mierzalnej $(G_1/H, \mathfrak{F})$, (gdzie $\mathfrak{F} = 2^{G_1/H}$ i $\kappa : G_1 \to G_1/H$ jest homomorfizmem kanonicznym na grupę ilorazową G_1/H) ma rozkład równomierny na grupie ilorazowej G_1/H oraz dla każdego $k = 1, 2, \ldots, \frac{\#G_1}{\#H}$ mamy:*

$$P(\kappa \circ X = H_k) = \frac{\#H}{\#G_1},$$

   gdzie H_k jest k-tą warstwą grupy ilorazowej G_1/H.

2. *Jeśli dodatkowo mamy grupę skończoną G_2 i homomorfizm $f : G_1 \to G_2$ grupy G_1 na G_2 oraz $\ker f = H$ to zmienna losowa $f(X)$ ma rozkład równomierny na G_2.*

Dowód. Część 1 tezy twierdzenia wynika z twierdzenia Lagrange'a i z faktu, że wszystkie warstwy grupy ilorazowej G_1 mają tyle samo $\#H$ elementów. Część 2-ga tezy jest bezpośrednią konsekwencją twierdzenia o homomorfizmie grup. □

Rozważmy teraz odwzorowanie $f_i : Z_p^* \to Z_p^*$ zadane dla $a \in Z_p^*$ i każdego $i \in < 1, r >$ wzorem

$$f_i(a) = a^{\frac{p-1}{q_i}} (\mathrm{mod}\, p)$$

Łatwo sprawdzić, że f_i jest homomorfizmem grupy Z_p^* w grupę Z_p^*. Istotnie, ponieważ Z_p^* jest grupą abelową mamy dla każdego $a, b \in Z_p^*$

$$\begin{aligned} f_i(a \cdot b) &= (a \cdot b)^{(p-1)/q_i} (\mathrm{mod}\, p) \\ &= (a^{(p-1)/q_i} (\mathrm{mod}\, p) \cdot b^{(p-1)/q_i} (\mathrm{mod}\, p)) (\mathrm{mod}\, p) = f_i(a) \cdot f_i(b) \end{aligned}$$

Obraz homomorficzny grupy jest grupą zatem $f_i(Z_p^*) \subseteq Z_p^*$ jest grupą a ściślej podgrupą grupy Z_p^*. Oczywiście $f_i(Z_p^*)$ jest grupą cykliczna jako podgrupa grupy cyklicznej. Łatwo ustalić rząd tej grupy, bo jeśli g jest generatorem grupy Z_p^* to elementami grupy $f_i(Z_p^*)$ będą elementy:

$$f_i(g^1), f_i(g^2), \ldots, f_i(g^{p-1})$$

czyli mamy po kolei q_i wartości

$$g^{(p-1)\cdot 1/q_i}(\mathrm{mod}\, p), g^{(p-1)\cdot 2/q_i}(\mathrm{mod}\, p), \ldots, g^{(p-1)\cdot q_i/q_i}(\mathrm{mod}\, p)$$

które są parami różne (bo g jest generatorem grupy Z_p^* a wykładniki potęg są różnymi liczbami naturalnymi należącymi do zbioru $< 1, p-1 >$).

Oczywiście ostatnia wartość: $g^{(p-1)\cdot q_i/q_i}(\mathrm{mod}\, p) = g^{p-1}(\mathrm{mod}\, p) = 1$ ponieważ $p-1$ jest rzędem grupy Z_p^*. Rozważmy teraz wszystkie pozostałe interesujące nas wartości potęg elementu $g^{(p-1)/q_i}(\mathrm{mod}\, p)$ tzn. $g^{(p-1)\cdot s/q_i}(\mathrm{mod}\, p)$ dla wykładników $s \in < q_i, p-1 >$. Dla każdego $s \in < q_i, p-1 >$ istnieją takie $k \in N$, $z \in < 0, q_i - 1 >$, że $s = z + k \cdot q_i$.

