Fitness Comparison by Statistical Testing in Construction of SAT-Based Guess-and-Determine Cryptographic Attacks

Artem Pavlenko, Maxim Buzdalov, Vladimir Ulyantsev

GECCO 2019, July 16

Symmetric cryptography

Plaintext | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 |

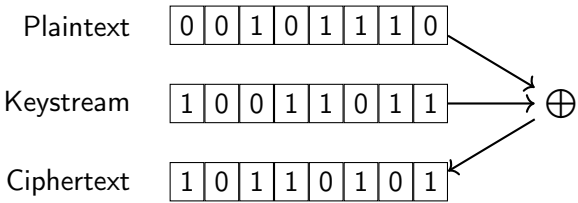
Alice wants to send a secret message to Bob.

Symmetric cryptography

Plaintext	0	0	1	0	1	1	1	0
Keystream	1	0	0	1	1	0	1	1

To do that, she generates a random sequence.

Symmetric cryptography

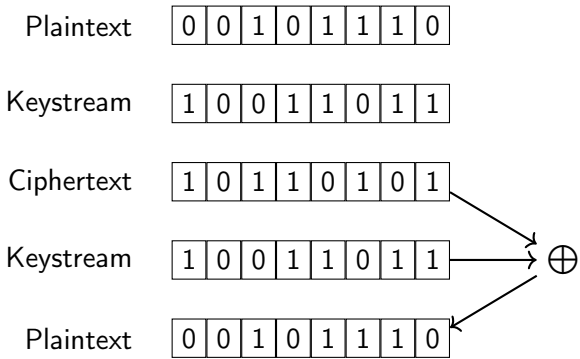


Symmetric cryptography

Plaintext	0	0	1	0	1	1	1	0
Keystream	1	0	0	1	1	0	1	1
Ciphertext	1	0	1	1	0	1	0	1
Keystream	1	0	0	1	1	0	1	1

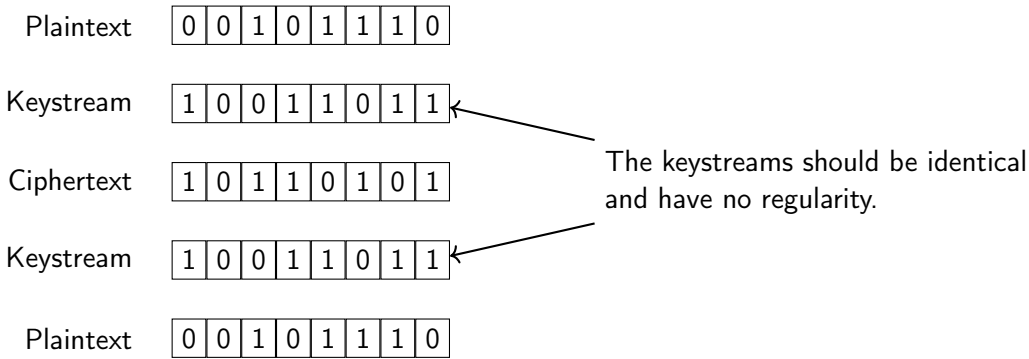
Bob also generates the same random sequence...

Symmetric cryptography

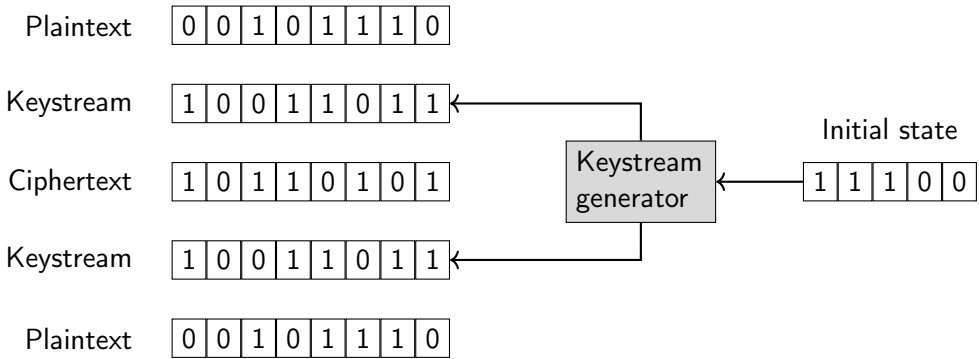


and also applies bitwise XOR
to recover the message.

