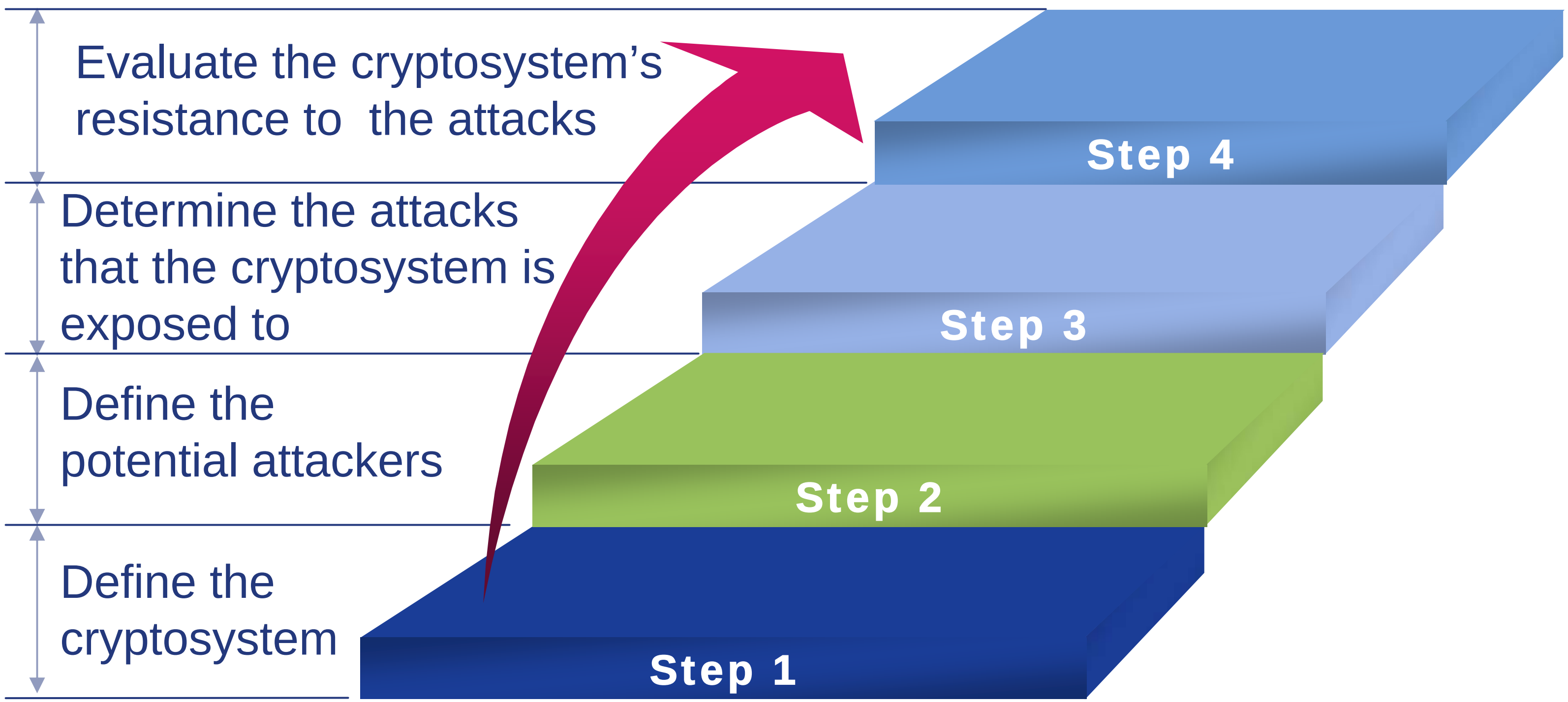
Formal methods and tools for evaluating cryptographic systems security

Alexandra A. Savelieva
Supervisor: Prof. Sergey M. Avdoshin

State University – Higher School of Economics, Russia
Software Engineering Department

