10 Problema del Logaritmo Discreto

Il **Problema Del Logaritmo Discreto (PLD)**, che è alla base di numerosi crittosistemi che vedremo nel corso, è definito come segue:

Problema 10.1. (Logaritmo Discreto)
Istanza: Siano $(G, \cdot)$ un gruppo moltiplicativo, $\alpha \in G$ di ordine n e $\beta \in \langle \alpha \rangle$.
Domanda: Determinare l'unico intero a, con $0 \leq a \leq n-1$ tale che

$$\alpha^a = \beta.$$

Siffatto intero a si dice logaritmo discreto di β in base α e lo si denota con $\log_\alpha \beta$.

L'utilità del **PLD** in Crittografia risiede nel fatto che determinare logaritmi discreti è (probabilmente) computazionalmente difficile, mentre l'operazione inversa, ovvero l'elevamento a potenza, è calcolata efficientemente attraverso il metodo dei quadrati ripetuti. Cioè,

$$F : \{0, ..., n-1\} \longrightarrow \langle \alpha \rangle, a \longmapsto \alpha^a$$

è una funzione **one-way**.

Nel 1985, T. Elgamal propose il seguente crittosistema basato sul PLD in $(\mathbb{Z}_p^*, \cdot)$, p primo.

Figura 10.1: Taher Elgamal (1955)

Definizione 10.2. **(Crittosistema di Elgamal)**
Sia p un primo tale che il PLD in $(\mathbb{Z}_p^*, \cdot)$ è computazionalmente intrattabile e sia α un elemento primitivo di $\mathbb{Z}_p^*$. Inoltre siano:

1. $\mathcal{P} = \mathbb{Z}_p^*$;
2. $\mathcal{C} = \mathbb{Z}_p^* \times \mathbb{Z}_p^*$;
3. $\mathcal{K} = \{(p, \alpha, a, \beta) : \alpha^a = \beta\}$, dove (p, α, β) è pubblica, mentre a è segreta.
4. Per $K = (p, \alpha, a, \beta)$ ed un elemento random (segreto) $k \in \mathbb{Z}_{p-1}$ vale che

$$\begin{aligned} e_{K,k} &: \mathbb{Z}_p^* \longrightarrow \mathbb{Z}_p^* \times \mathbb{Z}_p^*, x \longmapsto (y_1, y_2) = (\alpha^k \ mod\ p, x\beta^k \ mod\ p) \\ d_K &: \mathbb{Z}_p^* \times \mathbb{Z}_p^* \longrightarrow \mathbb{Z}_p^*, (y_1, y_2) \longmapsto y_2 (y_1^a)^{-1} \ mod\ p. \end{aligned}$$

Si noti che la cifratura nel crittosistema di Elgamal è randomizzata in quanto dipende, oltra che da x, anche da k. Quindi un testo in chiaro x può essere cifrato in $p-1$ modi diversi (pari al numero degli elementi di $\mathbb{Z}_{p-1}$).

Esempio 10.3. Sia $p = 2579$, siccome $\frac{p-1}{2} = 1289$ allora $2^{1289} \equiv \left(\frac{2}{2579}\right) \ mod\ 2579$ e quindi $2^{1289} \equiv -1 \ mod\ 2579$ essendo $2579 \equiv 3 \ mod\ 8$. Pertanto, $\alpha = 2$ è un elemento primitivo di $\mathbb{Z}_{2579}^*$. Sia $a = 765$, quindi $\beta = 2^{765} \ mod\ 2579 = 949$. Quindi

$$K = (2579, 2, 756, 949)$$

Supponiamo che un utente voglia trasmettere l'unità di messaggio $x = 1299$. Sia $k = 853$ l'intero generato casualmente, allora

$$\begin{aligned} y_1 &= 2^{853} \ mod\ 2759 = 435 \\ y_2 &= 1299 \times 949^{853} \ mod\ 2759 = 2396 \\ e_{K,853}(1299) &= (435, 2396) \\ d_K(435, 2396) &= 2396 \times (435^{765})^{-1} \ mod\ 2759 = 1299 = x. \end{aligned}$$

□

Il Crittosistem di Elgamal è dimostrabilmente sicuro: infatti fonda la sicurezza sull'intrattabilità computazione del problema del logaritmo discreto in $\mathbb{Z}_p^*$. Infatti, non ci sono algoritmi aventi complessità computazionale di tipo polinomiale, se p ha circa 300 cifre e $p-1$ ha un fattore primo elevato, e se α è un elemento primitivo modulo p.

