Parallel Logical Cryptanalysis of the Generator A5/1 in BNB-Grid System

Alexander Semenov[1], Oleg Zaikin[1], Dmitry Bespalov[1], and Mikhail Posypkin[2]

[1] Institute for System Dynamics and Control Theory
Siberian Branch of Russian Academy of Sciences,
Lermontov str. 134, 664033 Irkutsk, Russia
[2] Institute for Systems Analysis of Russian Academy of Sciences,
Pr. 60-letiya Oktyabrya 9, 117312 Moscow, Russia
biclop@rambler.ru, oleg.zaikin@icc.ru, bespalov@altrixsoft.com,
posypkin@isa.ru

Abstract. In logical cryptanalysis a problem of search of a secret key of a cryptographic system is formulated as a SAT problem, i.e. problem of search of a satisfying assignment for some CNF. In this paper we consider natural strategies for parallelization of these SAT problems. We apply coarse-grained approach which makes it possible to use distributed computing environments with slow interconnect. The main practical result of this paper is successful logical cryptanalysis of keystream generator A5/1 in BNB-Grid system.

Keywords: Logical cryptanalysis, SAT, stream ciphers, A5/1, coarse-grained parallelization, Grid

1 Introduction

The idea of using SAT-solvers for the problems of cryptanalysis was first introduced in [1]. Term "logical cryptanalysis" itself was proposed in [2]. Examples of successful application of SAT approach to cryptanalysis of some weak stream ciphers are shown in [3], [4], [5]. However, to the best of our knowledge there are no results of successful use of parallel algorithms in logical cryptanalysis of widely used stream encryption systems.

In this paper we consider the problem of parallel logical cryptanalysis of stream the generator A5/1 which is used to encrypt GSM-traffic. According to basic principles of logical cryptanalysis we reduce a problem of cryptanalysis of the generator A5/1 to a SAT-problem. Then we use special technique of parallelization to solve the SAT-problem obtained. This technique exploits peculiarities of the original SAT problem to decompose it into a large set of independent sub-problems. This approach was implemented in the BNB-Grid framework [6] specially developed for solving large scale problems in a heterogeneous distributed systems. Using this approach we were able to successfully perform the cryptanalysis of the generator A5/1 in reasonable time.

We also experimentally proved that the same keystream of arbitrary length can be generated from different secret keys and identified all such keys for the particular 144-bit fragment of keystream.

2 Reducing Cryptanalysis of the Generator A5/1 to SAT

In this section we give general formulation of cryptanalysis problem for keystream generators and describe the procedure of reduction of this problem to SAT. Let f_n,

$$f_n : \{0,1\}^n \to \{0,1\}^*,$$

be a discrete function defined by the algorithm of the generator, that produces a keystream from a secret key $x \in \{0,1\}^n$. We consider the problem of cryptanalysis of keystream generator on the basis of a known keystream. The problem is to find the secret key using some fragment of a keystream and a known algorithm of its generation. It is easy to see that this problem is equivalent to the problem of inversion of function f_n, i.e. the problem of finding such $x \in \{0,1\}^n$ that $f_n(x) = y$ if $y \in range\, f_n$ and an algorithm of computation of f_n are known.

The first step of logical cryptanalysis consists in building a conjunctive normal form (CNF) encoding an algorithm of keystream generator. To obtain this CNF we use Tseitin transformations which were proposed by G.S. Tseitin in 1968 in [7]. In these transformations original function is usually represented by a Boolean circuit over an arbitrary complete basis, for example $\{\&, \neg\}$.

