# Una strategia multistep per risolvere sistemi di equazioni polinomiali su campi finiti ed un nuovo attacco al cifrario Trivium

R. La Scala*, F. Pintore*, S.K. Tiwari**, A. Visconti†

*Università di Bari, **Technology Innovation Institute (UAE), †Università di Milano.

CrypTO Conference, 26 Maggio 2023

- Molti problemi di natura scientifica, tecnologica e industriale corrispondono a risolvere un sistema di equazioni polinomiali a più variabili su un campo finito $\mathbb{F}$. Tale problema generale si indica con il termine **MP-problem** oppure **MQ-problem** nel caso di equazioni quadratiche.
- Se consideriamo il campo binario $\mathbb{F} = \{0, 1\}$ allora questo problema è equivalente al **SAT-problem** (Soddisfacibilità Booleana) le cui innumerevoli applicazioni spaziano dalla verifica dell'hardware e del software mediante Symbolic Model Checking al cosiddetto Satplan (Planning as Satisfiability) utilizzato anche in AI.
- Altre applicazioni elettive del MP-problem si trovano in Crittografia e nella Teoria dei Codici. Per quest'ultima tematica ricordiamo, ad esempio, il calcolo degli Error Locator Polynomials per i codici ciclici.

- In Crittografia ricordiamo che lo stesso Shannon affermò che un crittosistema si può ritenere **sicuro**

  *"if we could show that solving a certain (crypto)system requires at least as much work as solving a system of simultaneous equations in a large number of unknowns, of a complex type".*

- In termini moderni, la sicurezza computazionale risiede nella complessità di un **attacco algebrico**. Un attacco algebrico è una istanza di un MP-problem dove le variabili da risolvere sono essenzialmente i bit della chiave e le equazioni si ottengono da plaintexts e ciphertexts noti all'attaccante (KPA, CPA o simili).

- In aggiunta ai cifrari a chiave privata (a blocchi ed a flusso), anche i crittosistemi a chiave pubblica basati su fattorizzazione e logaritmo discreto ammettono attacchi algebrici.
- Nella moderna Crittografia Post-Quantum molte primitive, specialmente le firme digitali, sono definite direttamente in termini di sistemi e mappe polinomiali a più variabili con coefficienti e soluzioni su un campo finito (Multivariate Cryptography).
- Una stima della complessità necessaria a risolvere i particolari sistemi di equazioni associati alla crittanalisi di questi cifrari è dunque di **vitale importanza per la loro validazione**. In questo tema si inserisce quindi il nostro contributo.

- Un MP-problem consiste nel cercare le soluzioni $(a_1, \ldots, a_n) \in \mathbb{F}^n$ di un sistema di equazioni

$$\begin{cases} f_1(x_1, \ldots, x_n) &= 0 \\ & \vdots \\ f_m(x_1, \ldots, x_n) &= 0 \end{cases} \tag{1}$$

  dove $f_i(x_1, \ldots, x_n)$ sono polinomi a coefficienti nel campo finito $\mathbb{F} = \mathrm{GF}(q)$.
- La complessità del problema è maggiorata da quella del brute force $\mathbf{q^n}$ che cresce esponenzialmente col numero di variabili.
- I migliori algoritmi di risoluzione a disposizione (SAT solvers, basi di Gröbner, XL solvers, etc) non cambiano essenzialmente questa complessità nel caso peggiore.

- As esempio, per le basi di Gröbner (algebra lineare sulle matrici di Macaulay) troviamo una complessità esponenziale del tipo ($k = n^2 q, 2 < \omega \leq 3$)

$$\mathcal{O}(k^{k\omega})$$

- Con tali complessità, risolvere un sistema di reale interesse con un numero elevato di variabili è generalmente impossibile con una singola istanza di un qualsiasi solver.
- Una tecnica ampiamente in uso consiste quindi nel fare brute force su un sottinsieme di variabili $\{x_1, \ldots, x_k\}$ e nel risolvere i $q^k$ sistemi corrispondenti a ciascuna valutazione:

$$\begin{cases} f_1(a_1, \ldots, a_k, x_{k+1}, \ldots, x_n) = 0 \\ \vdots \\ f_m(a_1, \ldots, a_k, x_{k+1}, \ldots, x_n) = 0 \end{cases} \tag{2}$$

per ogni $(a_1, \ldots, a_k) \in \mathbb{F}^k$.