Symmetric cryptography



Symmetric cryptography

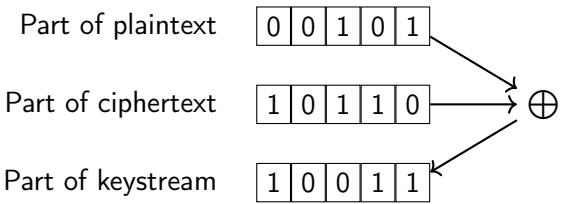


Attack on the keystream generator

Part of plaintext	0	0	1	0	1
Part of ciphertext	1	0	1	1	0

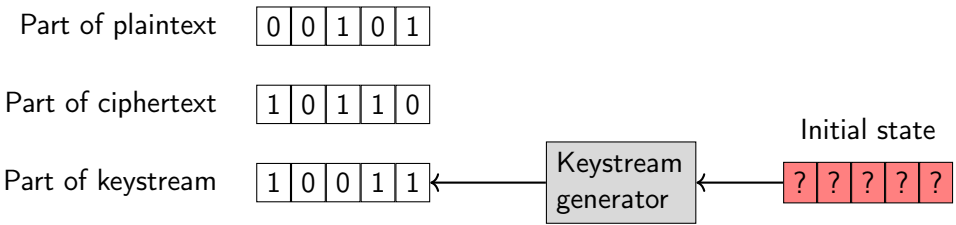
Eve has eavesdropped matching parts of plaintext and ciphertext.

Attack on the keystream generator

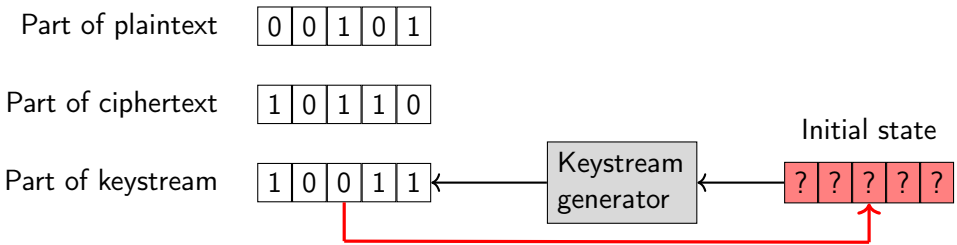


She applies bitwise XOR to reveal a part of keystream.