Let f_n be a discrete function defined by an algorithm of generator. We will consider f_n as a function of Boolean variables from the set $X = \{x_1, \ldots, x_n\}$. Let $S(f_n)$ be Boolean circuit which represents f_n over $\{\&, \neg\}$. Each variable from X corresponds to one of n inputs of $S(f_n)$. For each logic gate G some new auxiliary variable $v(G)$ is introduced. Every AND-gate G is encoded by CNF-representation of Boolean function $v(G) \leftrightarrow u\&w$. Every NOT-gate G is encoded by CNF-representation of Boolean function $v(G) \leftrightarrow \neg u$. Here u and w are variables corresponding to inputs of G. CNF encoding $S(f_n)$ is

$$\mathop{\&}_{G \in S(f_n)} C(G),$$

where $C(G)$ is CNF encoding gate G. Then

$$\left(\mathop{\&}_{G \in S(f_n)} C(G)\right) \cdot y_1^{\sigma_1} \cdot \ldots \cdot y_m^{\sigma_m}$$

is CNF encoding the inversion problem of the function f_n in point $y = (\sigma_1, \ldots, \sigma_m)$. Here

$$y^\sigma = \begin{cases} \bar{y}, \text{if } \sigma = 0 \\ y, \text{if } \sigma = 1 \end{cases}$$

and $y_1, \ldots, y_m$ are variables corresponding to outputs of $S(f_n)$.

Quite often a structure of an algorithm calculating a cryptographic function allows us to write a system of Boolean equations which encodes this algorithm

directly without constructing Boolean circuit. Using Tseitin transformations we can make a transition from the obtained system to one equation in the form "CNF=1".

Next, we consider the keystream generator A5/1 used to encrypt traffic in GSM networks. Algorithm of this generator became publicly available in 1999 after reverse engineering performed by M. Briceno. A lot of attacks on this cipher are described, however it is still actively used. The most recent attacks used technique of rainbow tables [8], however this approach can not guarantee the success in 100% of cases. Further we propose a new approach to cryptanalysis of the generator A5/1 that uses parallel algorithms of solving SAT problems.

The description of the generator A5/1 (see Fig. 1) was taken from the paper [9]. According to [9] the generator A5/1 contains three linear feedback shift registers (LFSR, see, e.g., [10]), given by the following connection polynomials: LFSR 1: $X^{19} + X^{18} + X^{17} + X^{14} + 1$; LFSR 2: $X^{22} + X^{21} + 1$; LFSR 3: $X^{23} + X^{22} + X^{21} + X^8 + 1$.

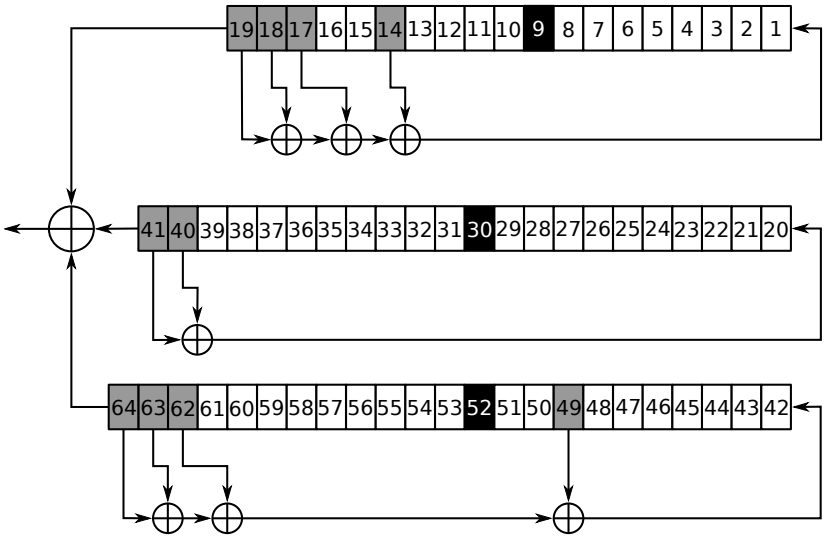

Fig. 1. Scheme of the generator A5/1.