- Tale approccio si chiama una **guess-and-determine** oppure **hybrid strategy**. Un buona scelta dell'insieme di variabili $\{x_1, \ldots, x_k\}$ garantisce generalmente che la complessità totale è minore di quella stimata per il sistema polinomiale iniziale (1)
- Per un MP-problem come quelli ottenuti da un attacco algebrico ad un crittosistema reale, se assumiamo che ciascun sistema (2) si possa risolvere in un tempo ragionevole, il numero $k$ di variabili da valutare esaustivamente e la corrispondente complessità $q^k$ risulta generalmente ancora troppo alta.
- Nel nostro lavoro si propone quindi una **multistep strategy** che prevede che il numero $k$ sia inizialmente moderato e che questo sia esteso progressivamente ogni qualvolta che una valutazione $(a_1, \ldots, a_k)$ sia insufficiente a garantire che il corrispondente sistema (2) si possa risolvere in tempo ragionevole.

- Resta determinata quindi una sequenza di passi $1 \leq k_1 \leq \ldots \leq k_r \leq n$ dove l'insieme massimo di variabili $\{x_1, \ldots, x_{k_r}\}$ deve garantire che **tutte** le sue valutazioni producano sistemi risolvibili in tempo ragionevole.
- La classica strategia guess-and-determine corrisponde quindi ad avere un unico passo $k_1 = k_r$.
- Nel paper dimostriamo che questa scelta produce la complessità peggiore e, per contro, la complessità ottimale si ottiene col massimo numero di passi ovvero per $k_{i+1} = k_i + 1$. Chiamiamo questa una **full multistep strategy**.
- Per complessità intendiamo qui il numero di sistemi del tipo (2), risolvibili in tempo ragionevole, che sono equivalenti al sistema iniziale (1).

- Per decidere se una valutazione $(a_1, \ldots, a_{k_i})$ produce un sistema (2) risolvibile in tempo ragionevole utilizziamo un preprocesso che chiamiamo GBELIMLIN.
- Dopo la valutazione $x_1 = a_1, \ldots, x_{k_i} = a_{k_i}$, questo preprocesso consiste nel calcolare una base di Gröbner incompleta cioè troncata ad un certo grado prefissato $D$, scelto generalmente basso per avere GBELIMLIN efficiente.
- Se il sistema possiede al più una soluzione, come generalmente accade nelle applicazioni crittografiche, il nuovo sistema di equazioni ottenuto conterrà equazioni lineari che vengono utilizzate per eliminare ulteriori variabili. Dopo tale eliminazione indichiamo con $\mathrm{NRV}$ il numero di variabili che si trovano nel sistema risultante.

- Fissato opportunamente un altro paramentro $B$, decideremo che il sistema ottenuto dalla valutazione $(a_1 \ldots, a_{k_i})$ mediante GBELIMLIN è risolubile in tempo ragionevole se $\mathrm{NRV} \leq B$. In questo caso, diremo che $(a_1 \ldots, a_{k_i})$ è un **tamed guess**.
- Altrimenti, estenderemo questa valutazione a tutte le valutazioni

$$(a_1, \ldots, a_{k_i}, a_{k_i+1}, \ldots, a_{k_{i+1}})$$

per ogni $(a_{k_i+1}, \ldots, a_{k_{i+1}}) \in \mathbb{F}^{k_{i+1}-k_i}$ e diremo che $(a_1 \ldots, a_{k_i})$ è un **wild guess**.

- Il metodo è iterato per tutti i passi $1 \leq k_1 < \ldots < k_r \leq n$. Il sistema corrispondente ad ogni tamed guess viene risolto da un qualsiasi solver, ad esempio una base di Gröbner completa oppure un SAT solver. Chiamiamo questo algoritmo MULTISOLVE.