Cryptosystem security assessment



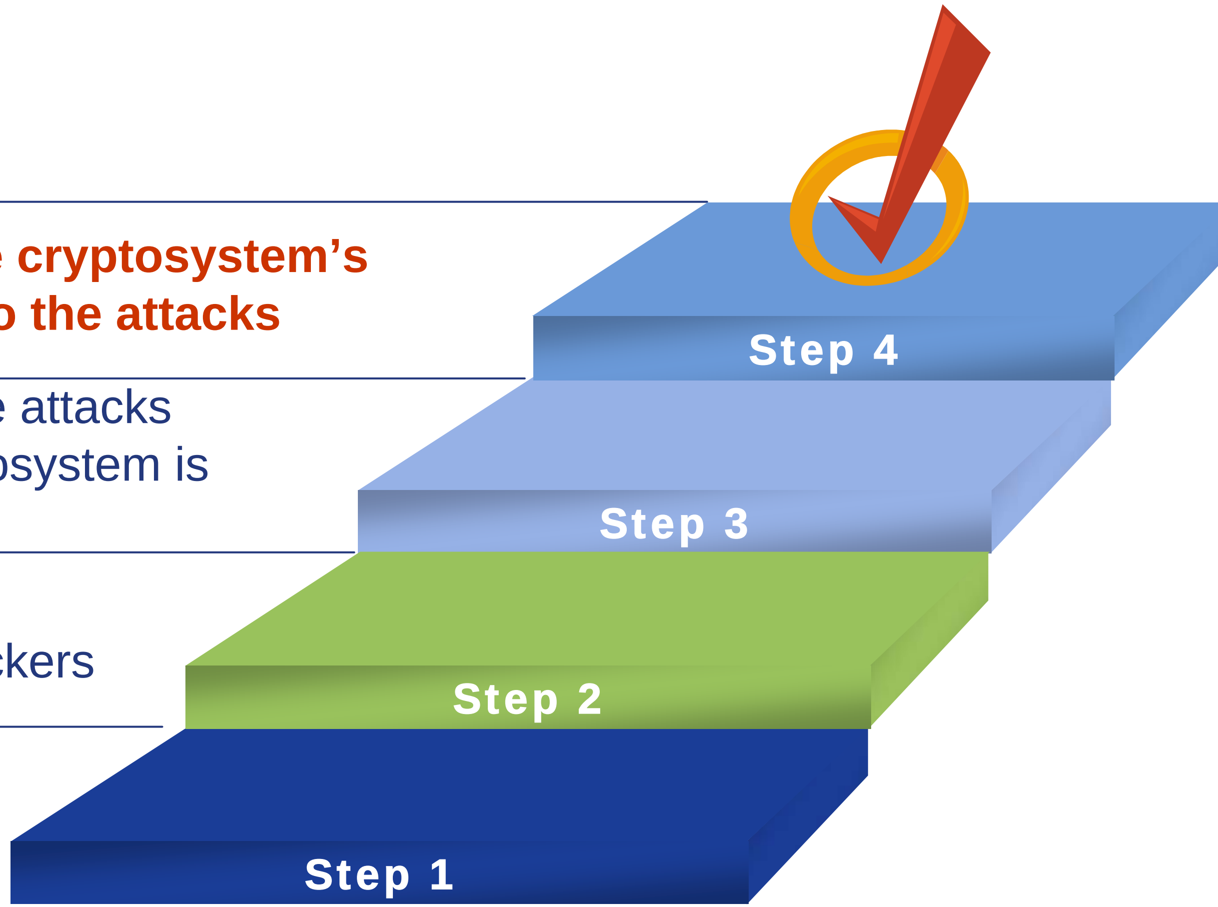
Cryptosystem security assessment

Evaluate the cryptosystem's resistance to the attacks

Determine the attacks that the cryptosystem is exposed to

Define the potential attackers

Define the cryptosystem



Optimizing tools in cryptanalysis

- **Public key cryptography: based on the complexity of**
  - **Factorization**
  - **Discrete logarithm computation**
- **Not NP-complete problems, but no polynomial algorithms is known to solve these problems**
- **The best known algorithms run in subexponential time of the form:**

$$L_x[\gamma; c] = e^{(c+o(1))(\log x)^{\gamma}(\log\log x)^{1-\gamma}}, \text{ where } x \to \infty, 0 < \gamma < 1, c = const, c > 0$$

Smoothness

- For $u, B \in \mathbb{Z}$, we say u is B –smooth, if all the primes q_i in the prime factorization of u are $\leq B$

$$u = \prod_{i=1}^{k} q_i^{\alpha_i}$$

- Primes q_i form Factor Base Q:

$$Q = \left\{ q \leq B = e^{const\sqrt{\log x \log\log x}} \right\}$$

Index-calculus methods

Relation Collection Step → **Matrix Step** → **Final Step**

Relation Collection Step

Find many linear relations involving the unknown logarithms of the primes in the factor base

Matrix Step

Solve the linear system for the unknown logarithms using techniques from linear algebra

Final Step

Compute individual logarithm using the logarithms of the primes in the factor base

Problem: $a^x = b \pmod{p}$

Generating relations

- **Pick a random integer** $v \in [1..p\text{-}1]$ **and compute**

$$c \equiv a^v \pmod{p}$$

- c **is an element of** Z_p^* **but we treat it as an integer and factorize it over** Q

- **Given that** $c = q_1^{b1} \cdot q_2^{b2} \cdot \ldots \cdot q_n^{bn}$ **where all** $q_i \in Q$**:**

$$v \equiv b_1 \log_a q_1 + b_2 \log_a q_2 + \ldots b_n \log_a q_n \pmod{p - 1}$$

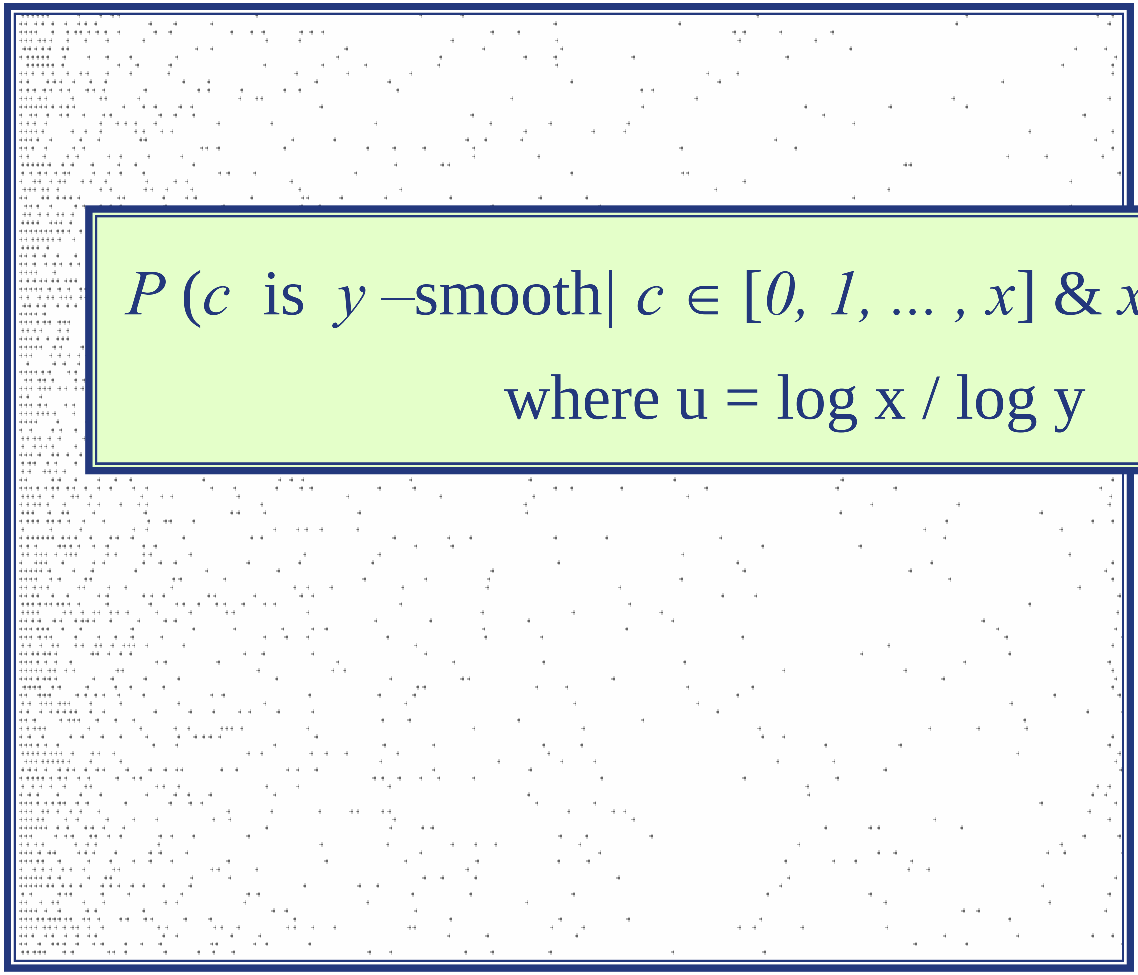
Problem: $a^x = b \pmod{p}$

Observations

- **Interesting properties of matrices:**
  - **Size**: 100 000 x 100 000 or larger
  - **Elements**: very small (the number of elements $\sim p \to 0$)
  - **Density**: Sparse, but not uniformly sparse!

