DLP

Adolphe Kankeu Tamghe
papibobo@informatik.uni-bremen.de

Fachbereich Mathematik und Informatik

ALZAGK SEMINAR

Bremen, den 18. Januar 2011

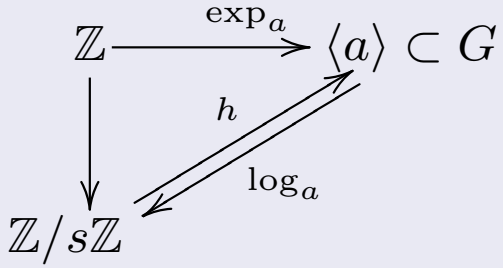
Inhaltsverzeichnis

Definition

Sei G eine Gruppe (multiplikativ geschrieben) und $a \in G$ ein Element der Ordnung s (die auch ∞ sein kann). Dann ist die **Exponentialfunktion** in G zur Basis a

$$\exp_a : \mathbb{Z} \to G, \qquad x \to a^x$$

ein Gruppenhomomorphismus und hat die Periode s. Nach dem Homomorphiesatz ist der induzierte Homomorphismus h



ein Isomorphismus. Es gibt also eine Umkehrabbildung

$$\log_a : \langle a \rangle \to \mathbb{Z}/s\mathbb{Z}$$

auf der zyklischen Untergruppe $\langle a \rangle \subset G$, den **diskreten Logarithmus** in G zur Basis a, der ein Gruppenisomorphismus ist.

Bemerkung

Der Fall $s = \infty$ passt, wenn man $s\mathbb{Z} = 0$ und $\mathbb{Z}/s\mathbb{Z} = \mathbb{Z}$ setzt.

Der wichtigste Spezialfall ist der Fall mit $G = \mathbb{F}_p$ wobei $p > 3$ und ein Primzahl ist. Insbesondere ist $\mathbb{F}_p^*$ zyklisch, d.h. es gibt ein $a \in \mathbb{F}_p$ mit $\langle a \rangle = \mathbb{F}_p^*$. Dies ist äquivalent zu Ordnung von a ist $p - 1$. Jedes $a \in \mathbb{F}_p$ deren Ordnung gleich $p - 1$ ist, nennt man Primitivwurzel modulo p. Gilt für $a, b \in \mathbb{F}_p$ und $x \in \mathbb{N}$ die Beziehung $a^x = b$, so heißt x der diskrete Logarithmus von b zur Basis.

Beispiel

Wir betrachten $p = 10^6 + 3$. Mit Hilfe von Pari und durch Probieren findet man:

- $2^x \equiv 3 \pmod{p}$ wird von $x = 254277$ gelöst
- $3^x \equiv 2 \pmod{p}$ hat keine Lösung.

Wie aber kann man diskrete Logarithmen berechnen?

Zunächst gibt es die naive Methode.

Beschreibung der naiven Methode

Ist p eine Primzahl und sind g und a gegeben mit $2 \leq g, a \leq p-1$ so probiert man für $x = 0, 1, 2, \ldots, p-1$, ob $g^x \equiv a \pmod{p}$ gilt. Genauer:

- Man setzt $g_0 = 1$ und berechnet rekursiv $g_n \equiv g_{n-1} \cdot g \pmod{p}$ für $n = 1, 2, \ldots$. (Dann ist $g^n \equiv g_n \pmod{p}$). Für jedes n testet man:
- Gilt $g_n = a$, so ist n der gesuchte diskrete Logarithmus $\log_g a$.
- Gilt $g_n = 1$, so gibt es keine Lösung und man hört auf.

Bemerkungen

Die obigen Beispielen deuten schon etwas an, dass die Berechnung diskreter Logarithmen praktisch nicht schnell geht. Zwar werden wir im Folgenden effizientere Algorithmen kennenlernen, dennoch bleibt die praktische Logarithmenberechnung ein schwieriges Problem.