- Il numero totale di sistemi da risolvere è quindi il numero di tamed guess che si ottiene mediante la **semplice formula**

$$C = (1 - p_B(k_1))q^{k_1} + (p_B(k_1) - p_B(k_2))q^{k_2} + \ldots$$
$$+ (p_B(k_{r-2}) - p_B(k_{r-1}))q^{k_{r-1}} + p_B(k_{r-1})q^{k_r}.$$

  dove $p_B(k_i)$ è la probabilità di avere un wild guess nello spazio $\mathbb{F}^{k_i}$.
- Qualunque siano queste probabilità, nel paper dimostriamo che $p_B(k_1) \geq \ldots \geq p_B(k_r)$ dove, per definizione di ultimo passo, deve essere $p_B(k_r) = 0$.
- La probabilità $p_B(k_i)$ può essere facilmente stimata su un opportuno testset $T \subset \mathbb{F}^{k_i}$ mediante la sola procedura GBElimLin. Questi dati statistici appaiono sufficientemente stabili per stimare $C$ con buona approssimazione.

- Si dimostra pure che il valore minimo della complessità $C$ si ottiene per una full multistep strategy, precisamente

$$C = \sum_{k' \leq k \leq k''} (p_B(k-1) - p_B(k))q^k,$$

  dove $p_B(k'') = 0$ e per convenzione $p_B(k'-1) = 1$.
- Osserviamo che la complessità di MULTISOLVE può essere stimata senza applicare l'algoritmo e quindi può essere ottimizzata variando i parametri $D, B$, come pure l'insieme di variabili $\{x_1, \ldots, x_{k''}\}$, che determinano le probabilità $p_B(k)$.
- Se stiamo facendo crittanalisi, il valore ottimale di $C$ fornirà quindi una buona stima della sicurezza del crittosistema.

- Un caso di studio analizzato nel paper è il **cifrario a flusso** TRIVIUM sviluppato in Europa nell'ambito del progetto eSTREAM.
- Questo cifrario, presentato nel 2005, non ha ricevuto nessun attacco che abbia messo in discussione la sua sicurezza nonostante la sua semplice (ed elegante) struttura.
- Il cifrario si basa su tre **FSR non-lineari**, diciamo $x, y, z$, che vengono aggiornati mediante il seguente sistema di equazioni (esplicite alle differenze) quadratiche

$$\begin{cases} x(93+t) = z(45+t) + x(24+t) + z(t) + z(1+t)z(2+t) \\ y(84+t) = x(27+t) + y(6+t) + x(t) + x(1+t)x(2+t) \\ z(111+t) = z(24+t) + y(15+t) + y(t) + y(1+t)y(2+t) \end{cases}$$

  per tutti i clock $t \in \mathbb{N}$.
- Il registro cumulativo di TRIVIUM ha dunque lunghezza $288 = 93 + 84 + 111$ bit.

- Il keystream è ottenuto dal valore del registro al clock $t$ (stato) mediante il polinomio lineare

$$f = x(27 + t) + x(t) + y(15 + t) + y(t) + z(45 + t) + z(t)$$

- Il keystream va in output dopo una fase di warm-up e precisamente al clock $T = 4 \cdot 288 = 1152$. Questo protegge lo stato iniziale ($t = 0$) che contiene la chiave ed un IV.
- Chiave ed IV hanno ciascuno lunghezza 80 bit. I restanti bit dello stato iniziale sono costanti.

- Utilizzando un approccio simbolico (basi di Gröbner) si può provare che la funzione di transizione di stato di TRIVIUM è invertibile e calcolarne l'inversa.
- Questo implica che attaccare un qualsiasi stato, ad esempio quello dove origina il keystream, è equivalente ad attaccare lo stato iniziale contenente la chiave e lo IV. Naturalmente questo significa dover risolvere tutti i 288 bit di questo stato interno col vantaggio però di risolvere un sistema **molto più semplice**.
- Utilizzando 240 bit del keystream (attacco KPA), generiamo quindi 240 equazioni sulle 288 variabili

$$x(0), \ldots, x(92), y(0), \ldots, y(83), z(0), \ldots, z(111).$$

- Di queste equazioni, solo 66 sono lineari e permettono l'eliminazione immediata di altrettante variabili il cui numero scende quindi a 222.