Problem: $a^x = b (\mathrm{mod}\ p)$

Matrix structure



$$P\,(c \text{ is } y\text{–smooth} \mid c \in [0, 1, \ldots, x]\ \&\ x \le y) = O(u^{-u}),$$

$$\text{where } u = \log x / \log y$$

Observations

- **Interesting properties of matrices:**
  - **Size**: 100 000 x 100 000 or larger
  - **Density**: Sparse, but not uniformly sparse!
  - **Elements**: very small (the number of elements $\sim p \to 0$)

- **Linear algebra needs to be done over the ring $\mathbb{Z}/n\mathbb{Z}$ for some composite n**

Problem: $a^x = b \pmod p$

Solving a linear system in Z/36

$$26\cdot 18\equiv 0 \pmod{36}$$

$$3\cdot 12\equiv 0 \pmod{36}$$

$$\begin{cases} 26x+3y=4 \\ 9x+34y=1 \end{cases}$$

$$9\cdot 4\equiv 0 \pmod{36}$$

$$34\cdot 18\equiv 0 \pmod{36}$$

All the coefficients are non-invertible; the solution however exists and is unique modulo 36

$$\begin{cases} x=17 \\ y=22 \end{cases}$$

Linear algebra techniques

Reducing the problem to:

i. **solving a number of systems over prime fields and combining the results using the Chinese Remainder Theorem**
ii. **solving a system of Diophantine equations**
iii. **solving an equation over a matrix ring**

Linear algebra techniques

Reducing the problem to:

i. **solving a number of systems over prime fields and combining the results using the Chinese Remainder Theorem**
ii. **solving a system of Diophantine equations**
iii. **solving an equation over a matrix ring**

Solving a number of systems over prime fields

$$\begin{cases} 26x + 3y = 4 \\ 9x + 34y = 1 \end{cases} \pmod{36}$$

$$\boxed{36 = 2^2 \cdot 3^2}$$

$$\begin{cases} 26x + 3y = 4 \\ 9x + 34y = 1 \end{cases} \pmod{2^2} \qquad \begin{cases} 26x + 3y = 4 \\ 9x + 34y = 1 \end{cases} \pmod{3^2}$$

$$\begin{cases} 2x + 3y = 0 \\ x + 2y = 1 \end{cases} \pmod{4} \qquad \begin{cases} 8x + 3y = 4 \\ 7y = 1 \end{cases} \pmod{9}$$

Solving a number of systems over prime fields

$$\begin{cases} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \end{pmatrix} \pmod 4 \\ \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 8 \\ 4 \end{pmatrix} \pmod 9 \end{cases}$$

Chinese Remainder Theorem yields the result:

$$\begin{cases} x = 17 \\ y = 22 \end{cases} \pmod{36}$$

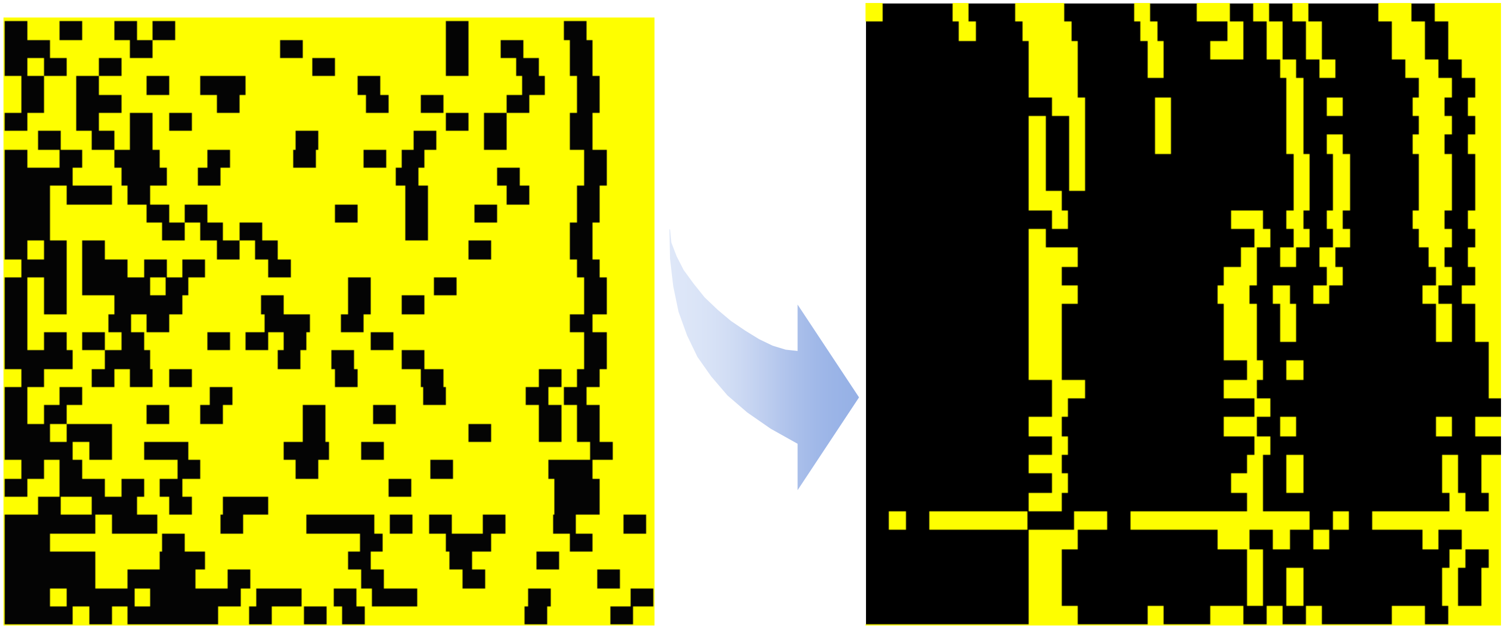
Gaussian-Jordan elimination

$$\left(\begin{array}{cccccc|c} a_{11} & \cdots & \cdots & a_{1n} & a_{1,n+1} \cdots & a_{1,m} & b_1 \\ \vdots & \ddots & & \vdots & \vdots & \vdots & \vdots \\ \vdots & & \ddots & \vdots & \vdots & \vdots & \vdots \\ a_{n1} & \cdots & \cdots & a_{nn} & a_{n,n+1} \cdots & a_{n,m} & b_n \end{array}\right)$$