10.1 L'algoritmo di Shanks

Il seguente algoritmo, dovuto a Shanks, rappresenta un compromesso tra memory utilizzata e tempo impiegato per il calcolo del Logaritmo discreto nel generico gruppo ciclico.

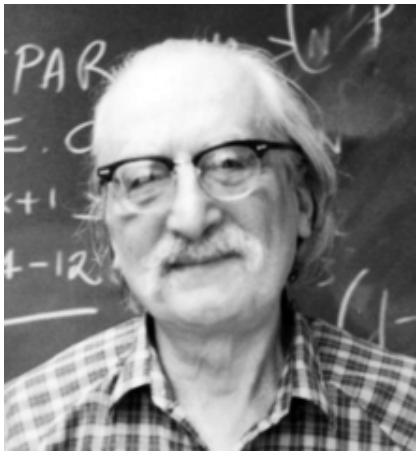

Figura 10.2: Daniel Shanks (1917 – 1996)

Sia $\alpha \in G$, con $o(\alpha) = n$ e sia $\beta \in \langle\alpha\rangle$. Allora l'**Algoritmo di Shanks** opera come segue

Algoritmo 10.4. **(Shanks**(G, n, α, β)**)**

$m \longleftarrow \lceil\sqrt{n}\rceil$

for $j \longleftarrow 0$ **to** $m-1$
do calcola α^{mj}

Crea una lista L_1 delle m coppie (j, α^{jm}) ordinate rispetto alla seconda coordinata.

for $i \longleftarrow 0$ **to** $m-1$
do calcola $\beta\alpha^{-i}$

Crea una lista L_2 delle m coppie $(i, \beta\alpha^{-i})$ ordinate rispetto alla seconda coordinata.

Cerca le coppie in L_1 e in L_2 aventi la stessa seconda coordinata. Cioè se $(j_0, y) \in L_1$ e $(i_0, y) \in L_2$.

$\log_\alpha \beta \longleftarrow j_0 m + i_0$.

Se $(j_0, y) \in L_1$ e $(i_0, y) \in L_2$, allora

$$\alpha^{j_0 m} = y = \beta\alpha^{-i_0} \Longrightarrow \beta = \alpha^{j_0 m + i_0} \Longrightarrow \log_\alpha \beta = j_0 m + i_0.$$

Viceversa, se dividiamo $0 \le \log_\alpha \beta \le n-1$ per m esistono degli interi i_0, j_0 con $0 \le i_0, j_0 \le m-1$ tali che

$$\log_\alpha \beta = j_0 m + i_0.$$

Si noti che $\sqrt{n} \le m$, allora

$$\log_\alpha \beta \le n - 1 \le m^2 - 1 = (m-1)m + m - 1$$

e quindi $j_0 \le m-1$. Pertanto l'algoritmo si Shanks ha successo se, e solo se, $\beta \in \langle\alpha\rangle$.

La complessità computazionale dell'Algoritmo di Shanks è

$$O(\lceil\sqrt{n}\rceil \ln \lceil\sqrt{n}\rceil).$$

Esempio 10.5. Determiniamo $\log_3 525$ in $\mathbb{Z}^*_{809}$.

Siccome 809 è primo, $808 = 2^3 101$ e $\left(\frac{3}{809}\right) = \left(\frac{2}{3}\right) = -1$, allora $3^{404} \equiv -1 \ mod\ 809$. Pertanto, $\alpha = 3$ è un elemento primitivo di $\mathbb{Z}^*_{809}$. Allora $n = 809$, quindi

$$m = \lceil\sqrt{809}\rceil = \lceil 28,\ 443\rceil = 29$$
$$\alpha^{29} \ mod\ 809 = 99$$

Allora calcoliamo $(j, 99^j)$ con $0 \le j \le 28$ ottenendo così

$(0,1)$	$(1,99)$	$(2,93)$	$(3,308)$	$(4,559)$
$(5,329)$	$(6,211)$	$(7,664)$	$(8,207)$	$(9,268)$
$(10,644)$	$(11,654)$	$(12,26)$	$(13,147)$	$(14,800)$
$(15,727)$	$(16,681)$	$(17,464)$	$(18,632)$	$(19,275)$
$(20,528)$	$(21,496)$	$(22,564)$	$(23,15)$	$(24,676)$
$(25,586)$	$(26,575)$	$(27,295)$	$(28,81)$	

che ordinate rispetto alla seconda coordinata producono la lista L_1.
Ora, siccome $3^{-1} \equiv 270 \ mod\ 809$ calcoliamo $(i, 525 \times 270^i)$ con $0 \le i \le 28$. Allora