- A questo punto, applichiamo MULTISOLVE per un numero $k$ di variabili su cui fare brute force nell'intervallo $106 \leq k \leq 116$.
- I tamed guess corrispondenti hanno $\mathrm{NRV} \leq B$ dove $B$ varia nell'intervallo $32 \leq B \leq 38$.
- Il grado $D$ utilizzato per GBELIMLIN è sempre fissato al max grado delle equazioni in input.
- Queste scelte sono determinate da alcune limitazioni nelle risorse di calcolo e nel software utilizzato per il calcolo delle basi di Gröbner.
- Per stimare la complessità di MULTISOLVE è sufficiente stimare le probabilità $p_B(k)$ sopra testset costituiti da molti guess random di $k$ variabili per sistemi ottenuti da differenti keystream corrispondenti a chiavi random.

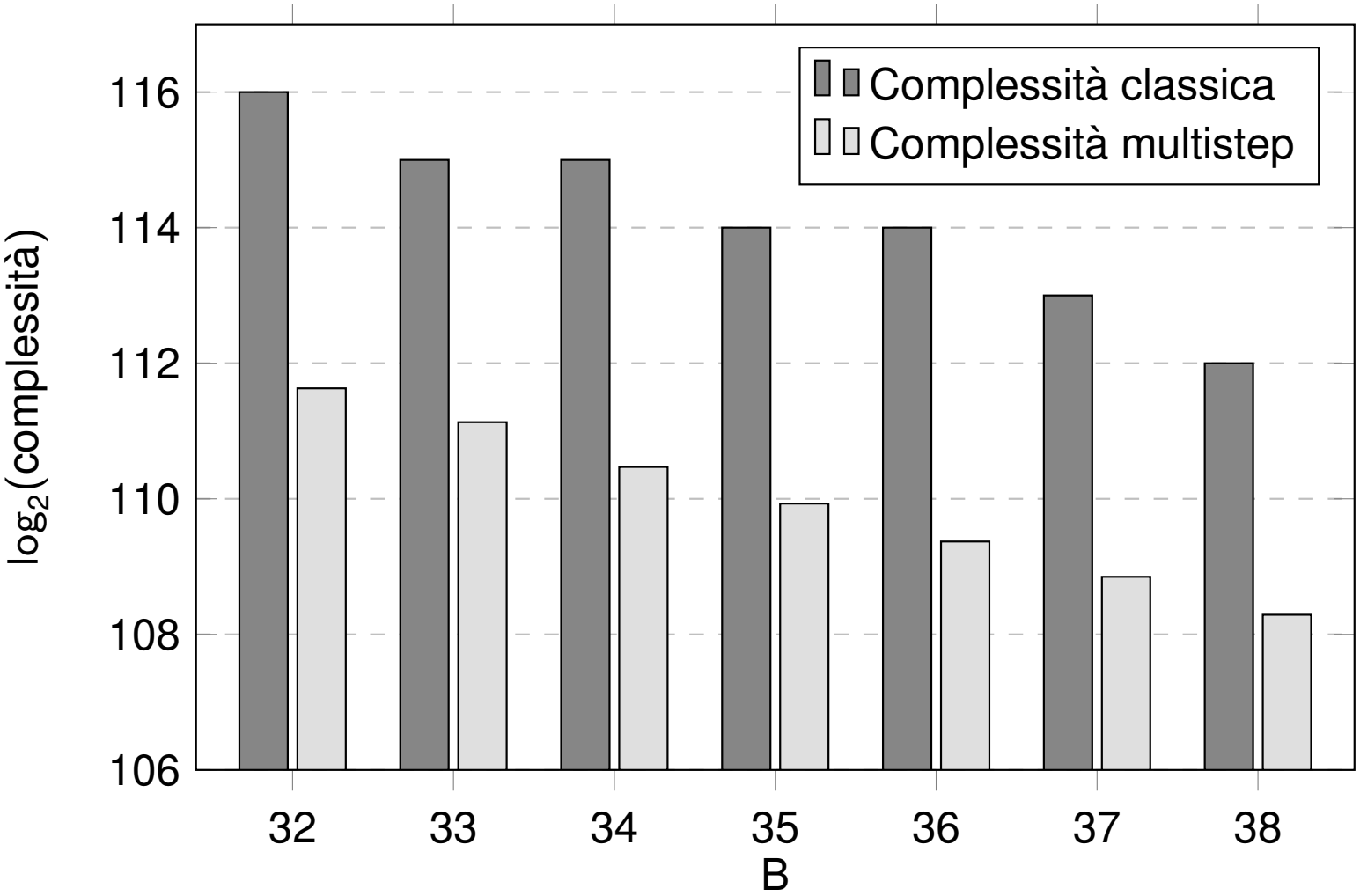
Probabilità $p_B(k)$ stimate su testset di guess random