The secret key of the generator A5/1 is the initial contents of LFSRs 1–3 (64 bits). In each unit of time $\tau \in \{1, 2, \dots\}$ ($\tau = 0$ is reserved for the initial state) two or three registers are shifted. The register with number r, $r \in \{1, 2, 3\}$, is shifted if $\chi_r^\tau\,(b_1^\tau, b_2^\tau, b_3^\tau) = 1$, and is not shifted if $\chi_r^\tau\,(b_1^\tau, b_2^\tau, b_3^\tau) = 0$. By $b_1^\tau,\ b_2^\tau,\ b_3^\tau$ we denote here the values of the clocking bits at the current unit of time. The clocking bits are 9-th, 30-th and 52-nd. Corresponding cells in Fig. 1 are black.

The function $\chi_r^\tau(\cdot)$ is defined as follows

$$\chi_r^\tau(b_1^\tau, b_2^\tau, b_3^\tau) = \begin{cases} 1, & b_r^\tau = \text{majority}(b_1^\tau, b_2^\tau, b_3^\tau) \\ 0, & b_r^\tau \neq \text{majority}(b_1^\tau, b_2^\tau, b_3^\tau) \end{cases}$$

where $\text{majority}(A, B, C) = A \cdot B \vee A \cdot C \vee B \cdot C$.

In each unit of time the values in the leftmost cells of the registers are added mod 2, the resulting bit is the bit of the keystream.

Thus, we can see that the generator A5/1 updates the content of each of the registers' cells as a result of conditional shifts: if the shift does not occur, then a new configuration of a register does not differ from the old one, otherwise values of all cells of the register are updated. Hence with each cell at each unit of time we can associate a Boolean equation linking a new state of the cell with the previous one. Let variables $x_1, \ldots, x_{64}$ encode the secret key of the generator A5/1 (x_i corresponds to cell with number $i \in \{1, \ldots, 64\}$). By $x_1^1, \ldots, x_{64}^1$ we denote variables encoding cells' state in the moment of time $\tau = 1$. System of equations which links these two sets of variables is:

$$\begin{cases} \left(x_1^1 \leftrightarrow x_1 \cdot \overline{\chi_1^1} \vee (\oplus_{i \in I} x_i) \cdot \chi_1^1\right) = 1 \\ \left(x_2^1 \leftrightarrow x_2 \cdot \overline{\chi_1^1} \vee x_1 \cdot \chi_1^1\right) = 1 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ \left(x_{20}^1 \leftrightarrow x_{20} \cdot \overline{\chi_2^1} \vee (\oplus_{j \in J} x_j) \cdot \chi_2^1\right) = 1 \\ \left(x_{21}^1 \leftrightarrow x_{21} \cdot \overline{\chi_2^1} \vee x_{20} \cdot \chi_2^1\right) = 1 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ \left(x_{42}^1 \leftrightarrow x_{42} \cdot \overline{\chi_3^1} \vee (\oplus_{k \in K} x_k) \cdot \chi_3^1\right) = 1 \\ \left(x_{43}^1 \leftrightarrow x_{43} \cdot \overline{\chi_3^1} \vee x_{42} \cdot \chi_3^1\right) = 1 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ \left(x_{64}^1 \leftrightarrow x_{64} \cdot \overline{\chi_3^1} \vee x_{63} \cdot \chi_3^1\right) = 1 \\ \left(g^1 \leftrightarrow x_{19}^1 \oplus x_{41}^1 \oplus x_{64}^1\right) = 1 \end{cases} \quad (1)$$

where $I = \{14, 17, 18, 19\}$, $J = \{40, 41\}$, $K = \{49, 62, 63, 64\}$ and g^1 is the first bit of keystream.

Let $g^1, \ldots, g^L$ be the first L bits of the keystream of the generator A5/1. To the each bit $g^i, i \in \{1, \ldots, L\}$ we associate a system of the form (1). To find the secret key it is sufficient to find a common solution of these systems. The problem of finding of this common solution can be reduced by the means of Tseitin transformations to the problem of finding a satisfying assignment of a satisfiable CNF.