$(0,525)$	$(1,175)$	$(2,328)$	$(3,379)$	$(4,396)$
$(5,132)$	$(6,44)$	$(7,554)$	$(8,724)$	$(9,511)$
$(10,440)$	$(11,686)$	$(12,768)$	$(13,256)$	$(14,355)$
$(15,388)$	$(16,399)$	$(17,133)$	$(18,314)$	$(19,644)$
$(20,754)$	$(21,521)$	$(22,713)$	$(23,777)$	$(24,259)$
$(25,356)$	$(26,658)$	$(27,489)$	$(28,163)$	

che ordinate rispetto alla seconda coordinata producono la lista L_2.
A questo punto si vede che $(10,644) \in L_1$ e $(19,644) \in L_2$, quindi

$$\log_3 525 = (29 \times 10 + 19) \ mod\ 809 = 309.$$

E' facile vedere che effettivamente $3^{309} \equiv 525 \ mod\ 809$.

□

10.2 L'algoritmo Rho di Pollard

Siano $(G,\cdot)$ un gruppo $\alpha \in G$ di ordine n e sia $\beta \in \langle\alpha\rangle$. Siccome $\langle\alpha\rangle$ è ciclico di ordine n, allora possiamo considerare $\log_\alpha \beta$ come un elemento di $\mathbb{Z}_n$.

Sia $\{S_1, S_2, S_3\}$ un partizione di G in parti della stessa grandezza circa. Quindi, $G = S_1 \biguplus S_2 \biguplus S_3$. Sia

$$\begin{aligned} f : \langle\alpha\rangle \times \mathbb{Z}_n \times \mathbb{Z}_n &\longrightarrow \langle\alpha\rangle \times \mathbb{Z}_n \times \mathbb{Z}_n, \\ (x,a,b) &\longmapsto \begin{cases} (\beta x, a, b+1) & \text{se } x \in S_1 \\ (x^2, 2a, 2b) & \text{se } x \in S_2 \\ (\alpha x, a+1, b) & \text{se } x \in S_3. \end{cases} \end{aligned} \tag{10.1}$$

Lemma 10.6. *Se (x,a,b) è tale che $x = \alpha^a\beta^b$, allora $(x',a',b') = f(x,a,b)$ è tale che $x' = \alpha^{a'}\beta^{b'}$*

Dimostrazione. Vale che

$$\begin{aligned} x \in S_1 &\Longrightarrow \beta x = \beta(\alpha^a\beta^b) = \alpha^a\beta^{b+1} \\ x \in S_2 &\Longrightarrow x^2 = (\alpha^a\beta^b)(\alpha^a\beta^b) = \alpha^{2a}\beta^{2b} \\ x \in S_3 &\Longrightarrow \alpha x = \alpha(\alpha^a\beta^b) = \alpha^{a+1}\beta^b. \end{aligned}$$

Pertanto, l'asserto.

□

Quindi, definiamo per ricorrenza

$$(x_i,a_i,b_i) = \begin{cases} (1,0,0) & \text{se } i = 0 \\ f(x_{i-1},a_{i-1},b_{i-1}) & \text{se } i \geq 1 \end{cases} \tag{10.2}$$

Osservazione 10.7.

- Si noti che $1 = \alpha^0\beta^0$, quindi, per il **Lemma 10.6**, $x_i = \alpha^{a_i}\beta^{b_i}$ per ogni $i \geq 1$.
- Si noti che, per ipotesi $1 \notin S_2$, altrimenti la successione (10.2) diventa costante di costante valore $(1,0,0)$.

Lemma 10.8. *Se (x_i,a_i,b_i) e (x_j,a_j,b_j) sono tali che soddisfano $x_i = x_j$ per ogni $0 \leq i < j \leq n-1$, allora esiste $0 \leq i' \leq n-1$ tale che $(x_{i'},a_{i'},b_{i'})$, $(x_{2i'},a_{2i'},b_{2i'})$ soddisfano $x_{i'} = x_{2i'}$.*

Dimostrazione. Segue dalla definizione di f in (10.1) che se $x_i = x_j$, allora

$$x_{i+\delta} = x_{j+\delta} \text{ per ogni } \delta \in \mathbb{N}. \tag{10.3}$$