Beschreibung der Pohlig-Hellman Methode

Wir geben das Verfahren zusammen mit den Begründungen an:

- Wir wollen die Gleichung $g^x = a$ für $g, a \in \mathbb{F}_p^\star$ lösen. Wir setzen voraus, dass g eine Primitivwurzel modulo p ist. Dann ist die Gleichung lösbar und x ist modulo $p-1$ bestimmt.
- Sei t ein Teiler von $p-1$. Dann gilt auch

$$g^{\frac{p-1}{t}x} = a^{\frac{p-1}{t}}.$$

Da $g^{\frac{p-1}{t}}$ Ordnung t hat, gibt es ein θ_t mit $0 \leq \theta_t < t$ und

$$g^{\frac{p-1}{t}\theta_t} = a^{\frac{p-1}{t}}$$

(θ_t kann man z.B. durch Ausprobieren finden: $\theta_t = 0, 1, 2, \ldots$) Nun hat man

$$g^{\frac{p-1}{t}\theta_t} = a^{\frac{p-1}{t}}$$

sodass mit $\text{ord}(g^{\frac{p-1}{t}}) = t$ sofort $x \equiv \theta_t \pmod{t}$ folgt.

- Wir nehmen an, wir können die Primfaktorzerlegung $p-1 = p_1^{e_1} \dots p_r^{e_r}$ von $p-1$ bestimmen. (Dann kann man auch schnell testen, ob g eine Primitivwurzel ist, denn es muss $g^{\frac{p-1}{p_i}} \neq 1$ gelten.)
- Für alle i gibt es ein x_i mit $g^{\frac{p-1}{p_i^{e_i}} x_i} = a^{\frac{p-1}{p_i^{e_i}}}$ und $0 \leq x_i < p_i^{e_i}$.
  Es gilt $x \equiv x_i \pmod{p_i^{e_i}}$. Man kann x_i z.B. durch naives Probieren von $x_i = 0, 1, 2, \dots$ bestimmen.
- Mit dem chinesischen Restsatz bestimmen wir ein $\tilde{x}$, sodass gilt

$$\tilde{x} \equiv x_i \pmod{p_i^{e_i}}, \qquad 1 \leq i \leq r.$$

  Dann folgt sofort $x \equiv \tilde{x} \pmod{p_i^{e_i}}$, also $x \equiv \tilde{x} \pmod{p-1}$, d.h. $\tilde{x}$ ist eine Lösung der Gleichung $g^x = a$. Wir sind fertig.

Beispiel $p = 10^{10} + 19$ und $g = 2$

Dann ist $p - 1 = 2 \cdot 131 \cdot 521 \cdot 73259$. Wir wollen die Gleichung $2^x \equiv 3 \pmod{p}$ lösen. Nach dem eben geschilderten Verfahren erhalten folgende 4 Gleichungen und durch Probieren die zugehörigen Lösungen.

$2^{\frac{p-1}{2} x_1} \equiv 3^{\frac{p-1}{2}} \pmod{p}, x_1 \equiv 0 \pmod{2}$

$2^{\frac{p-1}{131} x_2} \equiv 3^{\frac{p-1}{131}} \pmod{p}, x_2 \equiv 92 \pmod{131}$

$2^{\frac{p-1}{521} x_3} \equiv 3^{\frac{p-1}{521}} \pmod{p}, x_3 \equiv 223 \pmod{521}$

$2^{\frac{p-1}{73259} x_4} \equiv 3^{\frac{p-1}{73259}} \pmod{p}, x_4 \equiv 55292 \pmod{73259}$

Mit dem chinesischen Restsatz finden wir $x = 5181957398$, das $x \equiv x_i \pmod{p_i^{e_i}}$ für i erfüllt und damit die Gleichung löst.