3 Coarse-Grained Parallelization of the Problem of Logical Cryptanalysis of the Generator A5/1

In this section we describe a technology for solving SAT problems in distributed computing systems (hereinafter DCS). Such systems consist of sets of computing nodes connected by a communication network. Each node of a DCS has one or several processors. Typical examples of DCS are computing clusters which have become widespread

in recent years. The elementary computational units of modern DCS are cores of processors.

We consider an arbitrary CNF C over the set of Boolean variables $X = \{x_1, \ldots, x_n\}$ and select in the set X some subset

$$X' = \{x_{i_1}, \ldots, x_{i_d}\} \ , \ \{i_1, \ldots, i_d\} \subseteq \{1, \ldots, n\},$$

where $d \in \{1, \ldots, n\}$. We call $X' = \{x_{i_1}, \ldots, x_{i_d}\}$ a decomposition set and d is the power of the decomposition set. To the decomposition set X', $|X'| = d$, we associate the set $Y(X') = \{Y_1, \ldots, Y_K\}$ consisting from $K = 2^d$ different binary vectors of the length d, each of which is a vector of values of the variables $x_{i_1}, \ldots, x_{i_d}$. By $C_j = C|_{Y_j}$, $j = 1, \ldots, K$, we denote the CNF obtained after substitutions of the values from the vectors Y_j to C. A decomposition family generated from the CNF C by the set X', is the set $\Delta_C(X')$, formed by the following CNFs:

$$\Delta_C\left(X'\right) = \{C_1 = C|_{Y_1}, \ldots, C_K = C|_{Y_K}\}.$$

It is not difficult to see that any truth assignment $\alpha \in \{0,1\}^n$ satisfying C ($C|_\alpha = 1$) coincides with some vector $Y^\alpha \in Y(X')$ in the components from X' and coincides with some satisfying assignment of the CNF $C|_{Y^\alpha} \in \Delta_C(X')$ in the remaining components. In this case the CNF C is unsatisfiable if and only if all the CNF in $\Delta_C(X')$ are unsatisfiable. Therefore, the SAT problem for the original CNF C is reduced to K SAT problems for CNFs from the set $\Delta_C(X')$. For processing the set $\Delta_C(X')$ as a parallel task list a DCS can be used.

We use peculiarities of original problem to construct a decomposition set with "good" properties. In logical cryptanalysis problems decomposition set is usually chosen among the subsets of the set of input variables of cryptographic function considered. For the logical cryptanalysis of A5/1 we propose to include into the decomposition set X' the variables encoding the initial states of the cells of registers, starting with the first cells until the cells containing clocking bits inclusive (corresponding cells in the Fig. 2 are dark shaded). Thus, the decomposition set X' consists of 31 variables:

$$X' = \{x_1, \ldots, x_9, x_{20}, \ldots, x_{30}, x_{42}, \ldots, x_{52}\} \tag{2}$$

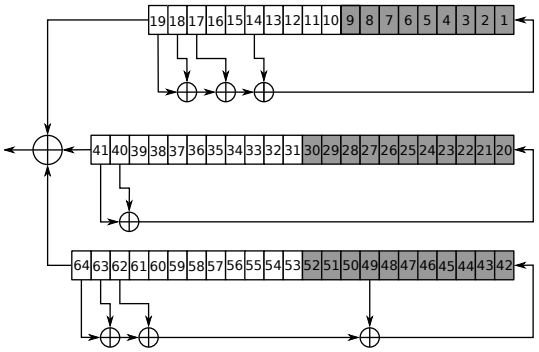

Fig. 2. Scheme of a decomposition set consisting of 31 variables.

This choice is motivated by the following considerations. Assigning values to all variables from X' we determine the exact values of clocking bits for a large number of subsequent states of all three registers. These clocking bits are the most informative because they determine the value of the majority function.