Sia $\ell = j - i$, poichè $\{i, i+1, \cdots, j-1\}$ è un sistema completo di residui modulo ℓ, esiste $i' \in \{i, i+1, \cdots, j-1\}$ tale che $i' = k_0\ell$. Sia $\delta = i' - i$, allora vale

$$x_{i'} = x_{i'+\ell}.$$

Ora applicando (10.3) con i' al posto di i, $i' + \ell$ al posto di j ed ℓ al posto di δ, si ha $x_{i'+\ell} = x_{i'+2\ell}$. Pertanto, $x_{i'} = x_{i'+2\ell}$ e, induttivamente,

$$x_{i'} = x_{j'} \text{ con } j' = i' + k\ell, k \in \mathbb{N}.$$

Se $k = k_0$, allora $j' = 2i'$ e quindi vale

$$x_{i'} = x_{2i'}.$$

□

Lemma 10.9. *Se (x_i, a_i, b_i) e (x_{2i}, a_{2i}, b_{2i}) sono tali che soddisfano $x_i = x_{2i}$ e $\gcd(b_{2i} - b_i, n) = 1$, allora*

$$\log_\alpha \beta = (a_i - a_{2i})\,(b_{2i} - b_i)^{-1} \ \ mod\ n.$$

Dimostrazione. Sia $c = \log_\alpha \beta$. Se $x_i = x_{2i}$, allora $\alpha^{a_i}\beta^{b_i} = \alpha^{a_{2i}}\beta^{b_{2i}}$ e quindi $\alpha^{a_i + cb_i} = \alpha^{a_{2i} + cb_{2i}}$. Pertanto,

$$a_i + cb_i \equiv a_{2i} + cb_{2i} \ \ mod\ n.$$

da cui si ricava $c \equiv (a_i - a_{2i})\,(b_{2i} - b_i)^{-1} \ \ mod\ n$, essendo $\gcd(b_{2i} - b_i, n) = 1$. □

Abbiamo provato che se si determinano due terne (x_i, a_i, b_i) e (x_{2i}, a_{2i}, b_{2i}) tali che $x_i = x_{2i}$ e $\gcd(b_{2i} - b_i, n) = 1$, allora si determina il logaritmo discreto.

Il caso in cui $\gcd(b_{2i} - b_i, n) > 1$ non è così intrattabile.
Infatti se $\gcd(b_{2i} - b_i, n) = d$ allora

$$c(b_{2i} - b_i) \equiv (a_i - a_{2i}) \ \ mod\ n$$

ha esattamente d possibili soluzioni. Se d non è troppo grande, è relativamente facile trovarle e vedere quale tra queste è quella che rappresenta il logaritmo discreto.

Esempio 10.10. L'intero $p = 809$ e si vede facilmente che $\alpha = 89$ ha ordine $n = 101$ in $\mathbb{Z}^*_{809}$. Inoltre $\beta = 618$ appartiene a $\langle\alpha\rangle$. Vogliamo, quindi, determinare $\log_{89} 618$.

Definiamo gli insiemi

$$\begin{aligned} S_1 &= \{x \in \mathbb{Z}^*_{809} : x \equiv 1 \ \ mod\ 3\} \\ S_2 &= \{x \in \mathbb{Z}^*_{809} : x \equiv 0 \ \ mod\ 3\} \\ S_3 &= \{x \in \mathbb{Z}^*_{809} : x \equiv 2 \ \ mod\ 3\} \end{aligned}$$

e consideriamo la $f : \langle\alpha\rangle \times \mathbb{Z}_{101} \times \mathbb{Z}_{101} \longrightarrow \langle\alpha\rangle \times \mathbb{Z}_{101} \times \mathbb{Z}_{101}$ definita come in (10.1).

Allora

i	(x_i, a_i, b_i)	(x_{2i}, a_{2i}, b_{2i})
1	$(618, 0, 1)$	$(76, 0, 2)$
2	$(76, 0, 2)$	$(113, 0, 4)$
3	$(46, 0, 3)$	$(488, 1, 5)$
4	$(113, 0, 4)$	$(605, 4, 10)$
5	$(349, 1, 4)$	$(422, 5, 11)$
6	$(488, 1, 5)$	$(683, 7, 11)$
7	$(555, 2, 5)$	$(451, 8, 12)$
8	$(605, 4, 10)$	$(344, 9, 13)$
9	$(451, 5, 10)$	$(112, 11, 13)$
10	$(422, 5, 11)$	$(422, 11, 15)$

La prima collisione del tipo $x_h = x_{2h}$ è $x_{10} = x_{20} = 422$, allora

$$\log_{89} 618 = (5-11)(15-11)^{-1} \ mod\ 101 = (-6 \times 76) \ mod\ 101 = 49.$$

Pertanto, $\log_{89} 618 = 49$ in $\mathbb{Z}^*_{809}$. Infatti $89^{49} \equiv 618 \ mod\ 809$.