Weitere Vereinfachungen im Fall $e_i \geq 2$:

- Sei jetzt q eines der p_i's und e das zugehörige e_i. Wir suchen eine lösung der Gleichung $g^{\frac{p-1}{q^e}\tilde{x}} = a^{\frac{p-1}{q^e}}$, wo wir $0 \leq \tilde{x} \leq q^e - 1$ annehmen können. Also gibt es eine Darstellung $\tilde{x} = y_0 + y_1 q + \ldots + y_{e-1} q^{e-1}$ mit $0 \leq y_j \leq q-1$. Durch Potenzieren mit q^{e-1} erhalten wir zunächst $g^{\frac{p-1}{q} y_0} = a^{\frac{p-1}{q}}$ wobei wir y_0 durch naives Probieren finden können. (Dies ergibt q Schritte). Durch Potenzieren der obigen Gleichung mit q^{e-j-1} erhält man

$$g^{\frac{p-1}{q^{j+1}}(y_0+y_1 q+\ldots+y_j q^j)} = a^{\frac{p-1}{q^{j+1}}}$$

und damit

$$g^{\frac{p-1}{q^{j+1}} y_j} = \left(a g^{q^{j+1}-(y_0+y_1 q+\ldots+y_{j-1} q^{j-1})}\right)^{\frac{p-1}{q^{j+1}}}$$

Mit dieser Gleichung können wir die y_j's rekursiv bestimmen: Sind $y_0, y_1, \ldots, y_{j-1}$ schon bestimmt, probieren wir für $0 \leq y_j \leq q-1$, bis wir eine Lösung haben ($\leq q$ Schritte). Damit erhalten wir schließlich

$$\tilde{x} = y_0 + y_1 q + \ldots + y_{e-1} q^{e-1} \qquad \text{mit} \quad g^{\frac{p-1}{q^e}\tilde{x}} = a^{\frac{p-1}{q^e}}$$

Schreiben wir jetzt $q = p_i, e = e_i$, so haben wir also x_i gefunden $g^{\frac{p-1}{p_i^{e_i}} x_i} = a^{\frac{p-1}{p_i^{e_i}}}$

Die Schrittzahl lässt sich hier durch $\mathcal{O}(e_i p_i)$ abschätzen. Mit dem naiven Bestimmen von $\mathcal{O}(p_i^{e_i})$

Bemerkungen

- Das obige Verfahren funktioniert nicht wenn g keine Primitivwurzel ist.(Gilt ord$(g)|\frac{p-1}{q_i}$, also $g^{\frac{p-1}{q_i}} = 1$, so liefert das Verfahren $x_i = 0$ und damit $x \equiv 0 \pmod{q_i^{e_i}}$, was natürlich nicht der Fall sein muss). Mann kann aber dann so vorgehen: Man verschafft sich eine Primitivwurzel g_0,(was leicht ist, da man $p-1$ faktorisiert hat), dann löst man mit dem Verfahren die Gleichungen $g_0^y = a$ und $g_0^z = g$. Nun haben die Äquivalenzen:

  $$g^x = a \Longleftrightarrow g_0^{xz} = g_0^y \Longleftrightarrow xz \equiv y \pmod{p-1}$$

  Die letzte Gleichung ist genau dann lösbar, wenn gilt $ggT(z, p-1)|y$. Die Lösung ist in diesem Fall auch leicht zu berechnen.
- Die Schrittzahl des Verfahrens bis zum chinesischen Restsatz ist also $\mathcal{O}(\sum e_i q_i)$. Der chinesische Restsatz ist allerdings praktisch unproblematisch. Aus dieser Abschätzung folgt auch, dass man das Verfahren nur sinnvoll verwenden kann, wenn die Gruppenordnung $p-1 = |\mathbb{F}_p^\star|$ einigermaßen glatt ist.
- Das Verfahren kann man nur starten, wenn man $p-1$ faktorisieren kann.

Bemerkung

Wir haben die Gleichungen des Typs $g^{\frac{p-1}{p_i}\hat{x}} = \hat{a}$ durch naives Probieren $x = 0, 1, \ldots p_i - 1$ gelöst, wozu man $\mathcal{O}(q_i)$ Schritte braucht. Mit dem später vorzustellenden baby-step-giant-step Algorithmus, kann man solche Gleichungen in $\mathcal{O}(\sqrt{p_i} \ln p_i)$ Schritte lösen, allerdings braucht man dazu Speicherplatz.