Z uwagi jednak na fakt, że $g^{(p-1)\cdot q_i/q_i}(\mathrm{mod}\, p) = g^{p-1}(\mathrm{mod}\, p) = 1$, wszystkie wartości potęg dla wykładników $s \in < q_i + 1, p-1 >$ muszą należeć do q_i elementowego zbioru

$$g^{(p-1)\cdot 1/q_i}(\mathrm{mod}\, p), g^{(p-1)\cdot 2q_i/q_i}(\mathrm{mod}\, p), \ldots, g^{(p-1)\cdot q_i/q_i}(\mathrm{mod}\, p)$$

Reasumując, $f_i(Z_p^*)$ jest podgrupą rzędu q_i grupy multiplikatywnej Z_p^*. Jeśli przyjmiemy teraz $G_1 = Z_p^*$ i $G_2 = f_i(Z_p^*)$ a jako homomorfizm grupy G_1 na G_2 przyjmiemy f_i a ponadto jeśli zmienna losowa X ma rozkład równomierny na Z_p^* to z twierdzenia 10 dostajemy, że zmienna losowa $f_i(X) = X^{(p-1)/q_i}(\mathrm{mod}\, p)$ ma rozkład równomierny na podgrupie $G_2 \subset Z_p^*$ rzędu q_i. Zatem prawdopodobieństwo

$$f_i(X) \neq 1) = P(X^{(p-1)/q_i}(\mathrm{mod}\, p) \neq 1) = \frac{q_i - 1}{q_i}$$

Prawdopodobieństwo $f_i(X) \neq 1) = \frac{q_i-1}{q_i}$ jest prawdopodobieństwem opuszczenia wewnętrznej pętli **repeat**... **until** (prawdopodobieństwem sukcesu) dokładnie zaraz po pierwszym losowaniu a.

Niech teraz $(X_j)_{j=1}^{\infty}$ będzie ciągiem niezależnych zmiennych losowych określonych na przestrzeni probabilistycznej $(\Omega, \mathfrak{M}, P)$ i o wartościach w grupie multiplikatywnej Z_p^*. Zakładamy, że wszystkie zmienne losowe ciągu mają rozkład równomierny na Z_p^* (kolejne losowania wartości $a \in Z_p^*$ są niezależne). Można powiedzieć używając języka cyfrowego przetwarzania sygnałów, że ciąg $(X_j)_{j=1}^{\infty}$ jest białym szumem dyskretnym z rozkładem równomiernym. Jest to proces stochastyczny opisujący losowanie kolejnych elementów wewnątrz pętli **repeat**... **until**. Warunki wyjścia z i-tej pętli **repeat**... **until** opisuje więc teraz proces stochastyczny:

$$(\mathrm{sgn}(f_i(X_j) - 1))_{j=1}^{\infty} \tag{3}$$

Z poprzednich rozważań wynika, że: Jeśli $\mathrm{sgn}(f_i(X_k) - 1 = 0$ to nie jest spełniony warunek wyjścia z pętli **repeat**... **until** (brak sukcesu) oraz

$$P(\mathrm{sgn}(f_i(X_k) - 1) = 0) = \frac{1}{q_i}$$

Jeśli $\mathrm{sgn}(f_i(X_k) - 1 = 1$ to jest spełniony warunek wyjścia z pętli **repeat**... **until** (sukces) oraz

$$P(\mathrm{sgn}(f_i(X_k) - 1) = 1) = \frac{q_i - 1}{q_i}$$

W algorytmie z rys. 1 wychodzimy z pętli **repeat**... **until** natychmiast jak tylko warunek wyjścia zostanie spełniony (pierwszy sukces).

Proces stochastyczny zadany wzorem (3) jest nieskończonym ciągiem prób Bernoulliego z prawdopodobieństwem sukcesu równym $\frac{q_i-1}{q_i}$. Oznaczmy przez Y_i zmienną losową o wartościach w zbiorze $N \cup \{+\infty\}$, określoną na przestrzeni probabilistycznej $(\Omega, \mathfrak{M}, P)$ zdefiniowaną następująco : dla każdego $k \in N$ i każdego $\omega \in \Omega$

$Y_i(\omega) = k$ wtedy i tylko wtedy, gdy trajektoria $(\mathrm{sgn}(f_i(X_j(\omega)-1))_{j=1}^{\infty})$ procesu $(\mathrm{sgn}(f_i(X_j - 1))_{j=1}^{\infty})$ jest taka, że: $\mathrm{sgn}(f_i(X_1(\omega) - 1)) = 0$, $\mathrm{sgn}(f_i(X_2(\omega)-1)) = 0, \ldots, \mathrm{sgn}(f_i(X_{k-1}(\omega)-1)) = 0, \mathrm{sgn}(f_i(X_k(\omega)-1)) = 1$, czyli pierwszy sukces (pierwsza jedynka) pojawia się dokładnie w chwili k-tej oraz
$Y_i(\omega) = +\infty$ wtedy i tylko wtedy, gdy trajektoria $(\mathrm{sgn}(f_i(X_j(\omega) - 1))_{j=1}^{\infty})$, jest ciągiem samych zer.