□

Supponiamo come sempre che $(G, \cdot)$ un gruppo $\alpha \in G$ di ordine n e sia $\beta \in \langle \alpha \rangle$.

Algoritmo 10.11. (Rho di Pollard per il Logaritmo Discreto(G, n, α, β))

Procedura $f(x, a, b)$
if $x \in S_1$
then $f \longleftarrow (\beta \cdot x, a, (b+1) \ mod\ n)$
else if $x \in S_2$
then $f \longleftarrow (x^2, (2 \cdot a) \ mod\ n, (2 \cdot b) \ mod\ n)$
else $f \longleftarrow (\alpha \cdot x, (a+1) \ mod\ n, b)$
return (f)
main
definiamo la partizione $\{S_1, S_2, S_3\}$ di G
$(x, a, b) \longleftarrow (1, 0, 0)$
$(x', a', b') \longleftarrow f(x, a, b)$
while $x \neq x'$
do $\begin{cases} (x, a, b) \longleftarrow f(x, a, b) \\ (x', a', b') \longleftarrow f(x', a', b') \\ (x', a', b') \longleftarrow f(x', a', b') \end{cases}$
if $\gcd(b' - b, n) \neq 1$
then return ("Insuccesso)
else return $((a - a')(b' - b)^{-1} \ mod\ n)$

Si prova che, sotto opportune ipotesi relative alla casualità di f, l'**Algoritmo 10.11** ha compessità $O(\sqrt{n})$.

10.3 L'algoritmo di Pohlig-Hellmann

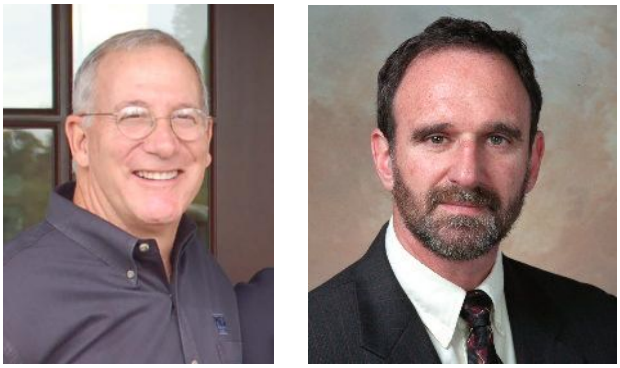

Figura 10.3: Stephen Pohlig e Martin Edward Hellman

Siano $(G,\cdot)$ e α un suo elemento di ordine n. Sia ora $\beta \in \langle\alpha\rangle$ il valore $c = \log_\alpha \beta$ è univocamente determinato modulo n. Infatti, $\alpha^{c+hn} = \alpha^c = \beta$. Pertanto, ai fini della determinazione del logaritmo discreto è sufficiente determinare $c \ mod \ n$.

Sia

$$n = \prod_{i=1}^{\ell} p_i^{k_i}$$

la fattorizzazione completa di n. Per determinare $c \ mod \ n$ grazie al **Teorema Cinese dei Resti** è sufficiente determinare $c \ mod \ p_i^{k_i}$ per ogni $i = 1, .., \ell$. Quindi si p uno dei divisori primi di n e sia p^k la massima potenza di p che divide n.

Vogliamo determinare

$$x \equiv c \ mod \ p^k$$

dove $0 \le x \le p^k - 1$. Rappresentiamo x in base p. Allora

$$x = \sum_{i=0}^{k-1} x_i p^i, \text{ dove } 0 \le x_i \le p-1 \text{ per } 0 \le i \le k-1.$$

Pertanto l'obbiettivo è determinare tutti gli x_i (e quindi x). Per fare ciò esprimiamo c come segue

$$c = x + \mu p^k, \quad \mu \in \mathbb{Z}$$

Quindi,

$$c = \sum_{j=0}^{k-1} x_j p^j + \mu p^k.$$

Lemma 10.12. *Sia*

$$\beta_j = \begin{cases} \beta & \text{se } j = 0 \\ \beta\alpha^{-(x_0+x_1p+\cdots x_{j-1}p^{j-1})} & \text{se } 1 \le j \le k-1 \end{cases}$$

allora

$$\beta_j^{n/p^{j+1}} = \alpha^{x_j n/p} \text{ per } 0 \le j \le k-1. \tag{10.4}$$

