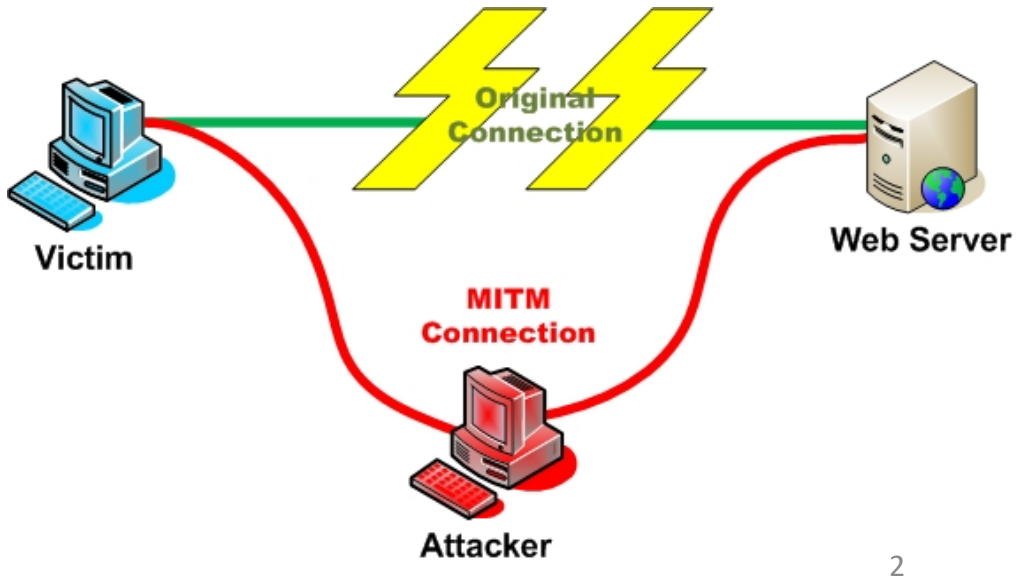
Evolutionary Computation Techniques for Constructing SAT-based Attacks in Algebraic Cryptanalysis

Artem Pavlenko, Alexander Semenov, Vladimir Ulyantsev
{alpavlenko,ulyantsev}@corp.ifmo.ru

ITMO University, St. Petersburg, Russia
ISDCT SB RAS, Irkutsk, Russia

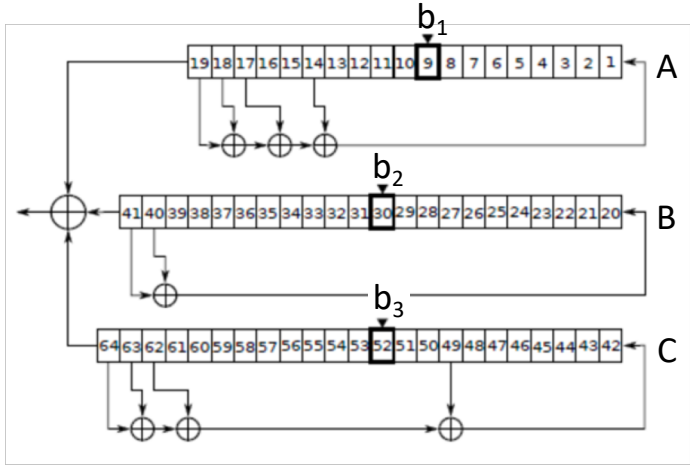
Cryptanalysis

- There are a lot of ways to encode and to decode information
  - HTTPS, mobile traffic …
  - man in the middle
- **Algebraic cryptanalysis** is a way of analyzing and breaking ciphers
- Type of attacks:
  - Brute-force attack
  - **Guess-and-determine** attack



Stream ciphers and cryptanalysis

Cipher A5/1 – used in 2G protocol



b_1, b_2, b_3 – clocking bits

$X = X_A \cup X_B \cup X_C$

$X = \{x_1, x_2, \ldots, x_{64}\}$

$Y = \{y_1, y_2, \ldots, y_{128}\}$

$$f: \{0,1\}^{64} \rightarrow \{0,1\}^{128}$$

$$f(x) = y$$

Research question: how practically hard it is to decrypt some encrypted text?

original text ⇄ encrypted text

fast

NP-hard

SAT and SAT-solvers

$$(\neg x1 \vee \neg x3 \vee \neg x4)$$
$$(x2 \vee x3 \vee \neg x4)$$
$$(x1 \vee \neg x2 \vee x4)$$
$$(x1 \vee x3 \vee x4)$$
$$(\neg x1 \vee x2 \vee \neg x3)$$

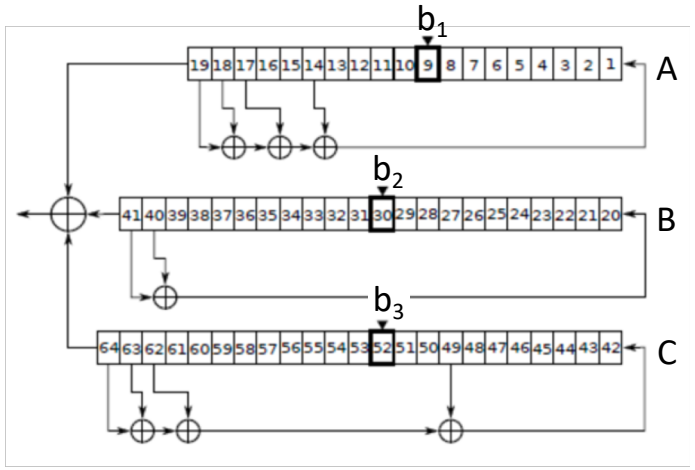
- Boolean SATisfiability – first known NP-complete problem
- A dozen of **applicable** SAT-solvers
  - minisat, lingeling, ROKK …
  - SAT, **UNSAT**
- Annular competitions in solving SAT!

⇓

good idea to **translate** hard problem to SAT

Encode to SAT using Transalg*

Cipher A5/1



b_1, b_2, b_3 – clocking bits

$X = X_A \cup X_B \cup X_C$

$X = \{x_1, x_2, \ldots, x_{64}\}$

$Y = \{y_1, y_2, \ldots, y_{128}\}$

manually ⇒

Transalg program

```
__in bit XA[19];
__in bit XB[22];
__in bit XC[23];
__out bit Y[128];

bit shift_rslosA() {
  bit x0 = XA[18]^XA[17]^XA[16]^XA[13];
  for(int j = 18; j > 0; j=j-1) {
    XA[j] = XA[j-1];
  }
  XA[0] = x0;
}
...

bit majority(bit A, bit B, bit C) {
    return A&B|A&C|B&C;
}

void main() {
  int b1 = 8;
  int b2 = 10;
  int b3 = 10;
  bit maj;
  for(int i = 0; i < 128; i=i+1) {
    maj = majority(XA[b1], XB[b2], XC[b3]);
    if(!(maj^XA[b1])) shift_rslosA();
    if(!(maj^XB[b2])) shift_rslosB();
    if(!(maj^XC[b3])) shift_rslosC();
    Y[i] = XA[18]^XB[21]^XC[22];
  }
}
```

automatically ⇒

SAT-formula

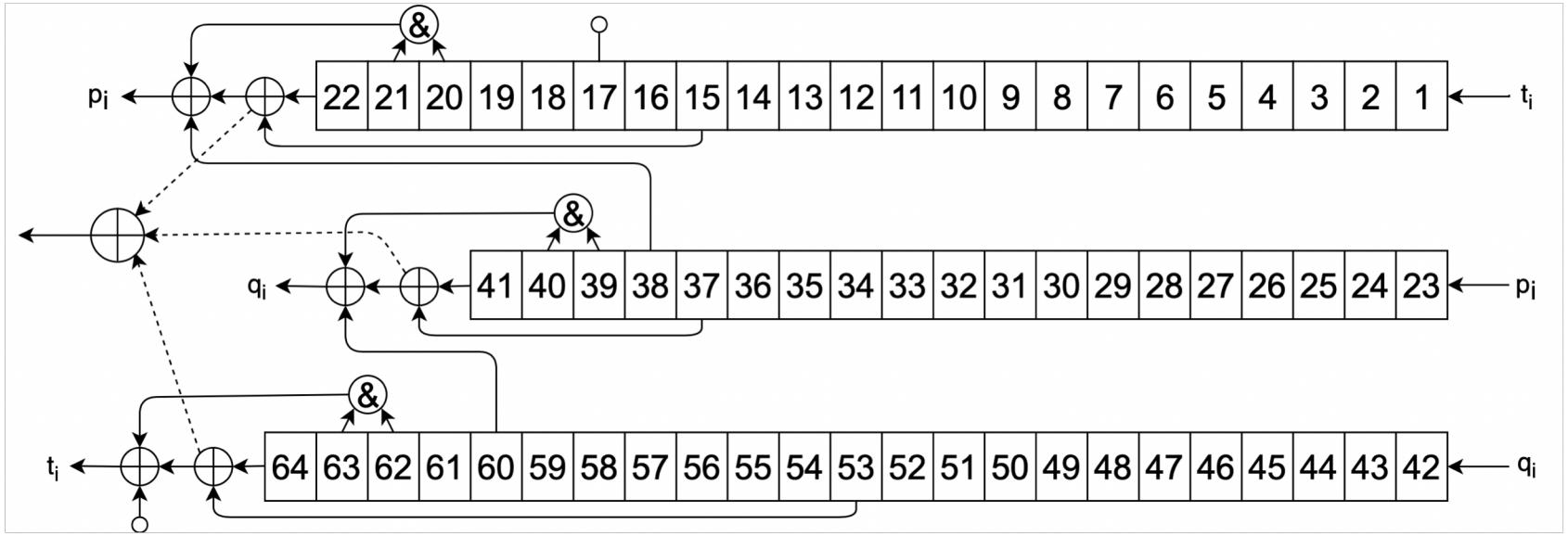
```
p cnf 8425 38262
c input variables 64
c literals count 128374
65 9 30 0
65 9 52 0
-65 9 -30 -52 0
65 -9 -52 0
-65 -9 30 52 0
65 -9 -30 0
66 -19 65 0
66 -18 -65 0
-66 19 65 0
-66 18 -65 0
67 -18 65 0
67 -17 -65 0
-67 18 65 0
-67 17 -65 0
68 -17 65 0
68 -16 -65 0
-68 17 65 0
-68 16 -65 0
69 -16 65 0
69 -15 -65 0
-69 16 65 0
... ...
38264 -8425 -8293 8295 -8297 0
38265 -8425 -8293 -8295 8297 0
```