Łatwo można wykazać, że tak zdefiniowana na przestrzeni probabilistycznej $(\Omega, \mathfrak{M}, P)$ funkcja Y_i jest istotnie zmienną losową. Zmienna losowa Y_i ma rozkład geometryczny z prawdopodobieństwem sukcesu $\frac{q_i-1}{q_i}$ tzn. dla każdego $k \in N$ mamy:

$$P(Y_i = k) = (\frac{1}{q_i})^{k-1} \cdot \frac{q_i - 1}{q_i}$$

Zmienna losowa Y_i przyjmuje wartość k wtedy i tylko wtedy, gdy z i-tego wykonania wewnętrznej pętli **repeat**... **until** wychodzimy dokładnie, bezpośrednio po k-tym losowaniu wewnątrz tej pętli.

Wartość oczekiwana zmiennej losowej Y_i (czyli średni czas wyjścia z pętli) jest równa

$$\mathrm{E}(Y_i) = \frac{q_i}{q_i - 1} \leq 2$$

a wariancja

$$D^2(Y_i) = \frac{q_i}{(q_i - 1)^2} \leq 2$$

Czas wykonania wszystkich r pętli **repeat**... **until** jest opisywany zmienną losową:

$$Z = Y_1 + Y_2 + \ldots + Y_r$$

Zatem średni czas wykonania wszystkich r pętli **repeat**... **until** jest równy

$$E(Z) = \sum_{i=1}^{r} E(Y_i) = \sum_{i=1}^{r} \frac{q_i}{q_i - 1} \leq 2 \cdot r$$

Ponieważ r jest jednocześnie liczbą różnych liczb pierwszych w rozkładzie na czynniki pierwsze liczby $p - 1 = q_1^{k_1} \cdot q_2^{k_2} \cdot \ldots \cdot q_r^{k_r}$, więc

$$\log_2 p > k_1 \log_2 q_1 + k_2 \log_2 q_2 + \ldots + k_r \log_2 q_r \geq r$$

Mamy zatem $r < \log_2 p$. Wynika stąd, że średnia liczba mnożeń modulo p wewnątrz wszystkich r pętli **repeat**... **until** przy wykorzystaniu algorytmu szybkiego podnoszenia do potęgi modulo p jest rzędu $O((\log_2 p)^2)p$. Podobnie liczba mnożeń na zewnątrz pętli **repeat**... **until** przy wykorzystaniu algorytmu szybkiego podnoszenia do potęgi modulo p jest rzędu $O((\log_2 p)^2)p$. Zatem średnia złożoność całego algorytmu z rys. 1 jest rzędu $O((\log_2 p)^2)p$ jeśli za działanie dominujące uważamy mnożenie modulo p. Średnia bitowa złożoność jest natomiast rzędu $O((\log_2 p)^4)p$.

Zmienne losowe $Y_1, Y_2, \ldots, Y_r$ są niezależne, zatem wariancja zmiennej losowej

$$Z = Y_1 + Y_2 + \ldots + Y_r$$

jest równa:

$$D^2(Z) = D^2(Y_1 + Y_2 + \ldots + Y_r) = \sum_{i=1}^{r} D^2(Y_i) = \sum_{i=1}^{r} \frac{q_i}{(q_i - 1)^2}$$

Nierozstrzygnięty jest jak dotąd problem jak duży może być najmniejszy pierwiastek pierwotny dla liczby pierwszej p (chodzi o oszacowanie od dołu i od góry). Jeśli przez $r(p)$ oznaczymy najmniejszy pierwiastek pierwotny dla liczby pierwszej p to znane jest np. oszacowanie od góry: $r(p) < p^{\frac{1}{4}+\varepsilon}$ dla dowolnego $\varepsilon > 0$. Więcej informacji na ten temat i inne oszacowania dla najmniejszego pierwiastka pierwotnego z n można znaleźć w pracach A. Paszkiewicza [10] i W. Narkiewicza [4].

Jeśli zamiast liczby pierwszej p użyjemy w analizowanym algorytmie liczby n dla której istnieje pierwiastek pierwotny to algorytm pozostanie poprawny. Dane wejściowe oprócz n będzie wówczas stanowił rozkład na czynniki pierwsze liczby $\varphi(n)$.