$$\downarrow$$

$$\left(\begin{array}{ccccccc|c} 1 & 0 & \cdots & 0 & a'_{1,i_{n+1}} & \cdots & a'_{1,i_m} & b'_1 \\ 0 & \ddots & & \vdots & \vdots & & \vdots & \vdots \\ \vdots & & \ddots & 0 & \vdots & & \vdots & \vdots \\ 0 & \cdots & 0 & 1 & a'_{n,i_{n+1}} & \cdots & a'_{n,i_m} & b'_n \end{array}\right)$$

Matrix structure: Gaussian-Jordan elimination



Step 1 Step 5

Disadvantages

- Factorization of $p - 1$ is unknown
- We have to solve many systems instead of one
- The coefficients in the dense part of the matrix grow rapidly
- Fill-in quickly causes matrix to become non-sparse

Linear algebra techniques

Reducing the problem to:

i. **solving a number of systems over prime fields and combining the results using the Chinese Remainder Theorem**
ii. **solving a system of Diophantine equations**
iii. **solving an equation over a matrix ring**

Solving a system of Diophantine equations

- Reduction:

$$\begin{cases} 26x + 3y + 36v_1 = 4 \\ 9x + 34y + 36v_2 = 1 \end{cases}$$

- General solution:

$$\begin{cases} x = 5653025 + t_0 \cdot 1224 + t_1 \cdot (-21492) \\ y = -1496390 + t_0 \cdot (-324) + t_1 \cdot 5688 \\ v_1 = -3958042 + t_0 \cdot (-857) + t_1 \cdot 15048 \\ v_2 = 0 + t_0 \cdot 0 + t_1 \cdot 1 \end{cases}, \quad t_0, t_1 \in \mathbb{Z}$$

exponential growth of coefficients

Linear algebra techniques

Reducing the problem to:

i. **solving a number of systems over prime fields and combining the results using the Chinese Remainder Theorem**
ii. **solving a system of Diophantine equations**
iii. **solving an equation over a matrix ring**

Solving an equation over a matrix ring

$Ax=b$ $\rightarrow$ $x=A^{-1}b$

Glukhov M.M., Elizarov V.P., Nechaev A.A. Algebra. Vol. I - M.: Gelios ARV, 2003

Elizarov V.P. Russian Mathematical Surveys. – 1993. Vol. 48, No. 2. pp. 181-182.

Efficient algorithm for matrix inverting

?

Generalized Gaussian-Jordan Elimination (GGJE) for residue rings

- The basic idea:
  - Extended Euclidean algorithm
  - Gaussian-Jordan elimination
- Applicable to:
  - Residue rings
  - Finite fields
- Efficiency:
  - At worst case time and space complexity equivalent to Gaussian-Jordan elimination for finite fields

Extended Euclidian Algorithm

- INPUT: $a, b \in \mathbb{Z}_+$
- OUTPUT: d, x, y, r, s:

$$\begin{cases} d = GCD(a,b) = a \cdot x + b \cdot y \\ 0 = a \cdot r + b \cdot s \end{cases}$$

SUB $Euclid(a,b)$

$$\begin{pmatrix} d & x & y \\ n & r & s \end{pmatrix} \leftarrow \begin{pmatrix} a & 1 & 0 \\ b & 0 & 1 \end{pmatrix}$$

WHILE $n \geq 0$ LOOP

$$c \leftarrow \lfloor d / n \rfloor$$

$$\begin{pmatrix} d & x & y \\ n & r & s \end{pmatrix} \leftarrow \begin{pmatrix} 0 & 1 \\ 1 & -c \end{pmatrix} \times \begin{pmatrix} d & x & y \\ n & r & s \end{pmatrix}$$

END WHILE

END SUB

Euclidian Algorithm applied to matrix

$$\begin{cases} 26x + 3y = 4 \\ 9x + 34y = 1 \end{cases} \pmod{36}$$

$$\begin{array}{c} [1] \\ [2] \end{array} \left(\begin{array}{cc|c} 26 & 3 & 4 \\ 9 & 34 & 1 \end{array}\right) \xrightarrow{[1]-[2]\cdot 2} \left(\begin{array}{cc|c} 8 & 7 & 2 \\ 9 & 34 & 1 \end{array}\right) \xrightarrow{[1]\leftrightarrow[2]} \left(\begin{array}{cc|c} 9 & 34 & 1 \\ 8 & 7 & 2 \end{array}\right)$$

$$\begin{array}{c} [1] \\ [2] \end{array} \left(\begin{array}{cc|c} 9 & 34 & 1 \\ 8 & 7 & 2 \end{array}\right) \xrightarrow{[1]-[2]\cdot 1} \left(\begin{array}{cc|c} 1 & 27 & 35 \\ 8 & 7 & 2 \end{array}\right) \xrightarrow{[1]\leftrightarrow[2]} \left(\begin{array}{cc|c} 8 & 7 & 2 \\ 1 & 27 & 35 \end{array}\right)$$

$$\begin{array}{c} [1] \\ [2] \end{array} \left(\begin{array}{cc|c} 8 & 7 & 2 \\ 1 & 27 & 35 \end{array}\right) \xrightarrow{[1]-[2]\cdot 8} \left(\begin{array}{cc|c} 0 & 7 & 10 \\ 1 & 27 & 35 \end{array}\right) \xrightarrow{[1]\leftrightarrow[2]} \left(\begin{array}{cc|c} 1 & 27 & 35 \\ 0 & 7 & 10 \end{array}\right)$$

$$\begin{array}{c} [1] \\ [2] \end{array} \left(\begin{array}{cc|c} 1 & 27 & 35 \\ 0 & 7 & 10 \end{array}\right) \xrightarrow{[2]\cdot 31} \left(\begin{array}{cc|c} 1 & 27 & 35 \\ 0 & 1 & 22 \end{array}\right) \xrightarrow{[1]-[2]\cdot 27} \left(\begin{array}{cc|c} 1 & 0 & 17 \\ 0 & 1 & 22 \end{array}\right)$$