Dimostrazione. Se $j = 0$, risulta

$$\begin{aligned} \beta_j^{n/p^{j+1}} &= \beta^{n/p} = (\alpha^c)^{n/p} = \left(\alpha^{x_0+x_1p+\cdots+x_{k-1}p^{k-1}+\mu p^k}\right)^{n/p} = \\ &= \left(\alpha^{x_0+M_0p}\right)^{n/p} = (\alpha^{x_0})^{n/p} + \left(\alpha^{M_0p}\right)^{n/p} = \alpha^{x_0 n/p}\alpha^{M_0 n} = \alpha^{x_0 n/p}. \end{aligned}$$

Se $j \ge 1$, vale che

$$\begin{aligned} \beta_j^{n/p^{j+1}} &= \left(\beta\alpha^{-(x_0+x_1p+\cdots+x_{j-1}p^{j-1})}\right)^{n/p^{j+1}} = \left(\alpha^c\alpha^{-(x_0+x_1p+\cdots+x_{j-1}p^{j-1})}\right)^{n/p^{j+1}} = \\ &= \left(\alpha^{c-(x_0+x_1p+\cdots x_{j-1}p^{j-1})}\right)^{n/p^{j+1}} = \left(\alpha^{x_jp^j+\cdots x_{k-1}p^{k-1}+\mu p^k}\right)^{n/p^{j+1}} = \\ &= \left(\alpha^{x_jp^j+M_jp^{j+1}}\right)^{n/p^{j+1}} = \left(\alpha^{x_jp^j}\right)^{n/p^{j+1}}\left(\alpha^{M_jp^{j+1}}\right)^{n/p^{j+1}} = \\ &= \alpha^{x_j n/p}\alpha^{M_j n} = \alpha^{x_j n/p}. \end{aligned}$$

□

Come diretta conseguenza del **Lemma 10.12** si ha

$$\beta_{j+1} = \beta_j\alpha^{-x_jp^j} \text{ per } 1 \le j \le k-1. \tag{10.5}$$

Sulla base delle precedenti osservazioni, l'**Algoritmo di Pohlig-Hellmann** opera come segue

1. Determina x_0 a partire da (10.4) per $j = 0$, ovvero da $\beta^{n/p} = \beta_0^{n/p} = \alpha^{x_0 n/p}$. Partendo da $\gamma = \alpha^{n/p}$ si calcola $\gamma^2, \gamma^3, \ldots$ fino a determinare $i \le p-1$ tale che $\gamma^i = \beta^{n/p}$. Quando questo accade, allora $x_0 = i$ (si noti che $x_0 = \log_{\alpha^{n/p}} \beta^{n/p}$).
2. Si calcola attraverso (10.5), ovvero $\beta_1 = \beta_0\alpha^{-x_0}$ (x_0 è stato determinato nel passo precedente!).
3. Siccome da (10.4) vale che $\beta_1^{n/p^2} = \alpha^{x_1 n/p}$ allora iniziando da $\gamma = \alpha^{n/p}$ si calcola $\gamma^2, \gamma^3, \ldots$ fino a determinare $i' \le p-1$ tale che $\gamma^{i'} = \beta_1^{n/p^2}$. Quando questo accade, allora $x_1 = i'$, dove $x_1 = \log_{\alpha^{n/p}} \beta^{n/p^2}$.

Si procede iterativamente determinando

$$x_0, \beta_1, x_1, \beta_2, ..., \beta_{k-1}, x_{k-1}$$

applicando in modo alternato (10.4) e (10.5). Alla fine del procedimento si conoscono tutti gli x_i (e quindi x). Quindi

$$c \equiv \sum_{j=0}^{k-1} x_j p^j \ \ mod\ p^k.$$

Applicando lo stesso procedimento a tutti i divisori primi di n si giunge al seguente sistema congruenziale

$$\begin{cases} c \equiv \sum_{j=0}^{k_1-1} x_{j1} p_1^j \ \ mod\ p_1^k \\ \vdots \\ c \equiv \sum_{j=0}^{k_\ell-1} x_{j\ell} p_\ell^j \ \ mod\ p_\ell^k \end{cases}$$

e quindi a $c\ mod\ n$ per il **Teorema Cinese dei Resti**.