Let C be the CNF encoding the problem of cryptanalysis of the generator A5/1 (see Section 2). By $\Delta_{A5/1}(X')$ we denote the decomposition family generated from the CNF C by the set X' defined by (2). Thus $|\Delta_{A5/1}(X')| = 2^{31}$.

Further we describe a procedure of processing of $\Delta_{A5/1}(X')$ as a parallel task list in DCS. Suppose that the considered DCS has $M < 2^{31}$ computing cores. Let us put the CNFs of the family $\Delta_{A5/1}(X')$ in some order. We call an arbitrary CNF from $\Delta_{A5/1}(X')$ locked if at the current moment of time the SAT problem for it has either been solved or is being solved on some core of the DCS. The other CNFs are called free. We select first M CNFs $C_1, \ldots, C_M$ from the family $\Delta_{A5/1}(X')$. For each of the selected CNFs we solve the SAT problem on a separate core of the DCS. Once some core is released we launch the procedure of solving of the SAT problem for the next free CNF of the family $\Delta_{A5/1}(X')$ on this core. This process continues until a satisfying assignment for some CNF from $\Delta_{A5/1}(X')$ is found, or until the unsatisfiability of all CNFs from $\Delta_{A5/1}(X')$ is proven.

4 Modification of a SAT Solver for Solving the Problems of Logical Cryptanalysis of the Generator A5/1

For solving of SAT problems from the decomposition family $\Delta_{A5/1}(X')$ solver we used modified version of well-known SAT solver MiniSat-C_v1.14.1 [11]. The first stage of the modification consists in changing the decision variable selection procedure (see [12]) implemented in Minisat. Namely, a procedure of assignment of initial activity (different from zero) for those variables in the CNF which correspond to the input variables of the function was added. For the problems of cryptanalysis of generators this method allows to select, on the initial stage of the solving process, the variables corresponding to the secret key as priority variables for decision variable selection procedure. Also some basic constants of the solver were changed. Like most of its analogs Minisat periodically changes the activity of all the variables and clauses in order to increase the priority of selection for variables from the clauses derived in the later steps of the search. Moreover, in 2% of cases the Minisat assigns a value to a variable selected randomly, rather than to the variable with the maximum activity. These heuristics show, on average, good results on a broad set of test examples used in the competitions of SAT solvers. However, for the CNFs encoding problems of cryptanalysis they are, usually, not efficient. In all the experiments described below we use the SAT solver in which periodical lowering of the activity and random selection of variables are prohibited. In total, this simple change led to a substantial increase in efficiency of the SAT solver on cryptographic tests. Unmodified SAT solvers Minisat-C v1.14.1 and Minisat 2.0 did not cope with CNFs from the decomposition family constructed during the logical cryptanalysis of the generator A5/1, even in 10 minutes of work (the computations were interrupted). The modified Minisat-C v1.14.1 solved these problems in less than 0.2 seconds on average.

In the preceding section a general procedure for parallel processing of a list of tasks was described. During this procedure the control process monitors the load of computing cores and sends new tasks to the released cores. In practice, a direct implementation

of this scheme leads to an excessive growth of transfer costs, but provides uniform load of the cores.

The efficiency of a SAT solver in a DCS can be improved by using job batches. Each job batch is a subset of the decomposition family $\Delta_{A5/1}(X')$. Sending batches instead of single CNFs allows to reduce the cost of the transfer. We decompose $\Delta_{A5/1}(X')$ into disjoint sets of job batches. The obtained set of the job batches is considered as a task list where each job batch is a list item. For processing this task list we use the technique described in the previous section.