Überlegungen zu der baby-step-giant-step Methode

Sei p eine Primzahl und g, a Zahlen mit $1 \leq g, a \leq p-1$ (g muss hier keine Primitivwurzel sein). Wir suchen nach einer Lösung der Gleichung $g^x \equiv a \pmod{p}$.

- Sei $m = \lceil \sqrt{p} \rceil$, x eine beliebige Zahl mit $0 \leq x \leq p-2$. Wir teilen x durch m und erhalten $x = qm + r$ mit$0 \leq r \leq m-1$ und $0 \leq q \leq m-1$. Nun ist $x = (q+1)m - (m-r)$ setzt man $j = q+1$ und $i = m-r$ so erhält man eine Darstellung
$$x = mj - i \quad \text{mit} \quad 1 \leq i, j \leq m.$$
- Wir können also für unsere Logarithmenberechnung auch den Ansatz
$$g^{mj-i} \equiv a \pmod{p} \quad \text{mit} \quad 1 \leq i, j \leq m$$
machen. Dies ist aber gleichwertig mit $ag^i \equiv (g^m)^j \pmod{p}$.
- Die allgemeine Idee ist nun: Man macht zwei Listen
$$\{ag^i : i = 1, \ldots, m\} \quad \text{(baby-steps) und } \{(g^m)^j : j = 1, \ldots, m\} \quad \text{(giant-steps)}$$
und vergleicht, wann $ag^i \equiv (g^m)^j \pmod{p}$ gilt. Gibt es eine Lösung, so findet man sie auf diese Weise. Die Lösung ist dann $x = mj - i$.

Beispiel

Wir nehmen $p = 101$ und wollen die Gleichung $2^x = 3$ betrachten. Dann ist $m = 11$.
Wir beginnen mit den baby-steps:

i	1	2	3	4	5	6	7	8	9	10	11
$ag^i \pmod p$	6	12	24	48	96	91	81	61	21	42	8

Nun folgen die giant-steps:

j	1	2	3	4	5	6	7	8	9	10	11
$g^{mj} \pmod p$	28	77	35	71	69	13	61	92	51	14	89

Wir sehen, dass für $i = 8$ und $j = 7$ die Einträge gleich sind, also ist $x = mj - i = 69$ unsere gesuchte Lösung.

Hauptproblem

Wie vergleicht man die beiden Listen?

Der naive Ansatz, alle i durchlaufen lassen, dann bei festem i alle $j = 1, \ldots, m$ durchzuprobieren, führt auf eine Schrittzahl von $\mathcal{O}(p)$ hat also nichts dem naiven Berechnungsverfahren voraus. Deshalb muss man anders vorgehen.

Eine Möglichkeit ist folgende:

- Berechne $x[i] = ag^i \pmod p$ für $i = 1, \ldots, m$ und speichere die Werte.
- Sortiere $x[1], \ldots, x[m]$, d.h. bestimme eine Permutation z mit $\{z_1, \ldots, z_m\} = \{1, \ldots, m\}$, so dass gilt

$$x[z_1] \leq x[z_2] \leq \ldots \leq x[z_m].$$

  Es gibt dazu verschieden Sortierverfahren, z.B. insertion sort, heapsort, quicksort. Bei heapsort weißt man, dass das Sortierverfahren deterministisch ist mit einer Schrittzahl von $\mathcal{O}(m \ln m)$.
- Nun berechnen man für $j = 1, 2, \ldots$ den Wert $y \equiv g^{mj} \pmod p$ und schaut mit binary search nach ob y in der x-Liste vorkommt. Bei festem j sind dies $\mathcal{O}(\ln m)$ Schritte, also ist man nach $\mathcal{O}(m \ln m)$ fertig.

Pollards ρ Methode

Wir wollen wieder die Gleichung $g^x = a \pmod p$ lösen. Wir nehmen jetzt an, daß g eine Primitivwurzel modulo p ist.