| $k/B$ | 32 | 33 | 34 | 35 | 36 | 37 | 38 |
|---|---|---|---|---|---|---|---|
| 106 | 0.63153 | 0.61703 | 0.60867 | 0.58736 | 0.55925 | 0.50848 | 0.44923 |
| 107 | 0.61670 | 0.60675 | 0.58188 | 0.55359 | 0.50471 | 0.44638 | 0.38121 |
| 108 | 0.57581 | 0.54169 | 0.49049 | 0.43468 | 0.37077 | 0.31258 | 0.23341 |
| 109 | 0.49127 | 0.43814 | 0.38190 | 0.29349 | 0.23219 | 0.16003 | 0.10303 |
| 110 | 0.43263 | 0.37582 | 0.28784 | 0.22722 | 0.15845 | 0.10179 | 0.04910 |
| 111 | 0.28415 | 0.22218 | 0.14967 | 0.09698 | 0.04762 | 0.02165 | 0.00670 |
| 112 | 0.14362 | 0.08954 | 0.04715 | 0.01627 | 0.00650 | 0.00053 | 0 |
| 113 | 0.08838 | 0.04610 | 0.01549 | 0.00650 | 0.00053 | 0 | 0 |
| 114 | 0.01498 | 0.00725 | 0.00053 | 0 | 0 | 0 | 0 |
| 115 | 0.00043 | 0 | 0 | 0 | 0 | 0 | 0 |
| 116 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

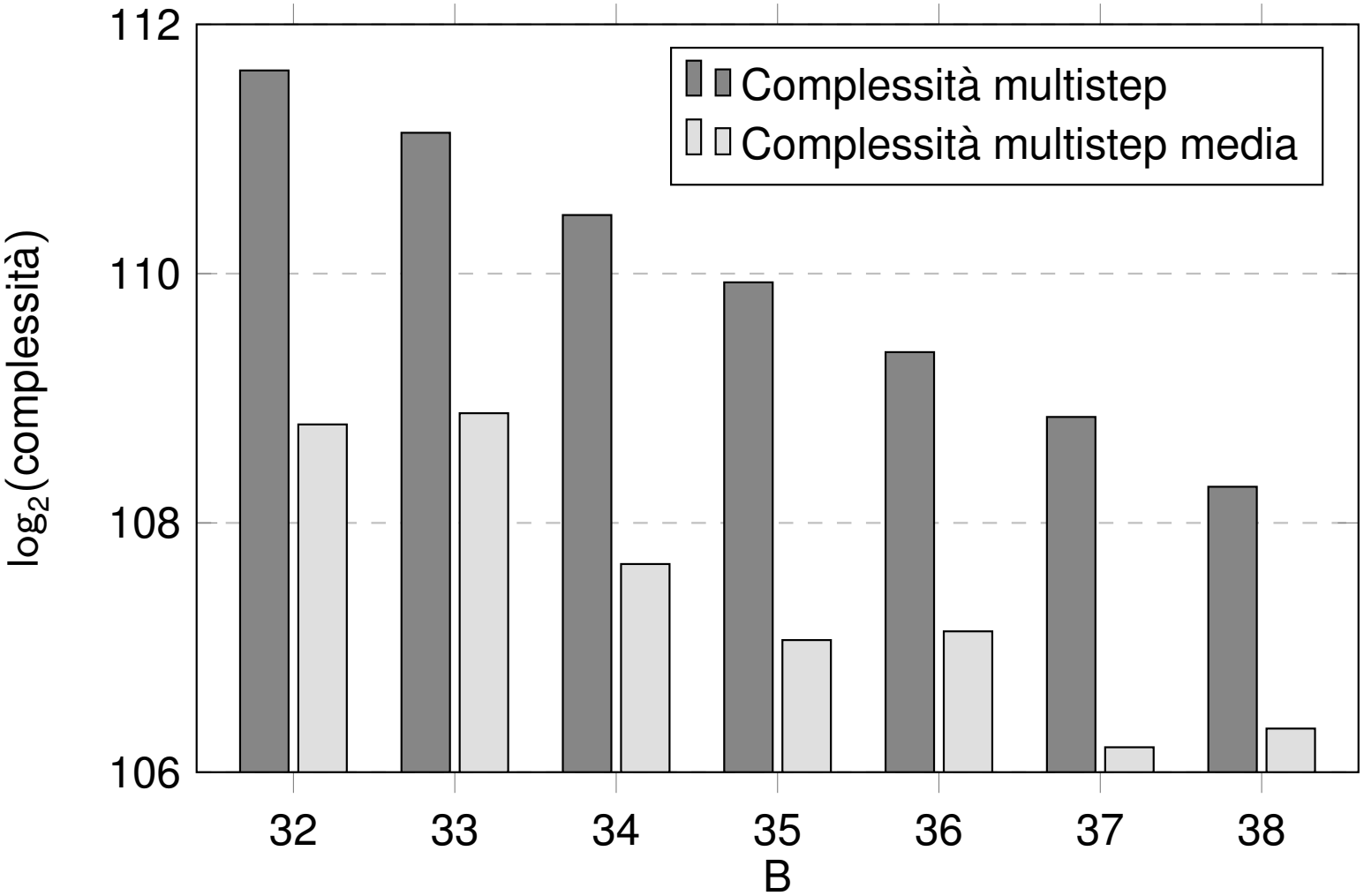
- Questa **semplice tabella** determina quali sono i passi finali $k''$ ($p_B(k'') = 0$) dell'algoritmo MULTISOLVE e la sua complessità per i diversi valori del parametro $B$.
- Se ne ottiene un confronto diretto fra la complessità di questo algoritmo multistep (con passo iniziale $k'$ fissato a 106) e quella della classica strategia guess-and-determine che corrisponde al solo passo finale $k''$.
- Ricordiamo che la formula della complessità (numero di tamed guess) per MULTISOLVE in full multistep strategy è