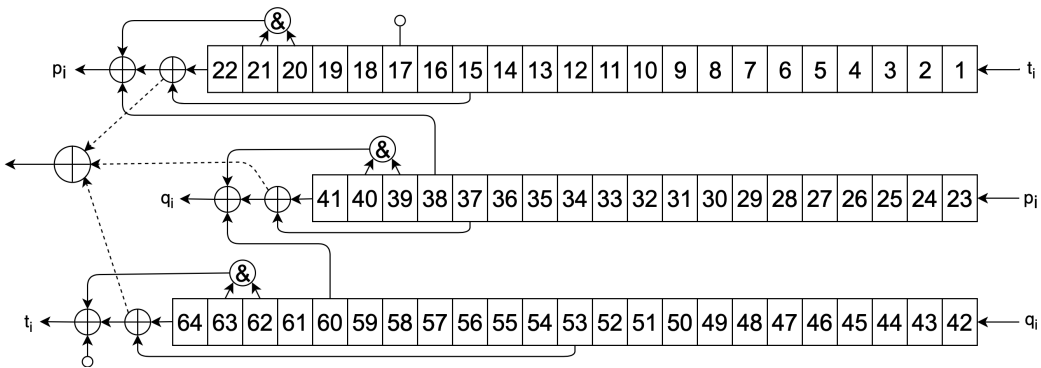
Attack on the keystream generator



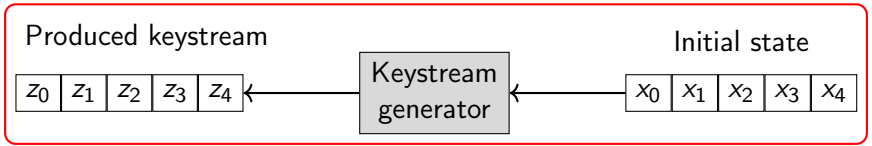
Attack on the keystream generator



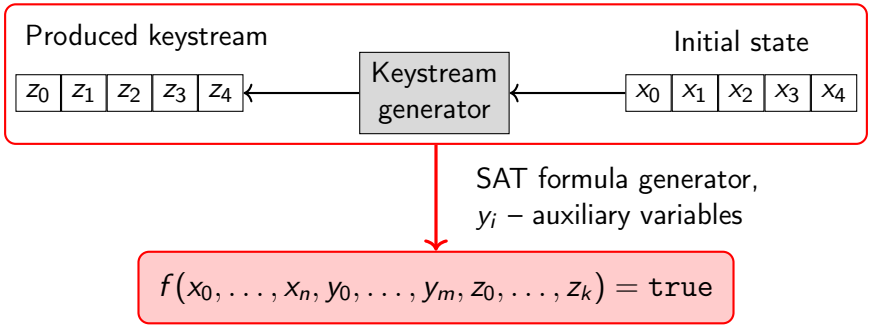
Example of a keystream generator: Trivium-64



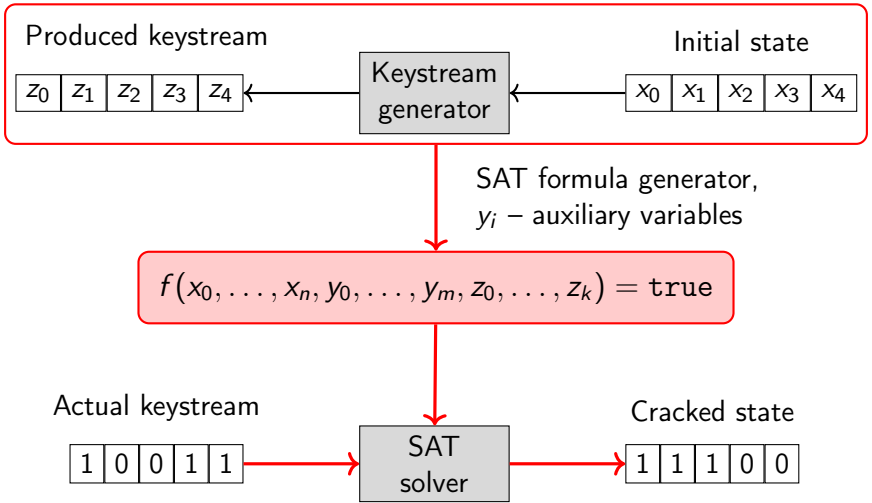
Algebraic cryptoanalysis



Algebraic cryptoanalysis



Algebraic cryptoanalysis



Guess-and-determine attacks

Standard way to solve SAT problems

- Take the formula
- Pass it to the SAT solver

Guess-and-determine attacks

Standard way to solve SAT problems

- Take the formula
- Pass it to the SAT solver

A possible alternative when solving hard SAT problems

- Choose a subset B of the formula's variables – the guessed bit set
- Iterate over all $2^{|B|}$ combinations of their values
- For each combination:
  - Take the formula, substitute these variables with their values
  - Pass it to the SAT solver
  - If solution found, terminate

Guess-and-determine attacks

Standard way to solve SAT problems

- Take the formula
- Pass it to the SAT solver

A possible alternative when solving hard SAT problems

- Choose a subset B of the formula's variables – the guessed bit set
- Iterate over all $2^{|B|}$ combinations of their values
- For each combination:
  - Take the formula, substitute these variables with their values
  - Pass it to the SAT solver
  - If solution found, terminate
- Sometimes this is faster. In cryptanalysis, it happens quite often

Attack time of a guess-and-determine attack

Several definitions possible. We use the following:

- Assume the keystream is infinite
- Set a time limit T for an attempt to solve one piece
  - Found a solution within $T \rightarrow$ congratulations!
  - Did not manage to find $\rightarrow$ continue with the next piece
- Let p be the (very small) probability that we find a solution:
  - Expected time of an attack: T/p
  - Time with 95% of confidence: $\approx 3T/p$

Attack time of a guess-and-determine attack

Several definitions possible. We use the following:

- Assume the keystream is infinite
- Set a time limit T for an attempt to solve one piece
  - Found a solution within $T \rightarrow$ congratulations!
  - Did not manage to find $\rightarrow$ continue with the next piece
- Let p be the (very small) probability that we find a solution:
  - Expected time of an attack: T/p
  - Time with 95% of confidence: $\approx 3T/p$

What is a good time of an attack?

- Any non-trivial result is important
- Example: "SHA-1 collisions now 2^{52}"
- A hint of a weakness $\rightarrow$ move to non-compromised ciphers until too late!

How to measure the attack time

Direct measurement?