Warto zauważyć, że często nie jest nam potrzebny generator grupy multiplikatywnej Z_p^* ale element tej grupy dostatecznie wysokiego rzędu. Z rozważań przeprowadzonych w tym podrozdziale wynika, że jeśli mamy tylko częściowy rozkład liczby $p-1$ na czynniki pierwsze tzn. wiemy, że $p-1 = q_1^{k_1} \cdot q_2^{k_2} \cdot \ldots \cdot q_r^{k_r} \cdot a$, gdzie $q_1, q_2, \ldots, q_r$ są różnymi liczbami pierwszymi oraz $k_1, k_2, \ldots k_r, a \in N$ to za pomocą opisanego algorytmu potrafimy znaleźć w grupie multiplikatywnej Z_p^* element rzędu $q_1^{k_1} \cdot q_2^{k_2} \cdot \ldots \cdot q_r^{k_r}$. Dla $a > 1$ nie będzie to generator grupy Z_p^* ale może to być element dla naszych celów dostatecznie wysokiego rzędu. Wiele algorytmów kryptograficznych zaczyna się bowiem tak: "weźmy element g dostatecznie wysokiego rzędu takiego by problem logarytmu dyskretnego przy podstawie g był praktycznie nierozwiązywalny".

5. Wnioski

Obliczona w rozdziale 4 średnia złożoność obliczeniowa analizowanego algorytmu jest zaskakująco mała. Z przedstawionego w rozdziale 3 uzasadnienia poprawności rozważanego algorytmu wyszukiwania pierwiastków pierwotnych modulo n wynika, że algorytm działa poprawnie dla dowolnej skończonej grupy cyklicznej G o ile znany jest rozkład na czynniki pierwsze rzędu grupy czyli liczby $\#G$. Również średnia złożoność obliczeniowa tego algorytmu daje się w analogiczny sposób jak w rozdziale 4 oszacować przez $O(log_2 \#G)^2$.

Spis oznaczeń

N – zbiór liczb naturalnych
Z – zbiór liczb całkowitych
$< a, b >$ – zbiór $\{k \in Z : \ a \leq k \leq b\}$, gdzie $a, b \in Z$
Z_n^* – grupa multiplikatywna pierścienia Z_n
C_n – grupa cykliczna rzędu n
$G_1 \times G_2$ – suma prosta grup G_1 i G_2
2^A – zbiór wszystkich podzbiorów zbioru A
$(\Omega, \mathfrak{M}, P)$ – przestrzeń probabilistyczna
φ – funkcja Eulera
$E(X)$ – wartość oczekiwana zmiennej losowej X

$D^2(X)$ – wariancja zmiennej losowej X
$NWD(n, m)$ – największy wspólny dzielnik n i m
$\#G$ – rząd grupy G
$< A >$ – grupa generowana przez zbiór A
$d|n$– d jest dzielnikiem n
$\ker f$ – jądro homomorfizmu f

Literatura

[1] V. Shoup, *A computational Introduction to Number Theory and Algebra*, Cambridge University Press, 2008.
[2] N. Koblitzc, *A Course in Number Theory and Cryptography*, Springer, New York 1994.
[3] C. Bagiński, *Introduction to Group Theory (in Polish)*, Script, Warszawa 2002.
[4] W. Narkiewicz, *Number Theory (in Polish)*, PWN. Warszawa, 1990.
[5] A. Menezes, P. Oorschot, S. Vanstone, *Handbook of Applled Cryptography*, CRC Press Inc., 1997.
(http://cacr.math.uwaterloo.ca/hac).
[6] D. Hankerson, A. Menezes, S. Vanstone, *Guide to Elliptic Curve Cryptography*, Springer, 2004.
[7] S. Yan; Number Theory for Computing, Springer, Berlin-Heidelberg, 2002.
[8] J. Pieprzyk, T. Hardjono, J. Seberry, *Fundamentals of Computer Security*, Springer, Berlin, Heidelberg, 2003.
[9] A. Białynicki, *Algebra*, Warszawa, PWN, 2010.
[10] A. Paszkiewicz, *Badania własności liczb pierwszych i wielomianów nieprzywiedlnych pod kątem zastosowania w telekomunikacji*, Oficyna Wydawnicza P.W.; Warszawa 2012.

THE AVERAGE COMPLEXITY OF THE PROBABILISTIC ALGORITHM FOR FINDING PRIMITIVE ROOTS MODULO n

Abstract. Primitive roots from a natural number n (i.e. generators of the multiplicative group Z_n^*) play an important role in many cryptographic algorithms like public key ciphers, digital signatures and key agreement algorithms. In the paper, proof of correctness of the probabilistic algorithm for finding primitive roots is given along with assessment of its average computational complexity. Results obtained for the multiplicative group Z_n^* can be in natural easy way generalized on the case of arbitrary finite cyclic groups.

Keywords: cryptographic algorithms, probabilistic algorithms, average computational complexity, primitive roots modulo n, cyclic groups.