Euclidian Algorithm applied to matrix

Bezout coefficients for $a=26, b=9$:

$$1 = 26\cdot(35) + 9\cdot(3) \qquad 0 = 26\cdot(9) + 9\cdot(10)$$

$$\begin{array}{c}[1]\\ [2]\end{array} \boxed{\left(\begin{array}{cc|c} 26 & 3 & 4 \\ 9 & 34 & 1 \end{array}\right)} \xrightarrow{[1]-[2]\cdot 2} \left(\begin{array}{cc|c} 8 & 7 & 2 \\ 9 & 34 & 1 \end{array}\right) \xrightarrow{[1]\leftrightarrow[2]} \left(\begin{array}{cc|c} 9 & 34 & 1 \\ 8 & 7 & 2 \end{array}\right)$$

$$\begin{array}{c}[1]\\ [2]\end{array} \left(\begin{array}{cc|c} 9 & 34 & 1 \\ 8 & 7 & 2 \end{array}\right) \xrightarrow{[1]-[2]\cdot} \quad \xrightarrow{[1]\leftrightarrow[2]} \left(\begin{array}{cc|c} 8 & 7 & 2 \\ 1 & 27 & 35 \end{array}\right)$$

$$\boxed{\begin{array}{l} [1]' = [1]\cdot(35) + [2]\cdot(3) \\ [2]' = [1]\cdot(9) + [2]\cdot(10) \end{array}}$$

$$\begin{array}{c}[1]\\ [2]\end{array} \left(\begin{array}{cc|c} 8 & 7 & 2 \\ 1 & 27 & 35 \end{array}\right) \xrightarrow{[1]-[2]\cdot 8} \left(\begin{array}{cc|c} 0 & 7 & 10 \\ 1 & 27 & 35 \end{array}\right) \xrightarrow{[1]\leftrightarrow[2]} \boxed{\left(\begin{array}{cc|c} 1 & 27 & 35 \\ 0 & 7 & 10 \end{array}\right)}$$

$$\begin{array}{c}[1]\\ [2]\end{array} \left(\begin{array}{cc|c} 1 & 27 & 35 \\ 0 & 7 & 10 \end{array}\right) \xrightarrow{[2]\cdot 31} \left(\begin{array}{cc|c} 1 & 27 & 35 \\ 0 & 1 & 22 \end{array}\right) \xrightarrow{[1]-[2]\cdot 27} \left(\begin{array}{cc|c} 1 & 0 & 17 \\ 0 & 1 & 22 \end{array}\right)$$

Algorithm output

- Solving the system: $\begin{cases} 26x+3y=4 \\ 9x+34y=1 \end{cases} \pmod{36}$

$$\begin{array}{c} [1] \\ [2] \end{array} \left(\begin{array}{cc|c} 26 & 3 & 4 \\ 9 & 34 & 1 \end{array}\right) \xrightarrow[{[2]'=[1]\cdot 9+[2]\cdot 10}]{[1]'=[1]\cdot 35+[2]\cdot 3} \left(\begin{array}{cc|c} 1 & 27 & 35 \\ 0 & 7 & 10 \end{array}\right) \xrightarrow[{[2]'=[2]\cdot 31}]{[1]'=[1]+[2]\cdot 27} \left(\begin{array}{cc|c} 1 & 0 & 17 \\ 0 & 1 & 22 \end{array}\right)$$

- Inverse matrix computation:

$$\begin{array}{c} [1] \\ [2] \end{array} \left(\begin{array}{cc|cc} 26 & 3 & 1 & 0 \\ 9 & 34 & 0 & 1 \end{array}\right) \xrightarrow[{[2]'=[1]\cdot 9+[2]\cdot 10}]{[1]'=[1]\cdot 35+[2]\cdot 3} \left(\begin{array}{cc|cc} 1 & 27 & 35 & 3 \\ 0 & 7 & 9 & 10 \end{array}\right)$$

$$\begin{array}{c} [1] \\ [2] \end{array} \left(\begin{array}{cc|cc} 1 & 27 & 35 & 3 \\ 0 & 7 & 9 & 10 \end{array}\right) \xrightarrow[{[2]'=[2]\cdot 31}]{[1]'=[1]+[2]\cdot 27} \left(\begin{array}{cc|cc} 1 & 0 & 26 & 21 \\ 0 & 1 & 27 & 22 \end{array}\right)$$

Algorithm GGJE

- Input: $A = (a_{ij})_{n\times m},\ a_{ij} \in \mathbb{Z}_p$ *{Extended matrix}*
- Output: A *{transformed matrix}*

SUB *GGJE(A, n, m, p)*

FOR i=1 TO n DO

{zero elements below a_{ii} }

FOR $j=i+1$ TO n DO

$$\text{COMPUTE}\quad x', y', r', s' : \begin{Bmatrix} \mathrm{GCD}(a_{ii}, a_{ji}) = a_{ii}\cdot x' + a_{ji}\cdot y' \\ 0 = a_{ii}\cdot r' + a_{ji}\cdot s' \end{Bmatrix}$$

$$\begin{pmatrix} A(i,*) \\ A(j,*) \end{pmatrix} \leftarrow \begin{pmatrix} x' & y' \\ r' & s' \end{pmatrix} \times \begin{pmatrix} A(i,*) \\ A(j,*) \end{pmatrix}$$

END FOR *{for j}*

Algorithm GGJE

IF $GCD(a_{ii}, p)>1$

THEN exit *{singular matrix}*

ELSE

{zeroing elements above a_{ii} }

$$A(i,*) := A(i,*)\cdot a_{i,i}^{-1}$$

$$A(j,*) \leftarrow A(j,*) - A(i,*)\cdot a_{ji}, \quad j=\overline{1,i-1}$$

END IF

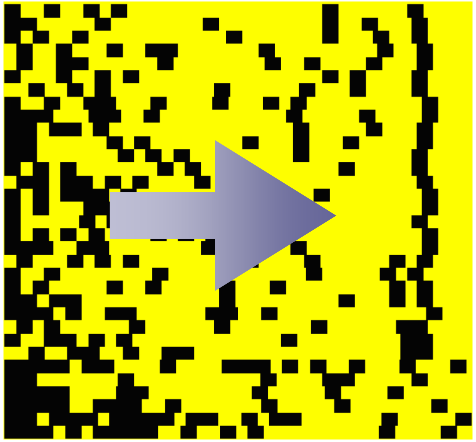
RETURN(A)