- Well, possible, but it will take way too long

How to measure the attack time

Direct measurement?

- Well, possible, but it will take way too long

Clever indirect measurement

- A Monte-Carlo technique

How to measure the attack time

Direct measurement?

- Well, possible, but it will take way too long

Clever indirect measurement

- A Monte-Carlo technique
  - Generate a random initial state

How to measure the attack time

Direct measurement?

- Well, possible, but it will take way too long

Clever indirect measurement

- A Monte-Carlo technique
  - Generate a random initial state
  - Compute the keystream of the needed length

How to measure the attack time

Direct measurement?

- Well, possible, but it will take way too long

Clever indirect measurement

- A Monte-Carlo technique
  - Generate a random initial state
  - Compute the keystream of the needed length
  - Submit it as an input to the solver. We know that the solution exists!

How to measure the attack time

Direct measurement?

- Well, possible, but it will take way too long

Clever indirect measurement

- A Monte-Carlo technique
  - Generate a random initial state
  - Compute the keystream of the needed length
  - Submit it as an input to the solver. We know that the solution exists!
  - Set the time limit equal to $T/2^{|B|}$
  - The solver may find the solution, or may fail to meet the time limit

How to measure the attack time

Direct measurement?

- Well, possible, but it will take way too long

Clever indirect measurement

- A Monte-Carlo technique
  - Generate a random initial state
  - Compute the keystream of the needed length
  - Submit it as an input to the solver. We know that the solution exists!
  - Set the time limit equal to $T/2^{|B|}$
  - The solver may find the solution, or may fail to meet the time limit
- N measurements, N_+ successes $\rightarrow$ the attack efficiency is $\approx \frac{N}{N_+} \cdot T$

How to measure the attack time

Direct measurement?

- Well, possible, but it will take way too long

Clever indirect measurement

- A Monte-Carlo technique
  - Generate a random initial state
  - Compute the keystream of the needed length
  - Submit it as an input to the solver. We know that the solution exists!
  - Set the time limit equal to $T/2^{|B|}$
  - The solver may find the solution, or may fail to meet the time limit
- N measurements, N_+ successes $\rightarrow$ the attack efficiency is $\approx \frac{N}{N_+} \cdot T$
- Estimation of the attack time just got $2^{|B|}$ times faster!

Evolutionary algorithms for attack construction

Problem definition

- Find the guessed bit set B with the smallest estimated attack time

Evolutionary algorithms for attack construction

Problem definition

- Find the guessed bit set B with the smallest estimated attack time

Settings

- Individual = a bit mask, where one-bits define the guessed bit set
- Fitness = the attack time as above

Evolutionary algorithms for attack construction

Problem definition

- Find the guessed bit set B with the smallest estimated attack time

Settings

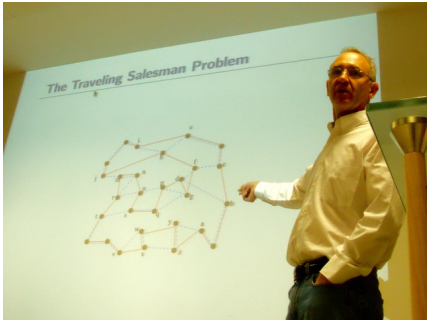
- Individual = a bit mask, where one-bits define the guessed bit set
- Fitness = the attack time as above

Existing techniques

- Local search, simulated annealing, tabu search, $(\mu + \lambda)$-style EAs
- Features: stochastic fitness, non-instant evaluation

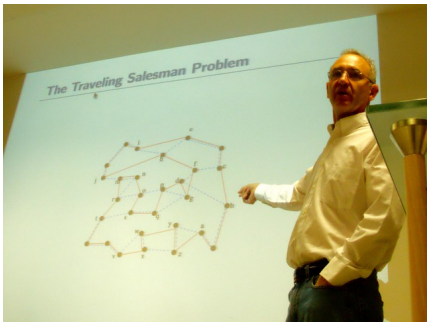
Why not using Gray-Box Optimization?

Why not using Gray-Box Optimization?