1. Wir suchen $s, t \in \mathbb{N}$ mit $a^s \equiv g^t \pmod p$. Wir teilen $\{0, 1, \cdots, p-2\}$ in ungefähr drei große disjunkte Mengen S_1, S_2, S_3 ein.
   (Wir haben gewählt $S_i = \{x \in \mathbb{Z}/p, x \neq 0 : x \equiv i \pmod 3\}$.) Dann definieren wir eine Folge $x_0, x_1, x_2, \cdots \in \mathbb{F}_p^*$ durch :

$$x_0 = 1 \text{ und } x_{i+1} = \begin{cases} ax_i & \text{für } x_i \in S_1, \\ x_i^2 & \text{für } x_i \in S_2, \\ gx_i & \text{für } x_i \in S_3, \end{cases}$$

   Schreibt man $x_i = a^{e_i} g^{f_i}$, so hat man

$$e_0 = 0 \text{ und } e_{i+1} = \begin{cases} e_i + 1 \pmod{p-1} & \text{für } x_i \in S_1, \\ 2e_i \pmod{p-1} & \text{für } x_i \in S_2, \\ e_i \pmod{p-1} & \text{für } x_i \in S_3, \end{cases}$$

$$f_0 = 0 \text{ und } f_{i+1} = \begin{cases} f_i + 1 \pmod{p-1} & \text{für } x_i \in S_3, \\ 2f_i \pmod{p-1} & \text{für } x_i \in S_2, \\ f_i \pmod{p-1} & \text{für } x_i \in S_1, \end{cases}$$

Der Vorteil ist nun, daß man die Folge $x_0, x_1, \cdots$ nicht speichern muss. Man berechnet einfach sukzessiv für $i = 0, 1, 2, \cdots$

$$(x_i, e_i, f_i, x_{2i}, e_{2i}, f_{2i})$$

und testet dann, ob $x_i = x_{2i}$ gilt.
Ist aber $x_i = x_{2i}$, so haben wir mit $s = e_i - e_{2i} \pmod{p-1}$ und $t = f_{2i} - f_i \pmod{p-1}$ eine Lösung von $a^s = g^t$.

2. Wir nehmen an, daß wir eine Relation $a^s \equiv g^t \pmod{p}$ gefunden haben. Mit dem euklidischen Algorithmus berechnen wir $d = ggT(s, p-1)$ und eine Darstellung $d = us + v(p-1)$. Dann gilt, wenn wir $g^x = a$ ansetzen:

$$g^{xd} = a^d = (a^s)^u = (g^t)^u = g^{tu}$$

Da g eine Primitivwurzel sein sollte, folgt $xd \equiv tu \pmod{p-1}$. Also gibt es ein j mit $xd = tu + j(p-1)$. Dann folgt auch $d|tu$ und damit

$$x = \frac{tu}{d} + j\frac{p-1}{d}.$$

Wir können also $j \in \{0, 1, \cdots, d-1\}$ annehmen. Durch Ausprobieren von $j = 0, 1, \cdots, d-1$ erhalten wir dann die Lösung.

Beispiel

$p = 113, g = 3, a = 5$. Die Folge $(x_i, e_i, f_i, x_{2i}, e_{2i}, f_{2i})$ sieht dann so aus:

$$\begin{aligned} i = 0 &: (1, 0, 0, 1, 0, 0) \\ i = 1 &: (5, 1, 0, 25, 2, 0) \\ i = 2 &: (25, 2, 0, 36, 3, 1) \\ i = 3 &: (12, 3, 0, 98, 3, 3) \\ i = 4 &: (36, 3, 1, 108, 7, 6) \\ i = 5 &: (108, 3, 2, 112, 14, 14) \\ i = 6 &: (98, 3, 3, 98, 15, 15) \end{aligned}$$

Nach 6 Schritten erhält man $s = e_6 - e_{12} = 100 \pmod{112}$ und $t = f_{12} - f_6 = 12 \pmod{112}$, so daß $a^s \equiv g^t \pmod{p}$ gilt. Nun ist $ggT(s, p-1) = 4$ und $4 = us + v(p-1)$ mit $u = 9$ und $v = -8$. Der Ansatz $g^x = a$ ergibt dann

$$g^{4x} = a^4 = a^{us} = a^{900} = a^{108} \pmod{p},$$

was mit $4x \equiv 108 \pmod{112}$, was wiederum mit $x \equiv 27 \pmod{28}$ äquivalent ist. Nun wird probiert: $x = 27, x = 27 + 28 = 55, x = 1 + 3.28 = 83$ und dann stellt fest, daß $x = 83$ eine Lösung ist.