The fact that a decomposition set is a set of Boolean variables makes the problem of transferring the batches to the cores very simple. Indeed, let X' be decomposition set defined by (2). And let M be the number of computing cores in the DCS. The core with the number $p \in \{1,\ldots,M\}$ we denote by e_p. For the sake of simplicity, assume that $M = 2^k,\ k \in \mathrm{N}_1$, and $k < 31$. If we suppose that all the tasks in the decomposition family $\Delta_{A5/1}(X')$ have approximately equal complexity, then when solving the problem in the DCS each core is going to process approximately the same number of tasks. This means that the decomposition family $\Delta_{A5/1}(X')$ can be partitioned into 2^k subfamilies of equal power and each subfamily can be further processed entirely on the corresponding core. For this purpose select in X' some subset X'_k of power k (X'_k can be formed, for example, by the first k variables from X'). The description of the job batch for a particular $e_p, p \in \left\{1,\ldots,2^k\right\}$, is a binary vector α_p of the length k, formed by the values of variables from X'_k. Next, for each $e_p, p = 1,\ldots,2^k$, we consider the set Λ_p, consisting of 2^{31-k} different vectors of the length 31 of the form $(\alpha_p|\beta)$, where β takes all 2^{31-k} possible values from the set $\{0,1\}^{31-k}$.

Each core $e_p, p \in \left\{1,\ldots,2^k\right\}$, receives its job batch from the control process as a vector α_p which is used for constructing the set Λ_p. A subfamily of the family $\Delta_{A5/1}(X')$ processed by e_p is obtained as a result of substituting vectors from Λ_p to CNF C which encodes the problem of cryptanalysis of the generator A5/1.

5 Implementation of Parallel Logical Cryptanalysis of the Generator A5/1 in BNB-Grid System

We used the results of computational experiments to determine an approximate time of logical cryptanalysis of the generator A5/1 on the "Chebyshev" cluster [13]. According to these results it would take about one day of "Chebyshev" work even if this cluster is fully dedicated to this task. However, exclusive use of publically available supercomputers is usually not possible. Thus it was clear that for a successful solving of cryptanalysis of the generator A5/1 we would need to combine computational powers of several supercomputers.

We decided to use the BNB-Grid [6] software package aimed at harnessing heterogeneous distributed computing resources (called computing nodes) for solving complex computational problems. This package has already proved its efficiency in solving several large scale optimization problems [6, 14].

BNB-Grid is a generic framework for implementing optimization algorithms on distributed systems. The BNB-Grid tool can harness the consolidated power of computing elements collected from service Grids, desktop Grids and standalone resources to solve hard optimization and combinatorial problems. Adding different types of computational resources is available (e.g., Unicore service Grid, BOINC desktop Grid system).

On the top level of the BNB-Grid the object Computing Space Manager (CS-Manager) is located. It decomposes the original problem into subproblems and dis-

tributes them among the computing nodes. For each computing node there is a corresponding object of the type Computing Element Manager (CE-Manager). CE-Manager provides communication between CS-Manager and the corresponding computing node and also starts and stops applications on this node. After receiving a task from the CS-Manager, CE-Manager transfers it to the corresponding node and starts MPI application BNB-solver which processes the received task on all available cores.

A module for processing SAT problems on a computing cluster was added to the BNB-Solver. The input data of the control object CS-Manager is a description of the original SAT problem in XML format. CS-Manager decomposes SAT problem for the original CNF C and obtains decomposition family. We developed special technique of job batches transfer for BNB-Grid system. Each job batch is a compact description of a subset of the decomposition family. Sending of batches instead of single CNFs allows to reduce the cost of the transfer.

The computations were carried out on a distributed system consisting of four computing clusters (see [15]): MVS-100k (Joint Supercomputer Center of RAS), SKIF-MSU Chebyshev (Moscow State University), cluster of RRC Kurchatov Institute, BlueGene P (Moscow State University).

In our experiments three test problems of cryptanalysis of the generator A5/1 were solved. During the computational experiment the number of simultaneously working computing cores varied from 0 to 5568, averaging approximately 2–3 thousand cores. For each test the computations stopped after finding the first satisfying assignment. The first test problem was solved (the secret key of the generator was found) in 56 hours, the second and the third – in 25 and 122 hours respectively.