*Transalg: [Otpuschennikov, I., Semenov, A., Gribanova, I., Zaikin, O., Kochemazov, S.: Encoding Cryptographic Functions to SAT Using TRANSALG System. In: ECAI 2016. FAIA, vol. 285, pp. 1594–1595 (2016)]

Example of breaking for Trivium 64

CPU:
AMD Opteron 6276
@ 2.3 GHz x32

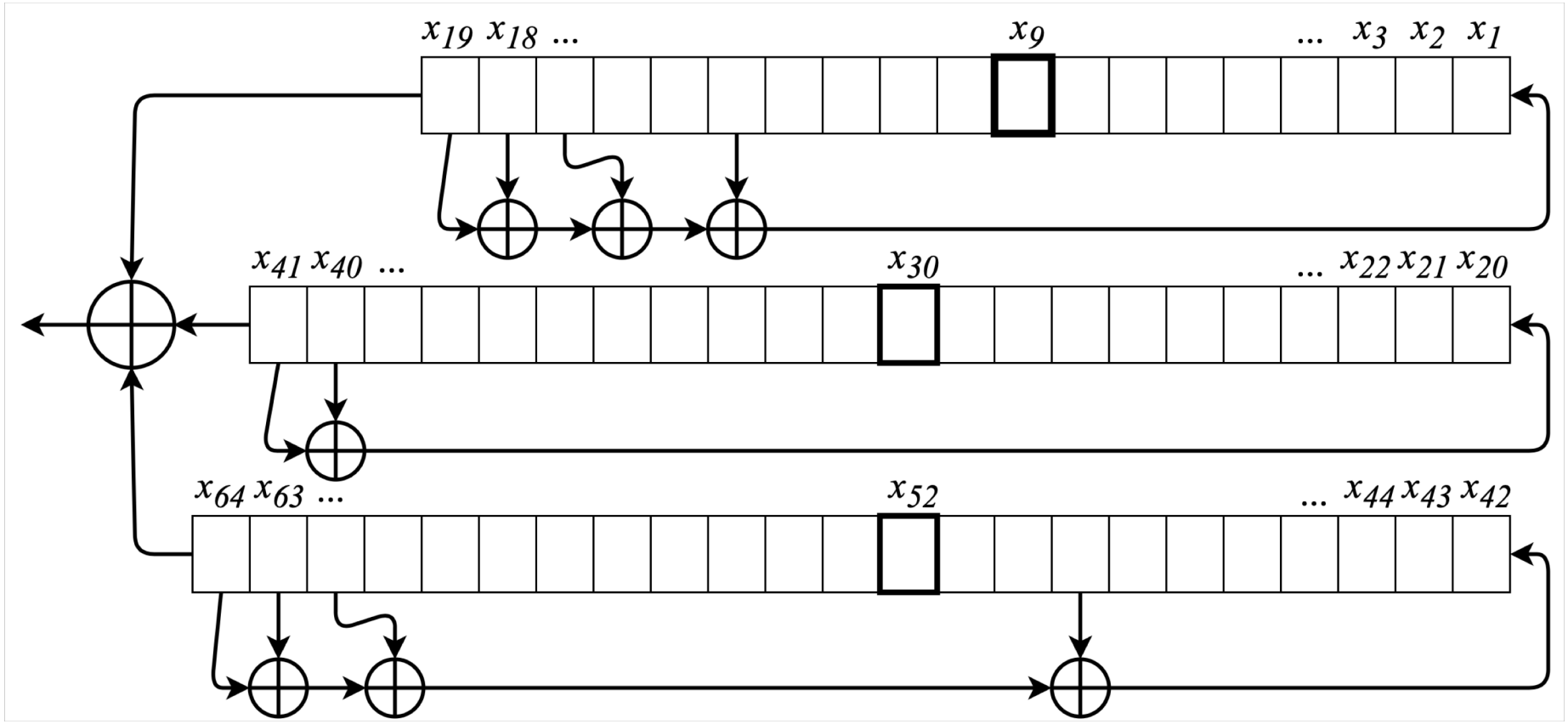
Timelimit:
7 days



	PLingeling	Treengeling	Guess-and-determine attack
task 1	interrupted	interrupted	2d 6h
task 2	interrupted	3d 2h	3d 19h
task 3	interrupted	4d 10h	15h
task 4	interrupted	interrupted	1d 21h
task 5	interrupted	interrupted	4d 3h

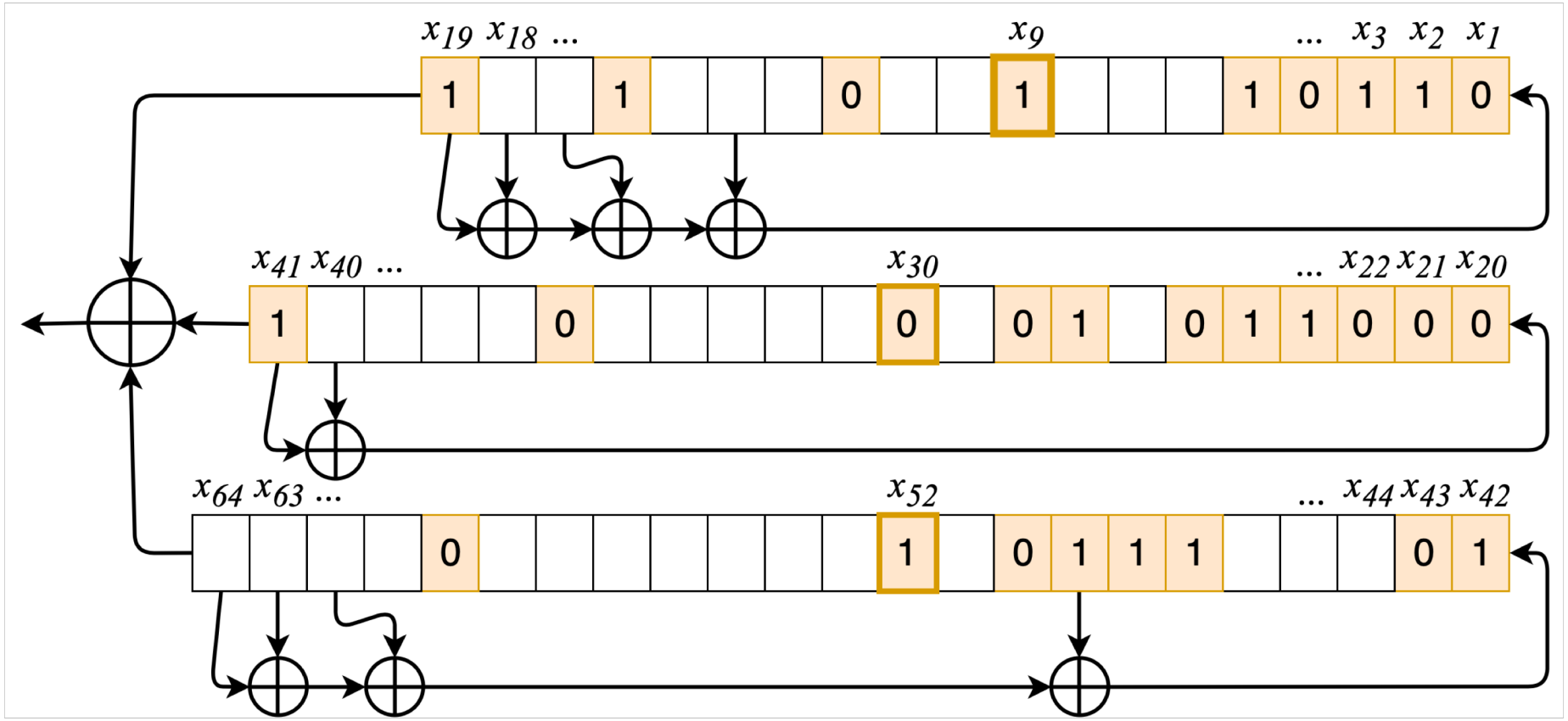
2. Guess-and-determine attacks

Guess-and-Determine. Backdoor



$$B = \{ x_1, x_2, x_3, x_4, x_5, x_9, x_{12}, x_{16}, x_{19}, x_{20}, x_{21}, x_{22}, x_{23}, x_{24}, x_{25}, x_{27}, x_{28}, x_{30}, x_{36}, x_{41}, x_{42}, x_{43}, x_{47}, x_{48}, x_{49}, x_{50}, x_{52}, x_{60} \}$$