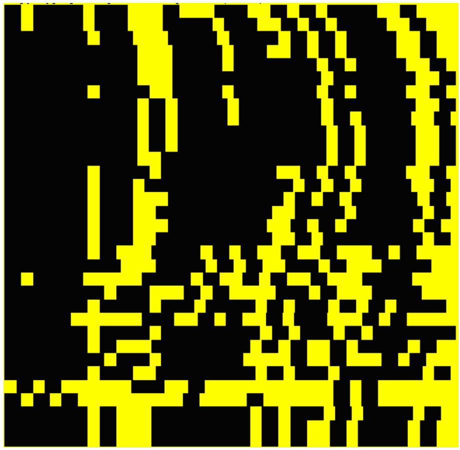
END SUB

Algorithm analysis

	Time complexity
Solving a number of systems over prime fields	$O\left(n\cdot(n\cdot m\cdot\sum_{k=1}^{t}\alpha_k+\log p)+\sqrt{\ln p\ln\ln p}\cdot e^{\sqrt{\ln p\ln\ln p}}\right)$
Solving a system of Diophantine equations	$O\left(n^2m^2\log p\right)$
Solving an equation over a matrix ring	$O(n^n)$
Generalized Jordan-Gaussian Elimination over residue rings	$O\left(n\cdot(nm+\log p)\right)$

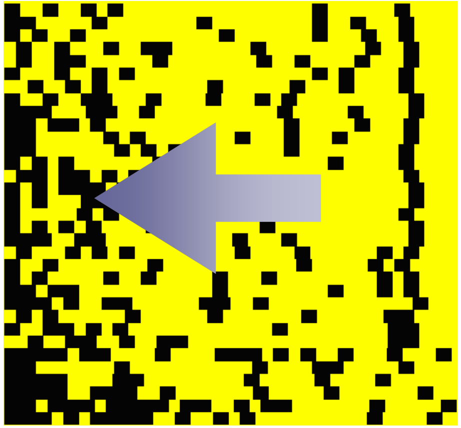
Further improvements

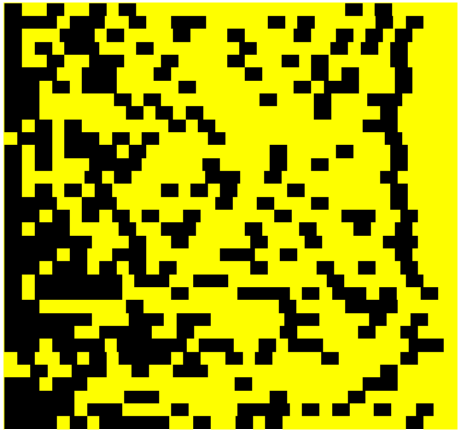




Fill-in: 65%

Non-zero elements ~P: 54%





Fill-in: 33%

Non-zero elements ~P: 13%

Experiments

- Set of non-zero coefficients at i^{th} iteration:

$$\mathrm{N}_i = \left\{ a_{kj} \in A_{n\times m} \mid a \neq 0,\ k = \overline{1, n},\ j = \overline{1,\ n-i} \right\}$$

- Set of big coefficients at i^{th} iteration:

$$\Lambda_i = \left\{ a_{kj} \in A_{n\times m} \mid a \neq 0,\ \log a_{ij} = O(\log p),\ k = \overline{1, n},\ j = \overline{1,\ n-i} \right\}$$

Experiments

- Set of non-zero coefficients at i^{th} iteration:

$$\mathrm{N}_i = \left\{ a_{kj} \in A_{n\times m} \mid a \neq 0,\ k = \overline{1,n},\ j = \overline{1,\ n-i} \right\}$$

- Set of big coefficients at i^{th} iteration:

$$\Lambda_i = \left\{ a_{kj} \in A_{n\times m} \mid a \neq 0,\ \log a_{ij} = O(\log p),\ k = \overline{1,n},\ j = \overline{1,\ n-i} \right\}$$

- Density

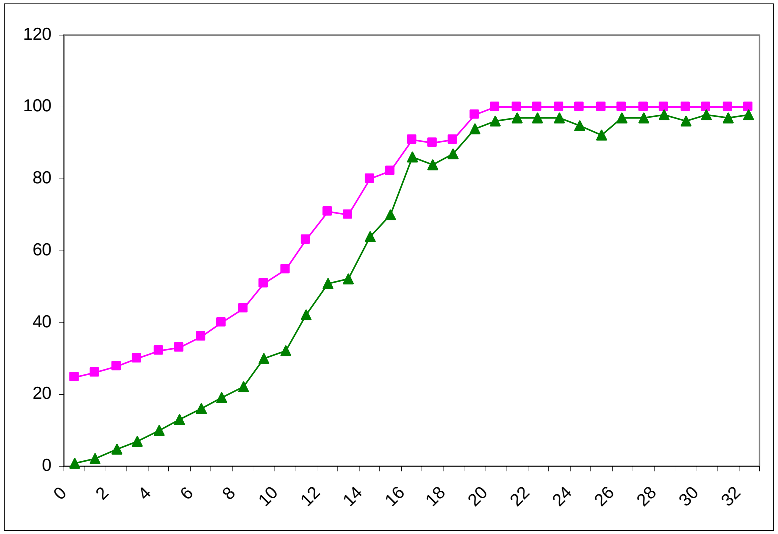
$$D(i) = \frac{|\mathrm{N}_i|}{n \cdot i} \cdot 100\%$$

- Magnitude

$$M(i) = \frac{|\Lambda_i|}{n \cdot i} \cdot 100\%$$

Implementation and results (1)

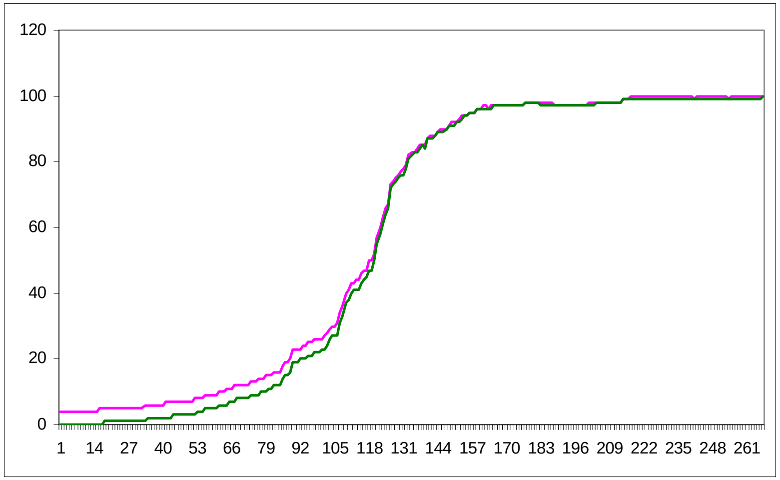
(1)