Die Index-Kalkül

Dieser Algorithmus funktioniert im Gegensatz zu den oberen Algorithmen nur für die multiplikative Gruppe von endlichen Körpern mit kleiner Charakteristik und großem Erweiterungsgrad, d.h. für $\mathbb{F}_q^*$ mit $q = p^n$ wobei p klein ist. Es ist nicht bekannt, wie man diesen Algorithmus z.B. für Elliptische Kurven anpassen kann. Deshalb kann man den Sicherheitsparameter in Kryptosystemen, die auf elliptischen Kurven basieren, deutlich geringer wählen.
Es gelte

$$\mathbb{F}_{p^n} \cong \mathbb{F}_p[x]/(f)$$

wobei $f \in \mathbb{F}_p[x]$, $\deg(f) = n$ und f irreduzibel. D.h. zu einem Element $a \in \mathbb{F}_{p^n}$ gehört ein Polynom $a(x)$ mit $grad(a(x)) < n$. Für Elemente $c(x) \in \mathbb{F}_{p^n}$ mit $c \in \mathbb{F}_p$ kann der diskrete Logarithmus bezüglich eines Generators $g \in \mathbb{F}_{p^n}$ leicht berechnet werden, da p nach Voraussetzung klein ist. Sei $\langle g \rangle = \mathbb{F}_{p^n}^*$, dann ist

$$g^{\frac{p^n-1}{p-1}} \in \mathbb{F}_p^* \text{ denn } \left(g^{\frac{p^n-1}{p-1}}\right)^{p-1} = 1$$

und

$$\left\langle g^{\frac{p^n-1}{p-1}} \right\rangle = \mathbb{F}_p^*$$

D.h. um die diskreten Logarithmen von Elementen $c \in \mathbb{F}_p$ zu erhalten, kann man eine Tabelle mit allen Elementen

$$\left\langle g^{\frac{p^n-1}{p-1} \cdot j} \right\rangle \qquad \text{mit } j \in \{0, \cdots, p-2\}$$

berechnen.

Funktionsweise des Algorithmus

Wähle eine Faktorbasis $B \subset \mathbb{F}_p[x]$. Z.B. $B = \{a \in \mathbb{F}_p[x] \,|grad(a) \leq k, a \text{ irreduzible }\}$ für eine geeignete Konstante k. Diese Elemente werden im weiteren als Elemente in $\mathbb{F}_{p^n}$ aufgefasst.

1- In der ersten Stufe sollen die diskrete Logarithmen von aller Elemente von B bestimmt werden.

- Wähle $t \in \{1, \cdots, p^n - 2\}$
- Berechne $c = g^t$, also $c(x) = g(x)^t \pmod{f(x)}$
- Teste, ob das Element $c(x)$ ein Produkt von Elementen $b(x) \in B$ ist, also über den Faktorbasis B zerfällt.

War der obige Test erfolgreich, gibt es Exponenten $\alpha_b \in \mathbb{N}$ und eine Konstante $c_0 \in \mathbb{F}_p$, so dass

$$c(x) = \left(\sum_{b(x) \in B} b(x)^{\alpha_b} \right) \cdot c_0$$

Dann gilt

$$\log_g c(x) = \log_g \left[\left(\sum_{b(x) \in B} b(x)^{\alpha_b} \right) \cdot c_0 \right].$$

- und damit

$$t = \log_g c_0 + \sum_{b(x) \in B} \left(\alpha_b \log_g b(x) \right)$$

Für jedes $c(x) \in \mathbb{F}_p [x] / (f)$, dass über der Faktorbasis zerfällt, erhält man eine lineare Gleichung in den gesuchten Werten $\log_g b(x)$. Für genügend viele Elemente $c(x)$, die über B zerfallen, können diese Werte daher durch Lösen des Gleichungssystems berechnet werden.