Remark 10.13. Il punto chiave su cui si basa l'efficienza dell'algoritmo è che il calcolo degli $x_{ji} = \log_{\alpha^{n/p_i}} \beta^{n/p_i^j}$ dove $0 \le x_{ji} \le p_i - 1$ deve essere piuttosto spedito. Ne segue che p_j non deve essere elevato. Pertanto n è prodotto di primi piccoli, ovvero è **piatto**.

Algoritmo 10.14. **(Pohlig-Hellmann**$(G, n, \alpha, \beta, p, k)$**)**

$j \longleftarrow 0$
$\beta_j \longleftarrow \beta$
while $j \le k-1$
do $\begin{cases} \delta \longleftarrow \beta_j^{n/p^{j+1}} \\ \text{determina } i \text{ tale che } \delta = \alpha^{in/p} \\ x_j \longleftarrow i \\ \beta_{j+1} \longleftarrow \beta_j \alpha^{-x_j p^j} \\ j \longleftarrow j+1 \end{cases}$
return $(x_0, ..., x_{k-1})$

Esempio 10.15. Supponiamo che $p = 29$ e $\alpha = 2$. E' facile vedere che α è un elemento primitivo di $\mathbb{Z}_{29}^*$. Allora $n = p - 1 = 28 = 2^2 \times 7$.

Sia $\beta = 18$ vogliamo determinare $c = \log_2 18$ in $\mathbb{Z}_{29}^*$.

1. Determiniamo $c \ mod\ 4$ e $c \ mod\ 7$ e da qui determiniamo $c \ mod\ 28$.
2. Siccome $c \equiv (x_{01} + 2x_{11}) \ mod\ 4$, da (10.4) segue che $18^{14} \equiv 2^{14x_{01}} \ mod\ 29$ con $0 \leq x_{01} \leq 1$. Si vede facilmente che $x_{01} = 1$.
3. Ora da (10.5) segue che $\beta_1 = 18 \cdot 2^{-1} \equiv 19 \ mod\ 9$.
4. Da (10.4) si ha $19^7 \equiv 2^{x_{11}14} \ mod\ 29$ con $0 \leq x_{11} \leq 1$ che in realtà produce $x_{11} = 1$.

Pertanto $c \equiv 3 \ mod\ 4$.
Siccome $c \equiv x_{02} \ mod\ 7$, da (10.4) segue che $18^4 \equiv 2^{4x_{02}} \ mod\ 29$ con $0 \leq x_{01} \leq 6$. Si vede facilmente che $x_{02} = 4$ e quindi $c \equiv 4 \ mod\ 7$. Allora

$$\begin{cases} c \equiv 3 \ mod\ 4 \\ c \equiv 4 \ mod\ 7 \end{cases}$$

Per il **Teorema Cinese dei Resti** vale che $c \equiv 11 \ mod\ 28$. Quindi, $\log_2 18 = 11$ in $\mathbb{Z}_{29}^*$.

□

Nell'**Algoritmo 10.14** ci sono al più c iterazioni ed in ognuna di esse si calcola $\log_{\alpha^{n/p}} \beta^{n/p^j}$ che ha complessità $O(\sqrt{n} \ln \sqrt{n})$ usando, per esempio, l'algoritmo di Shanks. Pertanto, la complessità dell'**Algoritmo 10.14** è $O(k\sqrt{n} \ln \sqrt{n})$.

10.4 Il Metodo del Calcolo dell'Indice

Gli algoritmi visti precedentemente si utilizzavano per il calcolo del logaritmo discreto in un qualsiasi gruppo finito. L'algoritmo che descriviamo ora, noto come **Metodo del Calcolo dell'Indice**, si utilizza per il calcolo del logaritmo discreto in $\mathbb{Z}_p^*$ con p primo. Il Metodo del Calcolo dell'Indice è più veloce degli altri algoritmi per il calcolo del logaritmo discreto in $\mathbb{Z}_p^*$.

Siano $\alpha, \beta \in \mathbb{Z}_p^*$, con α un elemento primitivo di $\mathbb{Z}_p^*$, e sia $\mathcal{B} = \{p_1, ..., p_B\}$ un insieme di primi "piccoli", allora l'algoritmo consiste in due passi:

1. Calcolo dei logaritmi discreti $\log_\alpha p_1, ..., \log_\alpha p_B$
2. Calcolo dei logaritmo discreto $\log_\alpha \beta$ attraverso $\log_\alpha p_1, ..., \log_\alpha p_B$.