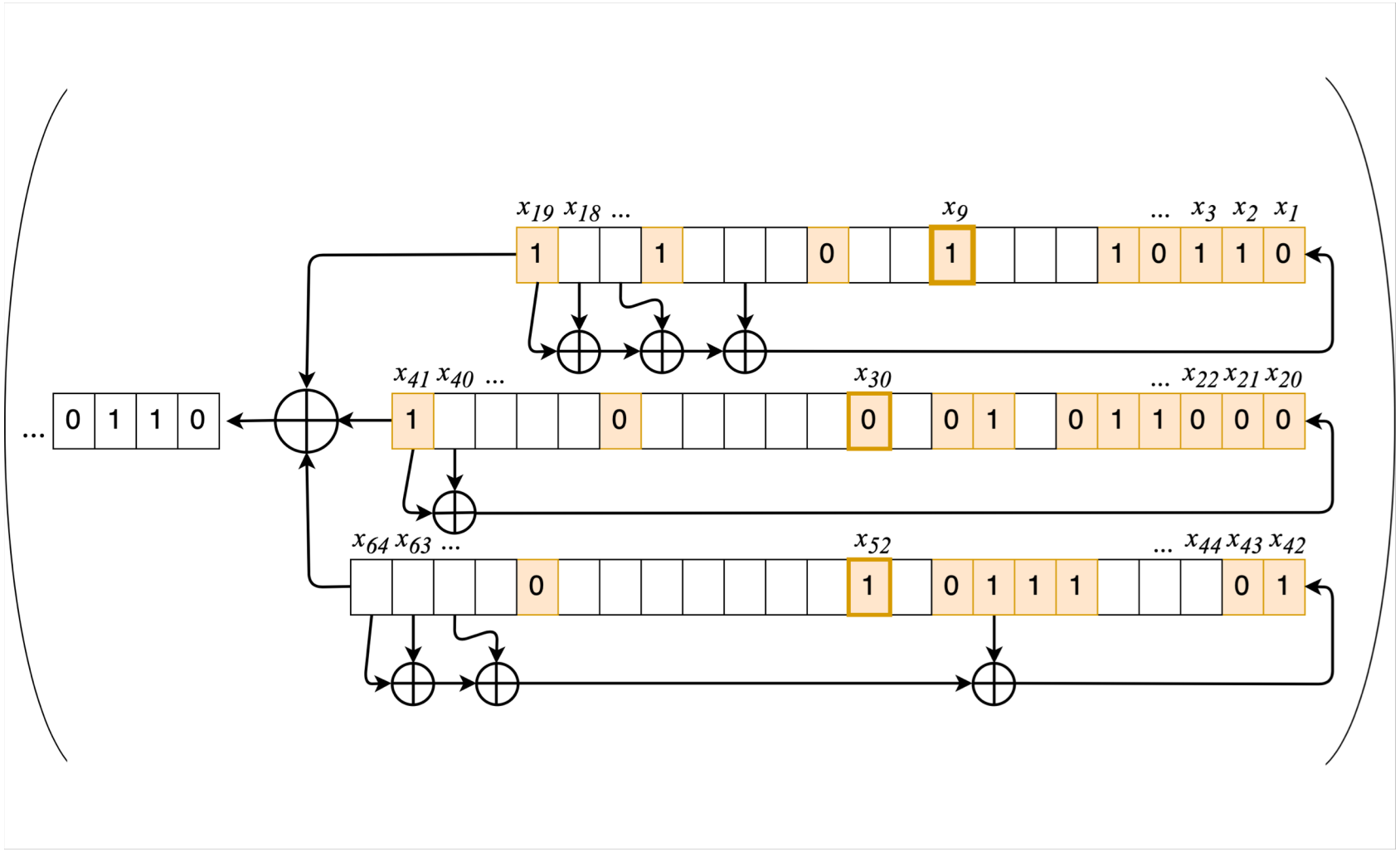
Guess-and-Determine. Guess



$$B = \{ x_1, x_2, x_3, x_4, x_5, x_9, x_{12}, x_{16}, x_{19}, x_{20}, x_{21}, x_{22}, x_{23}, x_{24}, x_{25}, x_{27}, x_{28}, x_{30}, x_{36}, x_{41}, x_{42}, x_{43}, x_{47}, x_{48}, x_{49}, x_{50}, x_{52}, x_{60} \}$$