2- In der zweiten Stufe wird $\log_g a$ für ein gegebenes Element $a \in \mathbb{F}_{p^n}^*$ berechnet.

- Wähle $t \in \{1, \cdots, p^n - 2\}$.
- Berechne $y = a \cdot g^t$, also $y(x) = a(x) \cdot g(x)^t \pmod f$.
- Teste, ob $y(x)$ über der Faktorbasis zerfällt
- Wenn nicht, wähle neues $t \in \{1, \cdots, p^n - 2\}$

War der obige Test erfolgreich, gibt es Exponenten $\beta_b \in \mathbb{N}$ und eine Konstante $y_0 \in \mathbb{F}_p$, so dass

$$y(x) = \left(\sum_{b(x) \in B} b(x)^{\beta_b} \right) \cdot y_0$$

Dann gilt

$$\log_g y(x) = \log_g \left[\left(\sum_{b(x) \in B} b(x)^{\beta_b} \right) \cdot y_0 \right] \text{ und } \log_g y(x) = log_g a + t.$$

und daher

$$\log_g a = \log_g y_0 + \sum_{b(x)\in B} \left(\beta_b \log_g b(x)\right) - t$$

da die Werte von $\log_g b(x)$ nach der ersten Stufe bekannt sind, ist somit $\log_g a$ berechnet.

Beispiel

Wir wollen hier mit dem Index-Kalkül Verfahren den Logarithmus von $X^3 + X^2 + 2$ in

$$\mathbb{F}_{3^5} = \mathbb{F}_3 [X] / \left(X^5 - X + 1\right)$$

zur Basis X. Wir nutzen als Faktorbasis

$$B = \{X, X + 1, X - 1\}$$

und als Zufallszahlen im ersten Schritt $t = \{20, 22, 23\}$ und im zweiten Schritt $t = 9$.
Lösung:
Es ist $f = X^5 - X + 1$ und $B = \{X, X + 1, X - 1\}$ gegeben. Es soll der Logarithmus von $a = X^3 + X^2 + 2$ bestimmt werden. Zuerst bestimmt man die Logarithmen der Elemente in $\mathbb{F}_3^*$, d.h. von 1 und -1. Diese sind

$$\log_g(1) = 0 \text{ und } \log_g(-1) = 121.$$

Jetzt kommen die zwei Stufen des Index-Kalkül Verfahrens.

1. Im ersten Schritt sollen die Logarithmen der Faktorbasis bestimmt werden. Wir berechnen

$$\begin{aligned} g^{20} &= (X-1)^4 \pmod f \\ g^{22} &= -(X+1)^2 (X-1) \pmod f \\ g^{23} &= -X(X+1)^2 (X-1) \pmod f \end{aligned}$$

Daraus folgen die (linearen) Gleichungen

$$\begin{aligned} 20 &= 4\log_g(X-1) \\ 22 &= 121 + 2\log_g(X+1) + \log_g(X-1) \\ 23 &= 121 + 1 + 2\log_g(X+1) + \log_g(X-1) \end{aligned}$$

und daraus $\log_g(X-1) = 5$ und $\log_g(X+1) = 69$.

2. Im zweiten Schritt berechnet man

$$ag^{19} = X^3 + 2X^2 = X^2\,(X-1)$$

und damit folgt

$$\log_g(a) + 19 = 2 + 5 \text{ und schließlich } \log_g(a) = 230 \pmod{242}.$$

Bemerkung

Eine wesentliche Vorarbeit im Index-Kalkül Verfahren ist der 1. Schritt. Man muß k, die Anzahl der Basisprimzahlen geeignet wählen. Ist k zu klein, findet man schwer Relationen, ist k zu groß muß man viele Relationen erzeugen, außerdem wird das Gleichungssystem im 2. Schritt schwierig.
Hat man die ersten beiden Schritte erledigt, lassen sich diskrete Logarithmen bei festem p und g schnell finden. Dies ist anders als bei anderen Verfahren.