The problem of cryptanalysis of the generator A5/1 is also interesting because the same keystream of arbitrary length can be generated from different secret keys. This fact was noted by J. Golic in [16]. We denote these situations as "collisions" using the evident analogy with the corresponding notion from the theory of hash functions. The approach presented in this article allows us to solve the problem of finding all the collisions of the generator A5/1 for a given fragment of a keystream. Using BNB-Grid all collisions for one test problem (we analyzed the first 144 bits of keystream) were found. It turned out that there are only three such collisions (see Table 1). Processing this test problem by the distributed system described above took 16 days.

Table 1. Original key and collisions of the generator A5/1 (in hexadecimal format)

	LFSR 1	**LFSR 2**	**LFSR 3**
	$x_1, \ldots, x_{19}$	$x_{20}, \ldots, x_{41}$	$x_{42}, \ldots, x_{64}$
original key	2C1A7	3D35B9	EEAF2
collision	2C1A7	3E9ADC	EEAF2
collision	2C1A7	3D35B9	77579

References

1. Cook, S.A., Mitchel, D.G.: Finding hard instances of the satisfiability problem: A survey. DIMACS Series in Discrete Mathematics and Theoretical Computer

Science, vol. 35, pp. 1–17 (1997)
2. Massacci, F., Marraro, L.: Logical Cryptanalysis as a SAT Problem. Journal of Automated Reasoning, vol. 24, no. 1–2, pp. 165–203 (2000)
3. McDonald, C., Charnes, C., Pieprzyk, J.: Attacking Bivium with Minisat. Technical Report, 2007/040, ECRYPT Stream Cipher Project (2007)
4. Semenov, A.A., Zaikin, O.S., Bespalov, D.V., Ushakov, A.A.: SAT-approach for cryptanalysis of some stream ciphering systems (in Russian). Journal of Computational Technologies, vol. 13, no. 6, pp. 134–150 (2008)
5. Soos, M., Nohl, K., Castelluccia, C.: Extending SAT Solvers to Cryptographic Problems. LNCS, vol. 5584, pp. 244–257 (2009)
6. Afanasiev, A., Posypkin, M., Sigal, I.: Project BNB-Grid: solving large scale optimization problems in a distributed environment. In: 21 International Symposium on Nuclear Electronics and Computing, pp. 15–19. Dubna (2008)
7. Tseitin, G.S.: On the complexity of derivation in propositional calculus. Studies in Constructive Mathematics and Mathematical Logic, part 2, pp. 115–125 (1968)
8. Guneysu, T., Kasper, T., Novotny, M., Paar, C., Rupp, A.: Cryptanalysis with COPACOBANA. IEEE Transactions on computers, vol. 57, no. 11, pp. 1498–1513 (2008)
9. Biryukov, A., Shamir, A., Wagner, D.: Real Time Cryptanalysis of A5/1 on a PC. In: Fast Software Encryption Workshop, pp. 1–18. Springer-Verlag (2000)
10. Menezes, A., Van Oorschot, P., Vanstone, S.: Handbook of Applied Cryptography. CRC Press (1996)
11. The MiniSat page, http://www.minisat.se
12. Marqeus-Silva, J.P., Sakallah, K.A.: GRASP: A search algorithm for propositional satisfiability. IEEE Trans. on Computers, vol. 48, no. 5, pp. 506–521 (1999)
13. Chebyshev supercomputer, http://parallel.ru/cluster/skif_msu.html
14. Evtushenko, Y., Posypkin, M., Sigal, I.: A framework for parallel large-scale global optimization. Computer Science – Research and Development, vol. 23, no. 3, pp. 211–215 (2009)
15. Top 50 CIS Supercomputers, http://www.supercomputers.ru
16. Golic, J.: Cryptanalysis of Alleged A5 Stream Cipher. In: EUROCRYPT'97, pp. 239–255 (1997)