Guess-and-Determine. Determine

solver.*solve*

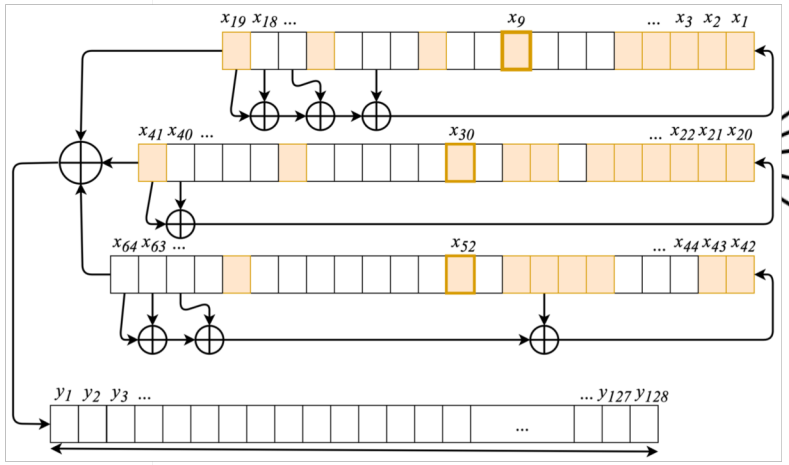


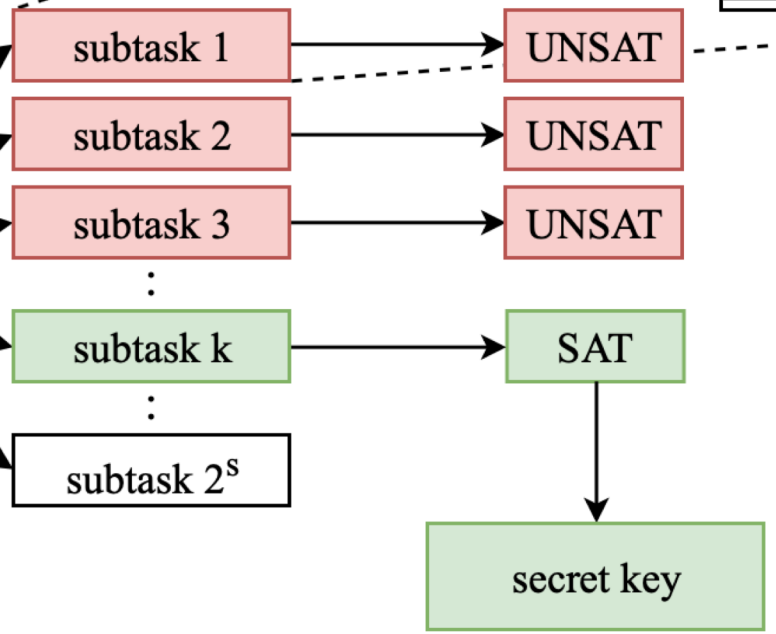
⇒ Result: UNSAT
Time: 1.243 c

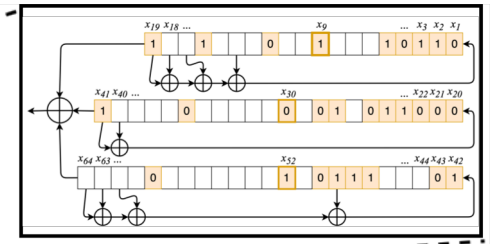
Guess-and-Determine. Definition

$$\sum_{i=1}^{2^s} \tau_i \ll T_{BruteForce},$$

where $s = |B|$

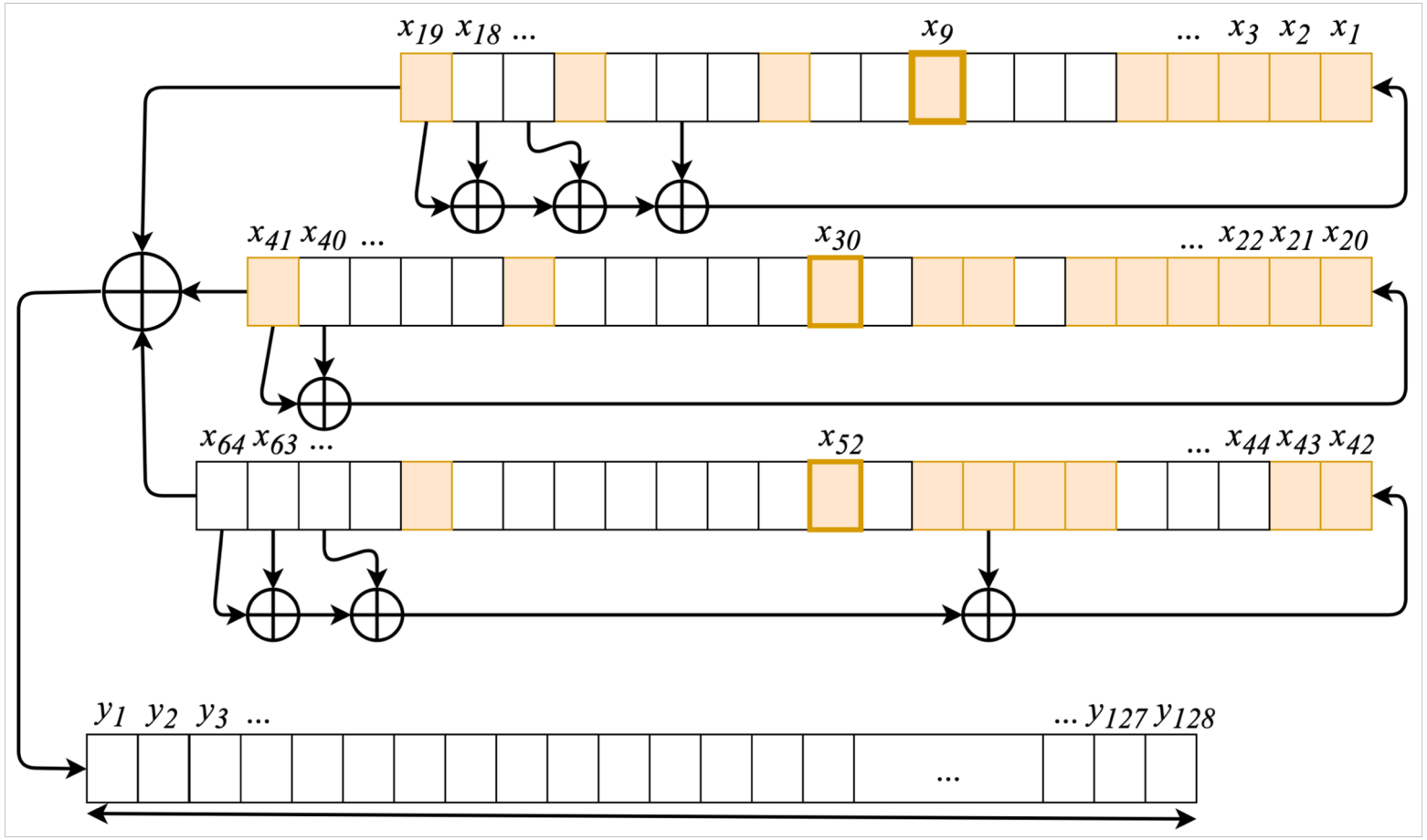




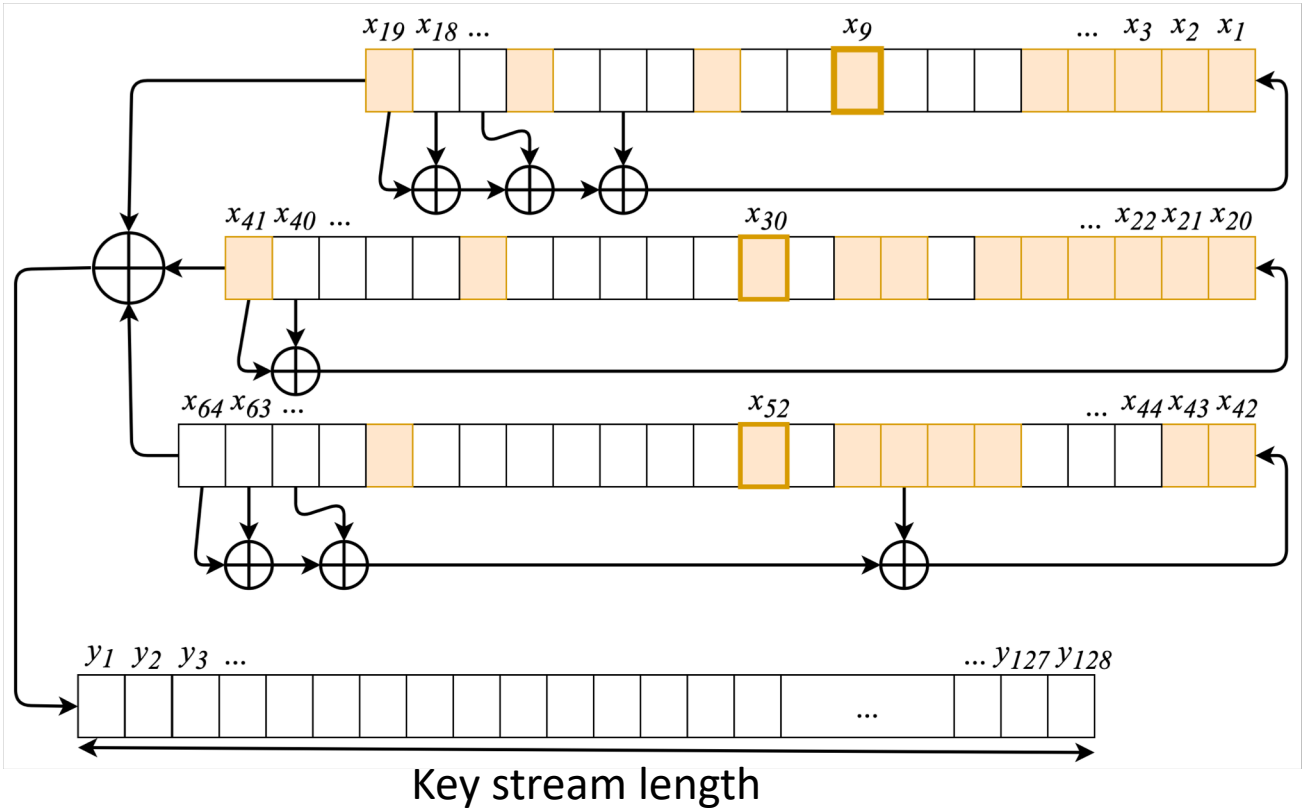


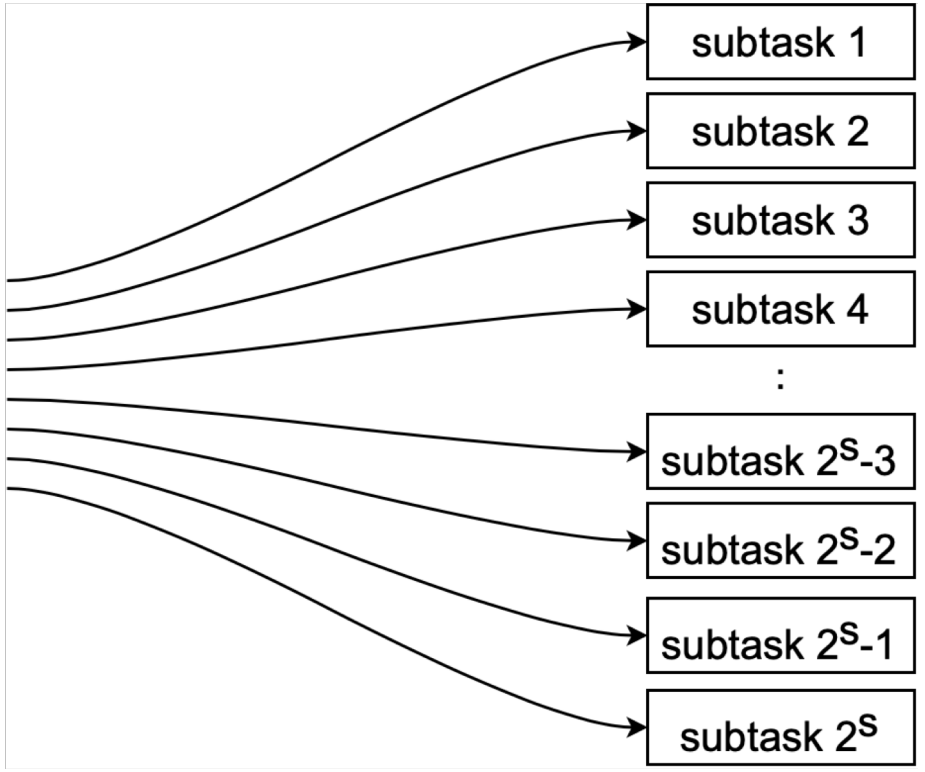
$\tau_1 = 1.243$ c

How to construct a efficient backdoor?