$$C = \sum_{k' \leq k \leq k''} (p_B(k-1) - p_B(k))2^k$$

mentre per l'algoritmo classico si ha $C = 2^{k''}$.


Complessità classica
Complessità multistep
$\log_2$(complessità)
116
114
112
110
108
106
32
33
34
35
36
37
38
B

- Di fatto questa analisi della complessità di MULTISOLVE corrisponde al **caso peggiore** in cui il guess corretto si ottenga come tamed guess solo all'ultimo passo $k''$.
- **Mediamente**, il guess corretto si ottiene invece in un passo $\bar{k}''$ molto precedente all'ultimo arrestando a questo passo l'algoritmo.
- Questo valore $\bar{k}''$ può essere stimato semplicemente calcolando le probabilità $p_B(k)$ per un testset di soli guess corretti per molti keystream corrispondenti a chiavi random.
- Mediamente significa che $p_B(\bar{k}'') \leq 1/2$ per un testset di guess corretti.


112
110
108
106
$\log_2$(complessità)
Complessità multistep
Complessità multistep media
32
33
34
35
36
37
38
B

- Il valore logaritmico migliore per la complessità media di questo attacco multistep a TRIVIUM è 106.2 bit ($B = 37$) che è estremamente vicino al passo iniziale scelto $k' = 106$.

- Lo sviluppo di metodi automatici per l'ottimizzazione dei parametri di MULTISOLVE potrebbe fornire stime ancora più accurate della complessità di specifici MP-problem.

## Referenze


- La Scala, R.; Pintore, F.; Tiwari S.K.; Visconti A., A multistep strategy for polynomial system solving over finite fields and a new algebraic attack on the stream cipher Trivium, Cryptology ePrint Archive, 2023/542, 27 pp.
- La Scala, R.; Polese S.; Tiwari S.K.; Visconti A., An algebraic attack to the Bluetooth stream cipher E0. Finite Fields Appl. 84 (2022), Paper No. 102102, 29 pp.
- La Scala, R.; Tiwari S.K., Stream/block ciphers, difference equations and algebraic attacks. J. Symbolic Comput. 109 (2022), 177–198.