Sia $C > B$, per esempio $C = B + 10$ e si costruiscono C congruenze modulo p, che hanno la seguente forma:

$$\alpha^{x_j} \equiv p_1^{\alpha_{1j}} p_2^{\alpha_{2j}} \cdots p_B^{\alpha_{Bj}} \ mod\ p. \tag{10.6}$$

Per $1 \leq j \leq C$ (per esempio gli x_j possono essere determinati in modo casuale). Siccome $\alpha^{\log_\alpha p_i} = p_i$, allora

$$\alpha^{x_j} \equiv \alpha^{\alpha_{1j}\log_\alpha p_1 + \alpha_{2j}\log_\alpha p_2 + \cdots \alpha_B \log_\alpha p_B} \ mod\ p$$

e quindi (10.6) è equivalente a

$$x_j \equiv (\alpha_{1j}\log_\alpha p_1 + \alpha_{2j}\log_\alpha p_2 + \cdots \alpha_{Bj}\log_\alpha p_B) \ mod\ (p-1)\,.$$

Se sistema congruenziale nelle variabili $\log_\alpha p_1, ..., \log_\alpha p_B$

$$\begin{cases} \alpha_{11}\log_\alpha p_1 + \alpha_{21}\log_\alpha p_2 + \cdots \alpha_{B1}\log_\alpha p_B \equiv x_1 \ mod\ (p-1) \\ \vdots \\ \alpha_{1C}\log_\alpha p_1 + \alpha_{2C}\log_\alpha p_2 + \cdots \alpha_{BC}\log_\alpha p_B \equiv x_C \ mod\ (p-1)\,. \end{cases}$$

ha una soluzione, questa viene memorizzata. Una volta fatto ciò il calcolo di $\log_\alpha \beta$ avviene attraverso un algoritmo randomizzato di tipo Las Végas: si genera un intero casuale s tale che $1 \leq s \leq p-1$ e si calcola $\gamma = \beta\alpha^s \ mod\ p$, lo si cerca di fattorizzare rispetto alla base $\mathcal{B}$, ovvero si cerca di scrivere

$$\beta\alpha^s \equiv p_1^{c_1} p_2^{c_2} \cdots p_B^{c_B} \ mod\ p,$$

o equivalentemente,

$$\log_\alpha \beta \equiv (c_1\log_\alpha p_1 + c_2\log_\alpha p_2 + \cdots c_B \log_\alpha p_B - s) \ mod\ (p-1).$$

Siccome, tutti i termini al secondo membro sono noti, è così determinato $\log_\alpha \beta$.

Esempio 10.16. Sia $p = 10007$ un primo, sia $\alpha = 5$ un elemento primitivo di $\mathbb{Z}^*_{10007}$ e sia $\mathcal{B} = \{2, 3, 5, 7\}$. Supponiamo che **alcuni** degli interi casuali generati siano $x_1 = 4063$, $x_2 = 5136$ e $x_3 = 9865$. Allora

$$\begin{cases} 5^{4063} \ mod\ 10007 = 42 = 2 \times 3 \times 7 \\ 5^{5136} \ mod\ 10007 = 54 = 2 \times 3^3 \\ 5^{9865} \ mod\ 1007 = 189 = 3^3 \times 7 \end{cases}$$

che è equivalente a

$$\begin{cases} \log_5 2 + \log_5 3 + \log_5 7 = 4063 \ mod\ 10006 \\ \log_5 2 + 3\log_5 3 = 5136 \ mod\ 10006 \\ 3\log_5 3 + \log_5 7 = 9865 \ mod\ 10006 \end{cases}$$

da cui si ricava

$$\log_5 2 = 6578 \quad \log_5 3 = 6190 \quad \log_5 7 = 1301.$$

Supponiamo di voler determinare $\log_5 9451$. Si sceglie un esponente "casuale" $s = 7736$ e si calcola

$$\gamma = \left(9451 \times 5^{7736}\right) \ mod\ 10007 = 8400 = 2^4 3^1 5^2 7^1,$$

quindi

$$\begin{aligned} \log_5 9451 &\equiv \left(4\log_5 2 + \log_5 3 + 2\log_5 5 + \log_5 7\right) \ mod\ 10006 \\ &\equiv \left(4 \times 6578 + 6190 + 2 \times 1 + 1301 - 7736\right) \ mod\ 10006 \\ &\equiv 6057. \end{aligned}$$

È facile verificare che effettivamente $5^{6057} \equiv 9451 \ mod\ 10007$.

□