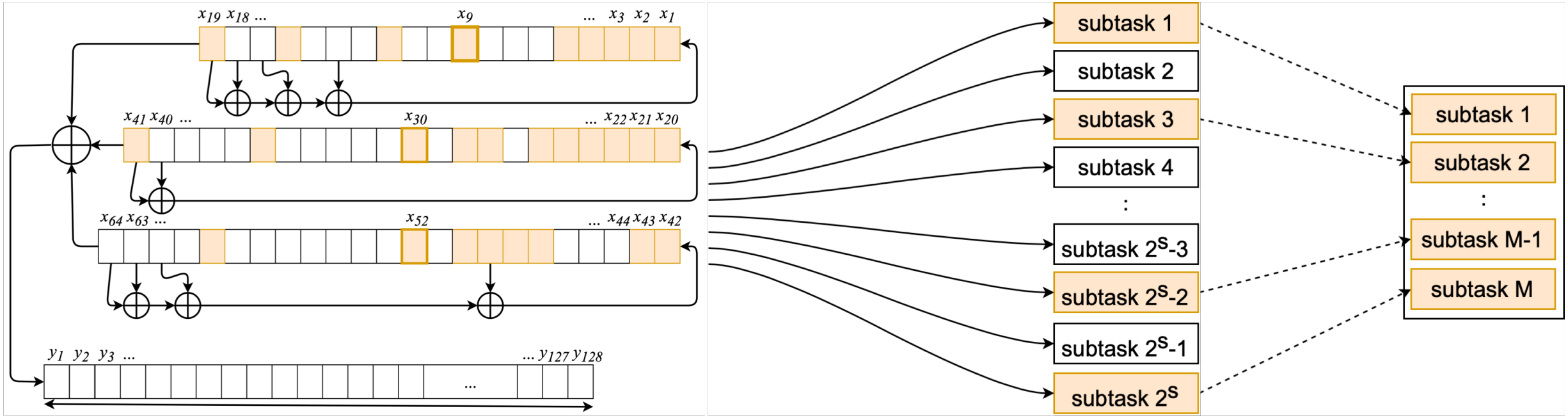
Backdoor-based Decomposition





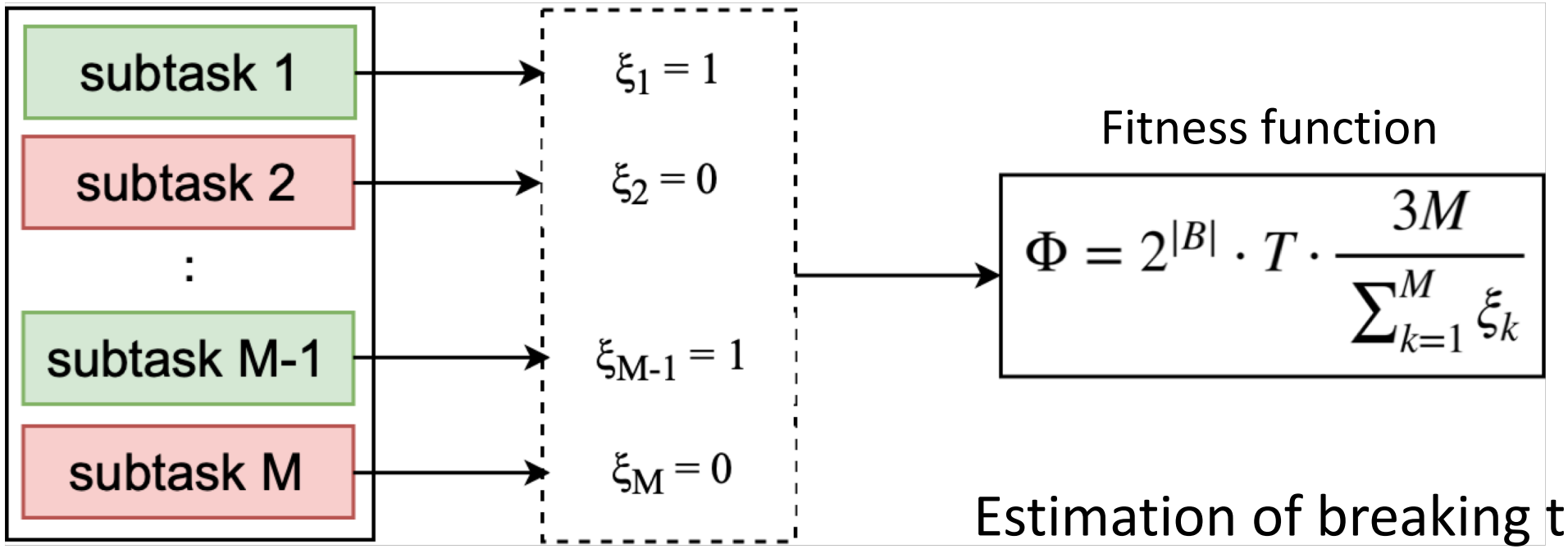
s = |B| – power of backdoor set

Monte-Carlo Sampling



Evaluating

If the task is solved in time T, then $\xi = 1$, else $\xi = 0$



Estimation of breaking time

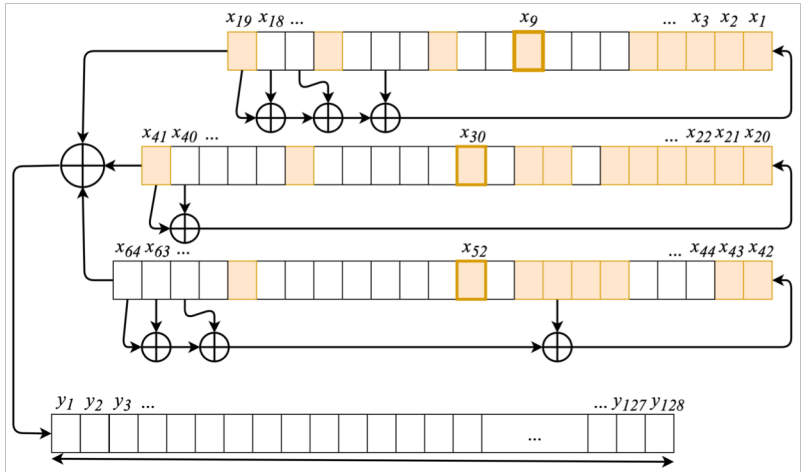
=

Fitness value

Estimation technics: [Semenov, A., Zaikin, O., Otpuschennikov, I., Kochemazov, S., Ignatiev, A.: On Cryptographic Attacks Using Backdoors for SAT. In: Proc. of AAAI 2018. pp. 6641–6648 (2018)]

Intermediate sum-up

- Analyzing **stream cyphers** is a hard problem
- We can **translate** the attack to **SAT**
- We can **speedup** the SAT-based attack using **backdoor**
- Selecting the **efficient** backdoor is a hard problem
- But there is a way to **estimate** the attack time for a given backdoor
- **Where are evolutionary algorithms?!**



magic

Estimation of breaking time

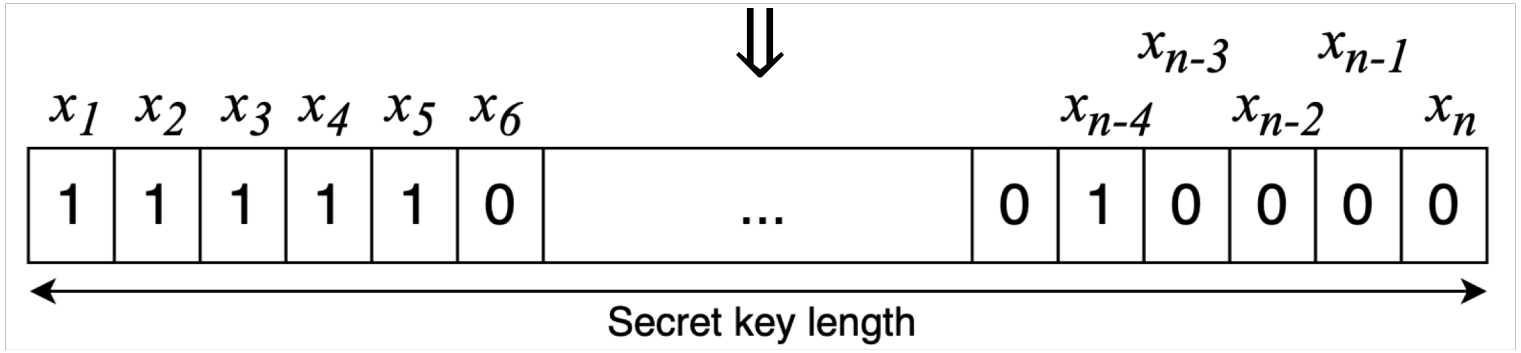
3. Applying EA to construct an efficient backdoor