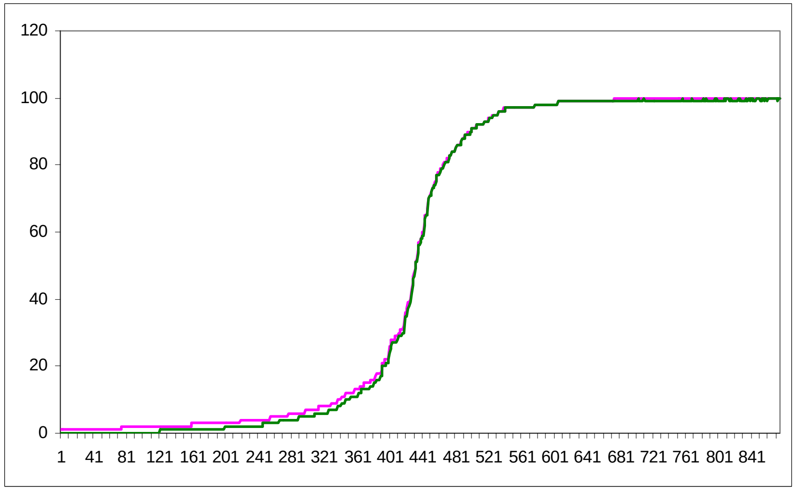
#	Size	N	P
1	Small	32	79833603
2	Medium	270	608658
3	Large	875	1237264621

Density	Magnitude

(2)

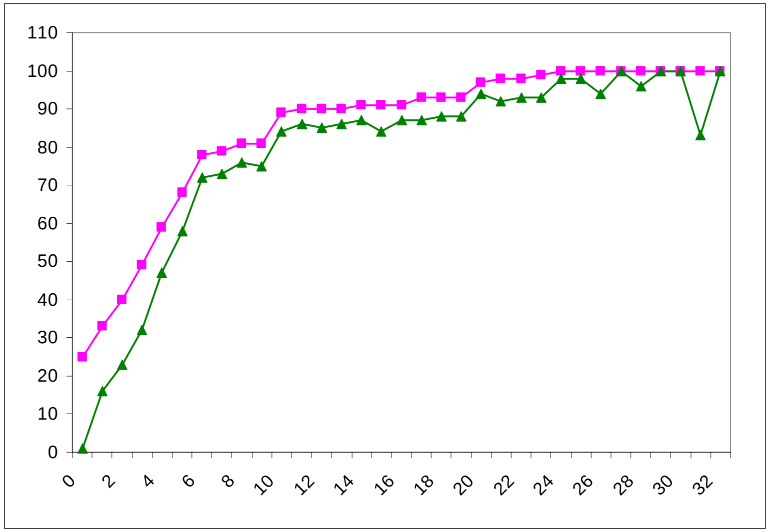


(3)



Implementation and results (2)

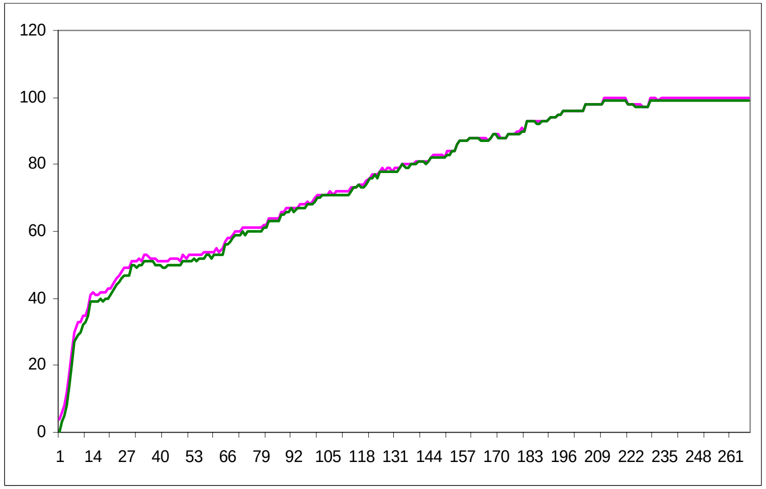
(1)



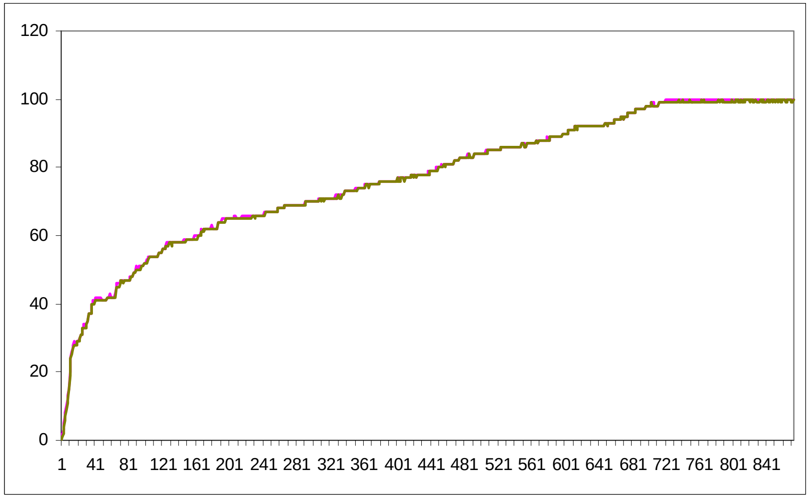
#	Size	N	P
1	Small	32	79833603
2	Medium	270	608658
3	Large	875	1237264621