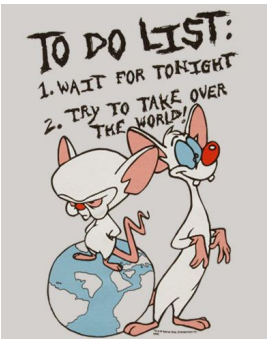


Darrell in da house!1

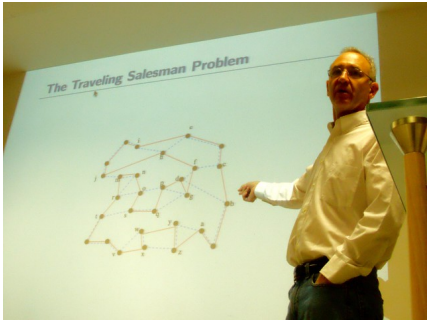
Why not using Gray-Box Optimization?



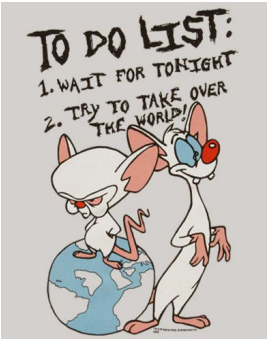
Darrell in da house!1



Why not using Gray-Box Optimization?



Darrell in da house!1



Quick answer: the crypto-nature of the problem implies the enormous number of non-zero Walsh coefficients!

Fitness comparison by statistical testing

Monte-Carlo fitness $\rightarrow$ multiple measurements

- Existing approaches in the domain: fixed number of evaluations (≈ 500)
- **Proposed**: check whether more measurements are needed using statistical testing

Fitness comparison by statistical testing

Monte-Carlo fitness $\rightarrow$ multiple measurements

- Existing approaches in the domain: fixed number of evaluations (≈ 500)
- **Proposed**: check whether more measurements are needed using statistical testing

Domain-related special features

- Fitness is evaluated on a distributed cluster
- Inefficient to do measurements one by one $\rightarrow$ use batches of ≈ 100 measurements

Fitness comparison by statistical testing

Monte-Carlo fitness $\rightarrow$ multiple measurements

- Existing approaches in the domain: fixed number of evaluations (≈ 500)
- **Proposed**: check whether more measurements are needed using statistical testing

Domain-related special features

- Fitness is evaluated on a distributed cluster
- Inefficient to do measurements one by one $\rightarrow$ use batches of ≈ 100 measurements

Statistical tests employed: significant difference $\rightarrow$ save computations

- Wilcoxon rank sum test
- Barnard's test (a simple test to compare two variables with two outcomes)

Fitness comparison by statistical testing

Monte-Carlo fitness → multiple measurements

- Existing approaches in the domain: fixed number of evaluations (≈ 500)
- **Proposed**: check whether more measurements are needed using statistical testing

Domain-related special features

- Fitness is evaluated on a distributed cluster
- Inefficient to do measurements one by one → use batches of ≈ 100 measurements

Statistical tests employed: significant difference → save computations

- Wilcoxon rank sum test
- Barnard's test (a simple test to compare two variables with two outcomes)
- p-values, multiple comparisons etc. – Statistics is a lie, but it is the lesser evil

Experiments

- Simple GA, population size $N = 10$, five experiments, 12 wall-clock hours each
- ROKK SAT solver, time limit for each run is 10 seconds
- Time limit for the final attack time is further refined

Cipher	$\lvert B \rvert$	Time limit	Attack time	#individuals w/ stats	#individuals w/o stats
A5/1	35	0.278	$2.19 \cdot 10^{12}$	1471	341
Bivium	28	2.715	$1.15 \cdot 10^{12}$	3616	2439
Trivium-64	21	2.373	$3.23 \cdot 10^{7}$	3398	1323
Trivium-96	35	2.485	$1.24 \cdot 10^{12}$	2494	1299

Assessment of statistical tests

Cipher	Wilcoxon only	Barnard only	Both	None
A5/1	215	146	1182	5812
Bivium	3786	946	9381	3974
Trivium-64	1943	560	5951	8476
Trivium-96	738	318	3322	8092

Conclusion

- An interesting application of evolutionary algorithms to serious cryptanalysis
- A few world records have been broken (for simplified ciphers however)
- Using statistical testing when comparing Monte-Carlo fitnesses was helpful
  - The number of tested individuals increased by a factor from $1.5\times$ to $4.3\times$
- The methodology is still young $\rightarrow$ more to come!

Conclusion

- An interesting application of evolutionary algorithms to serious cryptanalysis
- A few world records have been broken (for simplified ciphers however)
- Using statistical testing when comparing Monte-Carlo fitnesses was helpful
  - The number of tested individuals increased by a factor from $1.5\times$ to $4.3\times$
- The methodology is still young $\rightarrow$ more to come!

Thanks for listening!