Metaheuristic Algorithms

Applied to us	We apply
Tabu Search*	Evolutionary Computation
Simulated Annealing	

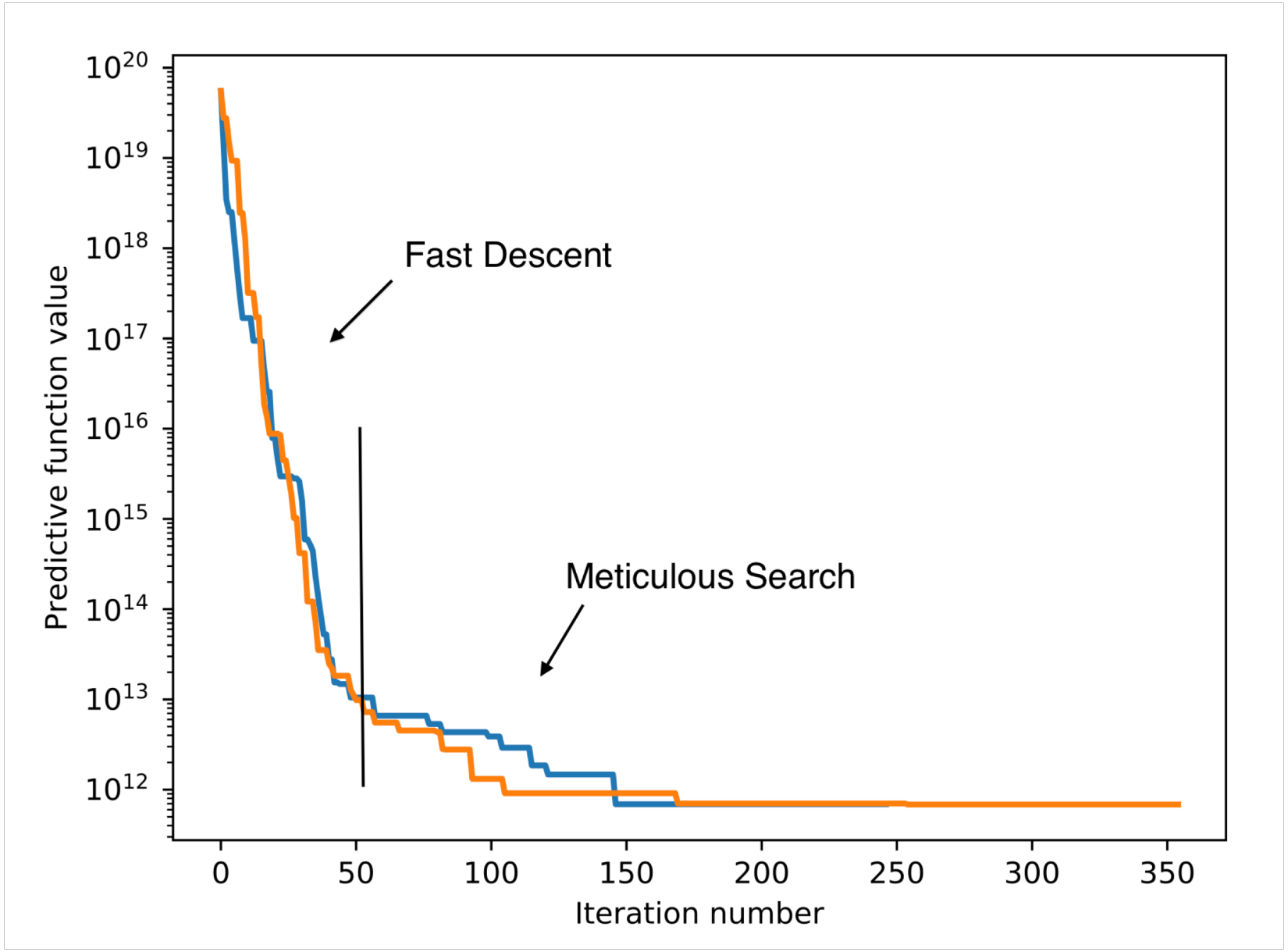
Individual: bit vector, which presents a set of guessed bits

$$B = \{ x_1, x_2, x_3, x_4, x_5, x_9, x_{12}, x_{16}, x_{19}, x_{20}, x_{21}, x_{22}, x_{23}, x_{24}, x_{25}, x_{27}, x_{28}, x_{30}, x_{36}, x_{41}, x_{42}, x_{43}, x_{47}, x_{48}, x_{49}, x_{50}, x_{52}, x_{60} \}$$

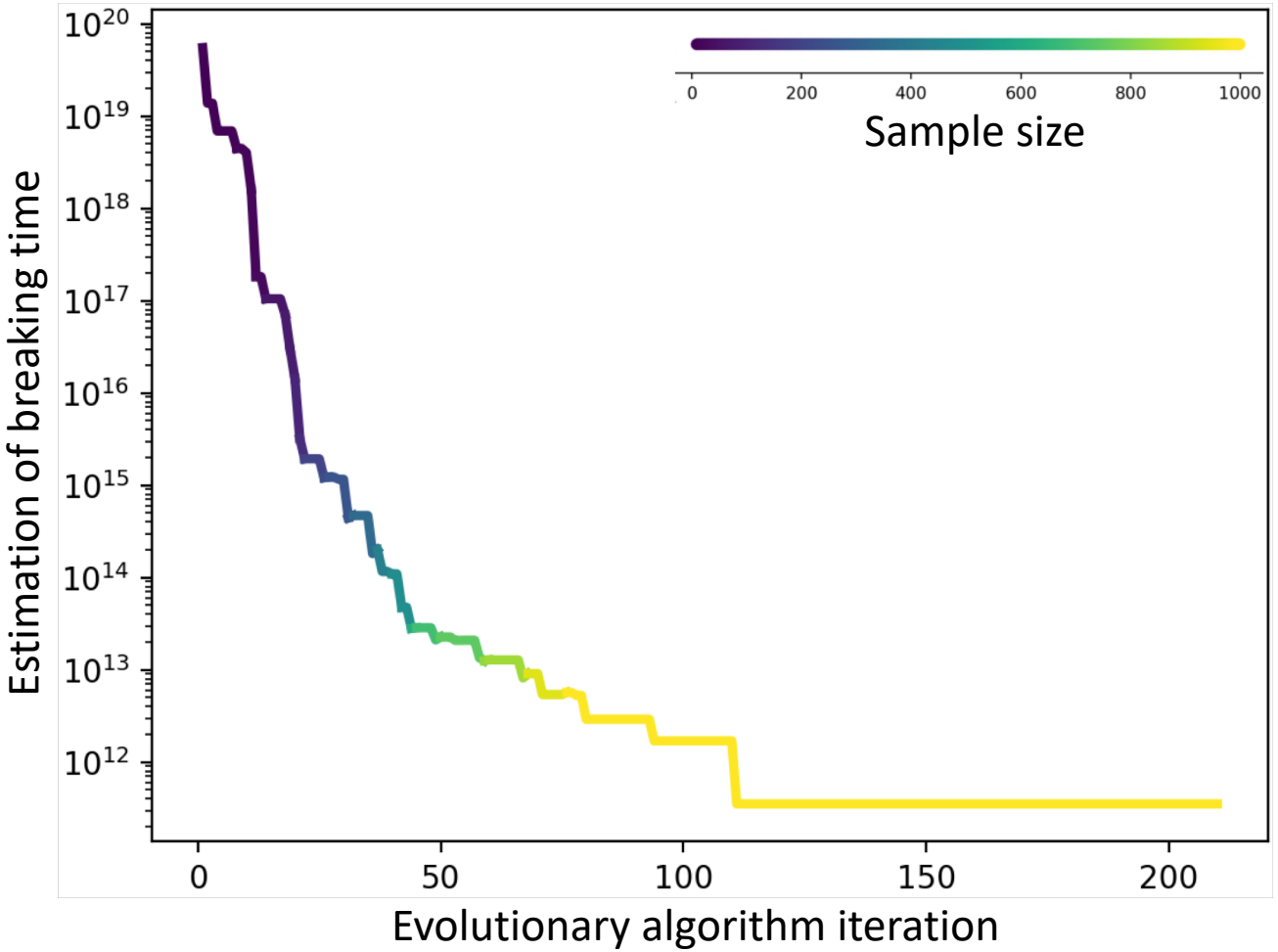


*Tabu Search Appling: [Semenov, A., Zaikin, O.: Algorithm for Finding Partitionings of Hard Variants of Boolean Satisfiability Problem with Application to Inversion of Some Cryptographic Functions. SpringerPlus 5(1), 554 (2016)]