Density **Magnitude**

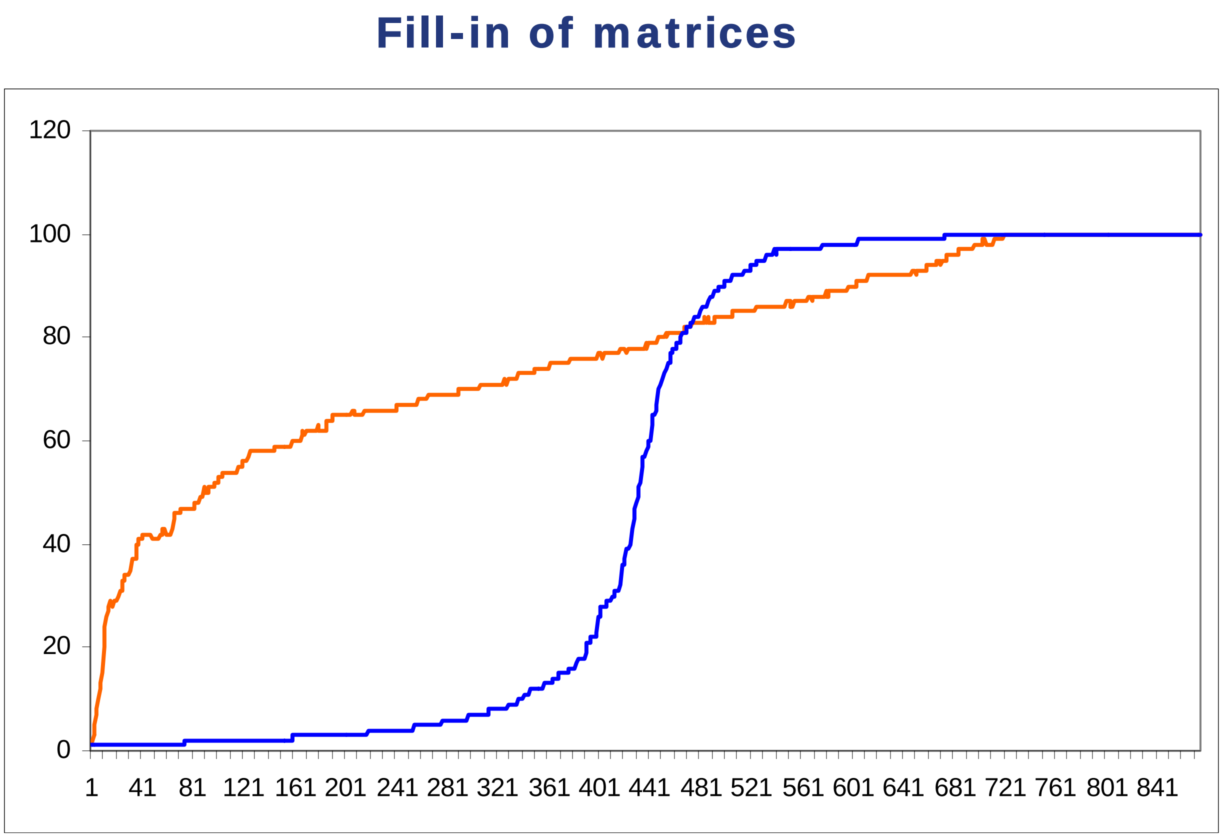
(2)



(3)



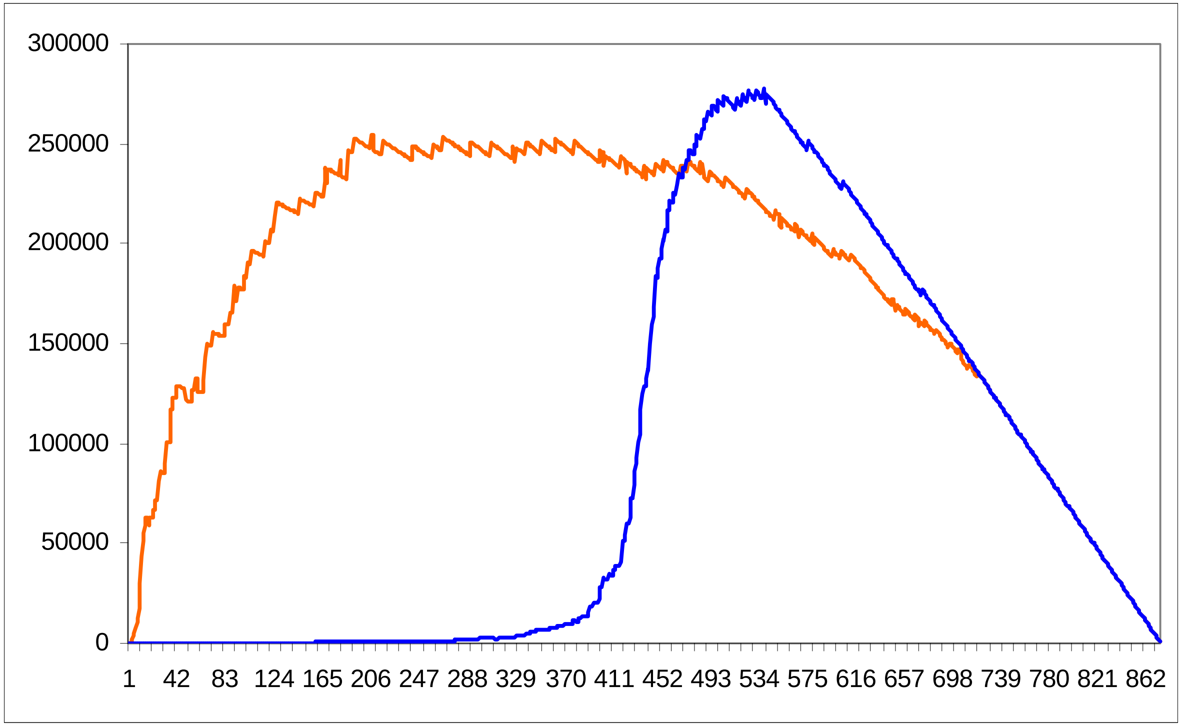
Implementation and results (3)



Left-to-right elimination Right-to-left elimination

Implementation and results (4)

Number of elementary operations



Left-to-right elimination **Right-to-left elimination**

Future work

- Further analysis of heuristic time complexity of the Generalized Gaussian-Jordan Elimination for residue rings
- Optimization problem of finding a transposition that minimizes the number of elementary operations
- Verifying hypotheses, e.g.
  - Time complexity of GGJE is better than $O(n(nm+\log P))$ for the matrices that occur in index-calculus algorithms
  - Slow-down of fill-in in sparse matrices transformed in reverse order reduces the number of operation by 1/2

Formal methods and tools for evaluating cryptographic systems security

alexandra.savelieva@gmail.com