Two Phases

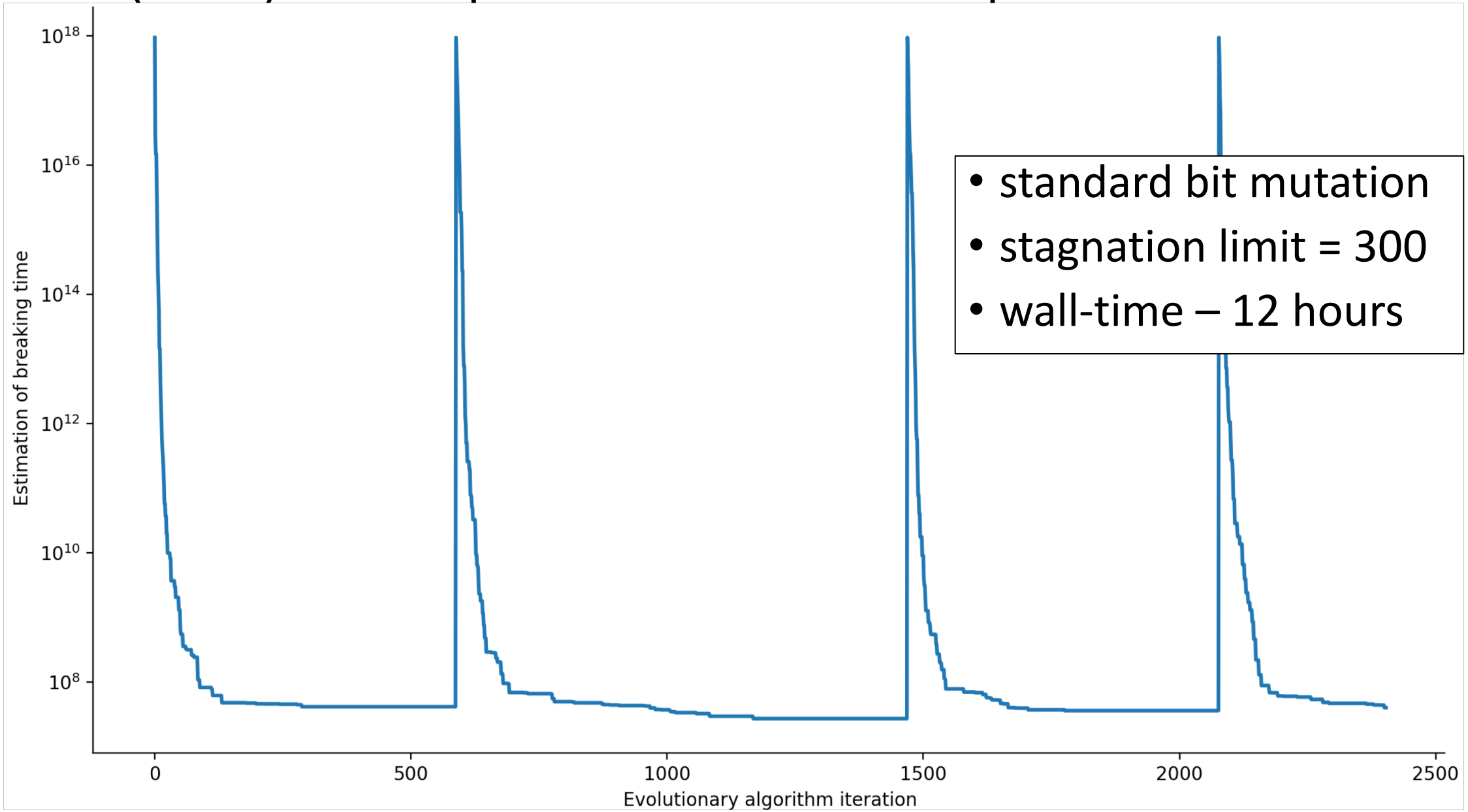

Predictive function value
10^{20}
10^{19}
10^{18}
10^{17}
10^{16}
10^{15}
10^{14}
10^{13}
10^{12}
Fast Descent
Meticulous Search
0
50
100
150
200
250
300
350
Iteration number

Adaptation Strategy



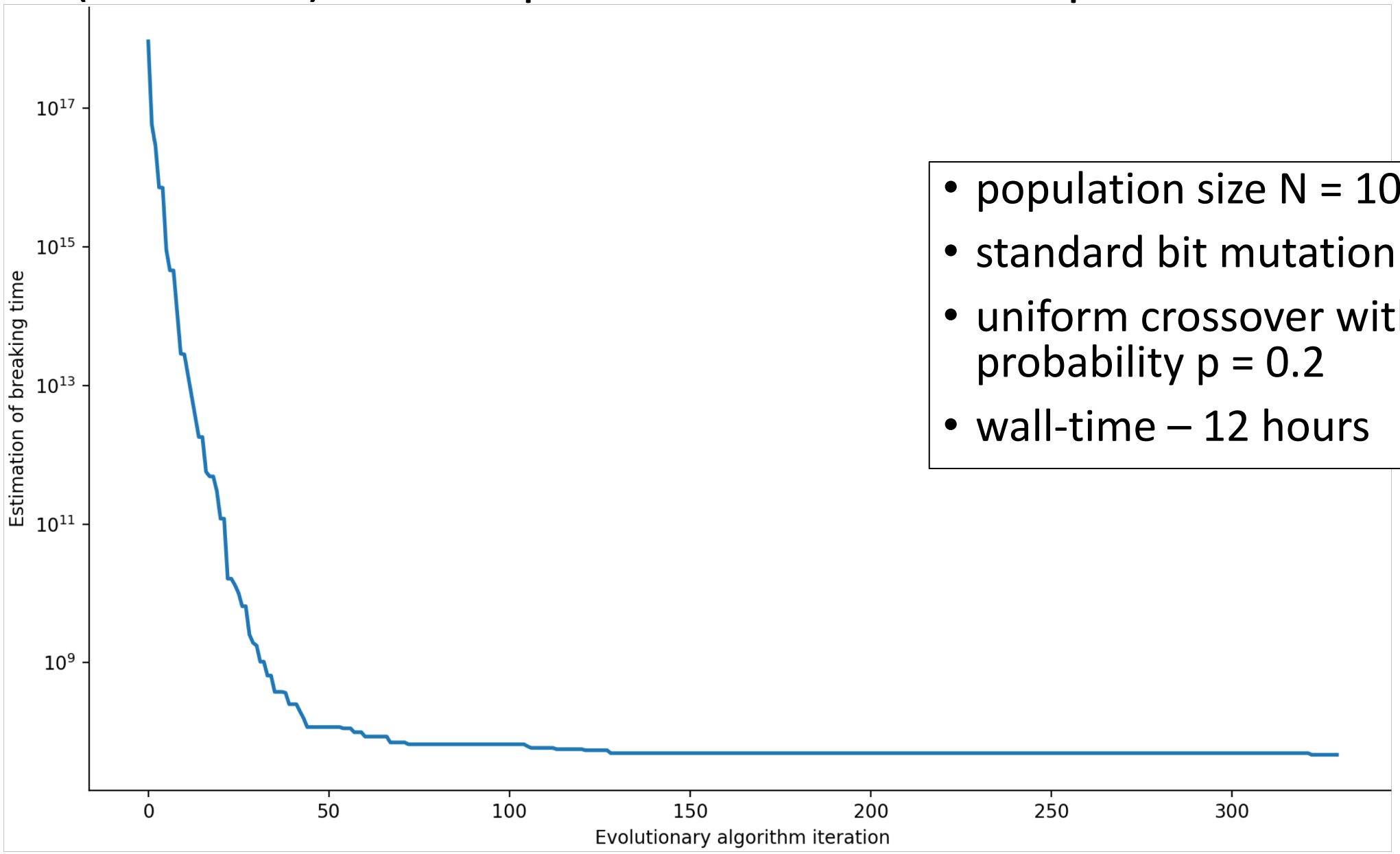
- Algorithm starts with Monte-Carlo sample size M = 10
- M is gradually increases to 1000 with the decrease of the fitness value

EA (1+1) example. Trivium 64 cipher



- standard bit mutation
- stagnation limit = 300
- wall-time – 12 hours

GA (Elitism) example. Trivium 64 cipher



- population size N = 10
- standard bit mutation
- uniform crossover with probability p = 0.2
- wall-time – 12 hours

Experimental results

	Tabu Search		(1+1)-EA		GA	
	|B|	Attack time (s)	|B|	Attack time (s)	|B|	Attack time (s)
Trivium-Toy 64/75	17	4.30e+07	21	**3.19e+07**	22	5.36+07
Trivium-Toy 96/100	34	3.14e+12	33	1.28e+13	40	**2.09+12**
Bivium 177/200	40	4.29e+12	32	2.60e+12	39	**1.49+12**
ASG 72/76	8	**5601.33**	9	5604.8	8	6155.19
ASG 96/112	14	3.95e+06	13	6.76e+06	16	**3.72e+06**
ASG 192/200	47	**1.14e+16**	47	2.27e+18	44	2.84e+17

Conclusion

- We used **(1+1)-EA** and **GA** to construct SAT-based guess-and-determine attacks on cryptographic ciphers.
- We proposed a sample size **adaptation strategy** to increase the number of individuals that the algorithm processes during a fixed time budget.
- **Backdoors** have been found, some of them are better than those found earlier, but estimation of breaking time is still very long.

- Another paper accepted to GECCO'19, see you there :)
- Supposed by the Russian Science Foundation (project No 18-71-00150)

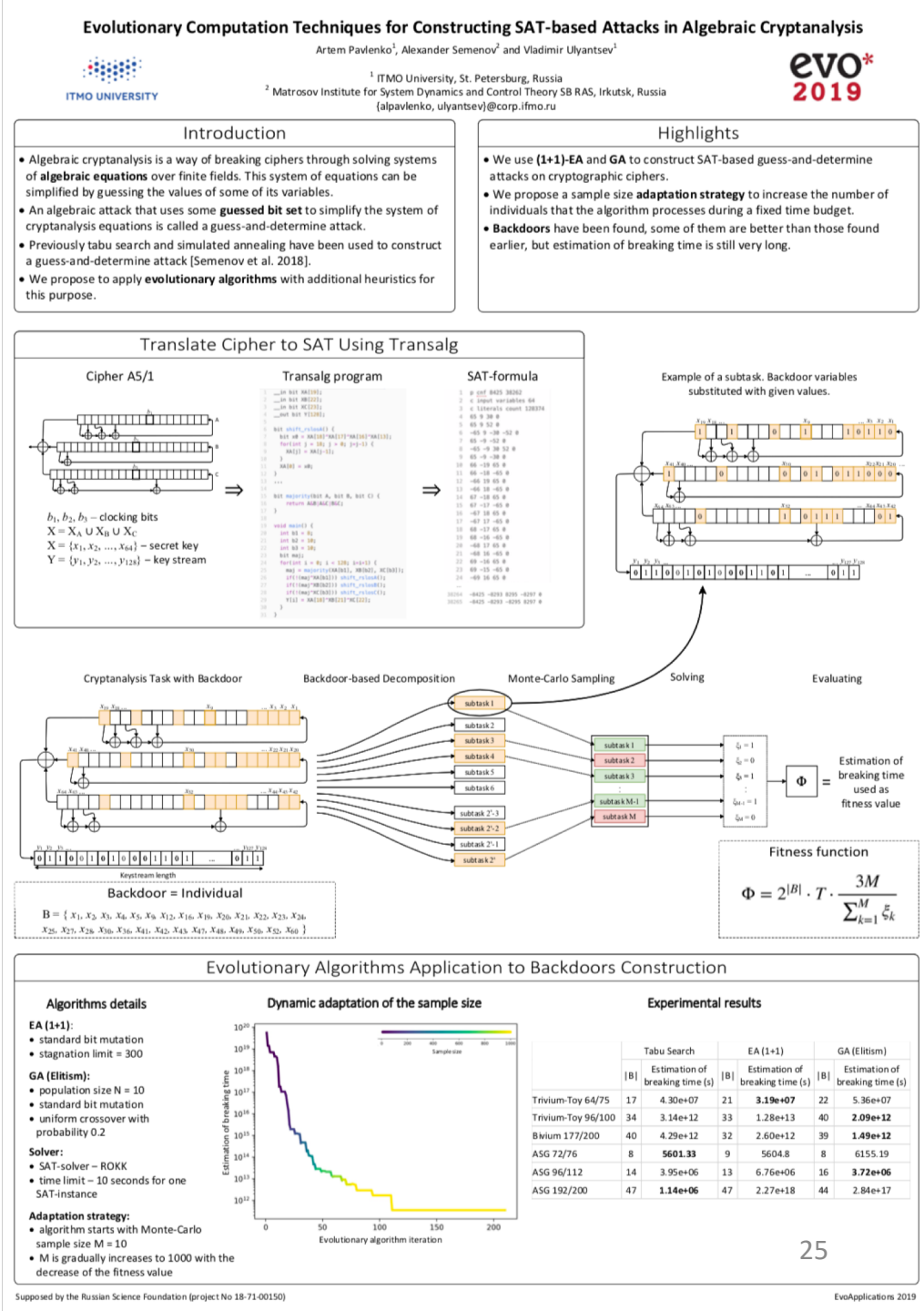
Thank you for attention!

CT Lab
ITMO UNIVERSITY

ITMO UNIVERSITY

Artem Pavlenko,
Alexander Semenov,
Vladimir Ulyantsev

{alpavlenko,ulyantsev}@corp.ifmo.ru
instagram.com/itmo.ctlab



Evolutionary Computation Techniques for Constructing SAT-based Attacks in Algebraic Cryptanalysis

Artem Pavlenko[1], Alexander Semenov[2] and Vladimir Ulyantsev[1]

[1] ITMO University, St. Petersburg, Russia
[2] Matrosov Institute for System Dynamics and Control Theory SB RAS, Irkutsk, Russia
{alpavlenko, ulyantsev}@corp.ifmo.ru

ITMO UNIVERSITY

evo* 2019

Introduction

- Algebraic cryptanalysis is a way of breaking ciphers through solving systems of **algebraic equations** over finite fields. This system of equations can be simplified by guessing the values of some of its variables.
- An algebraic attack that uses some **guessed bit set** to simplify the system of cryptanalysis equations is called a guess-and-determine attack.
- Previously tabu search and simulated annealing have been used to construct a guess-and-determine attack [Semenov et al. 2018].
- We propose to apply **evolutionary algorithms** with additional heuristics for this purpose.

Highlights

- We use **(1+1)-EA** and **GA** to construct SAT-based guess-and-determine attacks on cryptographic ciphers.
- We propose a sample size **adaptation strategy** to increase the number of individuals that the algorithm processes during a fixed time budget.
- **Backdoors** have been found, some of them are better than those found earlier, but estimation of breaking time is still very long.

Translate Cipher to SAT Using Transalg

Cipher A5/1 — Transalg program — SAT-formula

b_1, b_2, b_3 – clocking bits
$X = X_A \cup X_B \cup X_C$
$X = \{x_1, x_2, \ldots, x_{64}\}$ – secret key
$Y = \{y_1, y_2, \ldots, y_{128}\}$ – key stream

Example of a subtask. Backdoor variables substituted with given values.

Cryptanalysis Task with Backdoor — Backdoor-based Decomposition — Monte-Carlo Sampling — Solving — Evaluating

Keystream length

Backdoor = Individual

$B = \{ x_1, x_3, x_5, x_6, x_9, x_{12}, x_{16}, x_{19}, x_{20}, x_{21}, x_{22}, x_{23}, x_{24}, x_{25}, x_{27}, x_{28}, x_{30}, x_{36}, x_{41}, x_{42}, x_{43}, x_{47}, x_{48}, x_{49}, x_{50}, x_{52}, x_{60} \}$

Estimation of breaking time used as fitness value

Fitness function

$$\Phi = 2^{|B|} \cdot T \cdot \frac{3M}{\sum_{k=1}^{M} \xi_k}$$

Evolutionary Algorithms Application to Backdoors Construction

Algorithms details

EA (1+1):
- standard bit mutation
- stagnation limit = 300

GA (Elitism):
- population size N = 10
- standard bit mutation
- uniform crossover with probability 0.2

Solver:
- SAT-solver – ROKK
- time limit – 10 seconds for one SAT-instance

Adaptation strategy:
- algorithm starts with Monte-Carlo sample size M = 10
- M is gradually increases to 1000 with the decrease of the fitness value

Dynamic adaptation of the sample size

Estimation of breaking time — Evolutionary algorithm iteration — Sample size

Experimental results

	Tabu Search		EA (1+1)		GA (Elitism)	
	|B|	Estimation of breaking time (s)	|B|	Estimation of breaking time (s)	|B|	Estimation of breaking time (s)
Trivium-Toy 64/75	17	4.30e+07	21	**3.19e+07**	22	5.36e+07
Trivium-Toy 96/100	34	3.14e+12	33	1.28e+13	40	**2.09e+12**
Bivium 177/200	40	4.29e+12	32	2.60e+12	39	**1.49e+12**
ASG 72/76	8	**5601.33**	9	5604.8	8	6155.19
ASG 96/112	14	3.95e+06	13	6.76e+06	16	**3.72e+06**
ASG 192/200	47	**1.14e+06**	47	2.27e+18	44	2.84e+17

Supposed by the Russian Science Foundation (project No 18-71-00150)