Synthesizing Shortest Linear Straight-Line Programs over GF(2) for the AES using SAT*

Carsten Fuhs

LuFG Informatik 2, RWTH Aachen University, Germany
fuhs@informatik.rwth-aachen.de

Abstract. Recently, the use of SAT solving has expanded to the area of automatically synthesizing shortest linear straight-line programs from a specification [5,4]. We provide corresponding application benchmarks for parts of the implementation of the S-box of the Advanced Encryption Standard.

1 Introduction

The paper [5] describes the use for SAT solving for automated synthesis of shortest linear straight-line programs over the Galois field of two elements (GF(2)). Such a program of k lines can equivalently be expressed as a Boolean circuit of k XOR gates. The specification for the program then consists of several multivariate linear forms over GF(2).

For the SAT encoding, in addition to the specification one provides the maximum number k of lines of the straight-line program that is supposed to be synthesized. The goal is to find a program with the minimal number of lines that adheres to the specification. The resulting CNF encodes the following decision problem, which was recently shown to be NP-complete [1]:

> For the given specification and a given natural number k, is there a linear straight-line program over GF(2) with at most k lines that fulfills the specification?

If the CNF is satisfiable, then we can reconstruct the corresponding straight-line program of (at most) k lines from any model of the CNF. If however the CNF is unsatisfiable, then we have obtained the proof that no linear straight-line program with k lines adheres to the given specification.

For finding a linear straight-line program over GF(2) that is optimal (wrt. length), we need to discover a number k_{min} such that the CNF for k_{min} is satisfiable, whereas the CNF for $k_{min} - 1$ is not. The reconstructed program of k_{min} lines then is the desired implementation of the specification, and the unsatisfiability proof for $k_{min} - 1$ lines shows the optimality of the implementation. Therefore, when searching for such a number k_{min}, both a satisfiability proof and an unsatisfiability proof are part of the process if one uses reductions to a classic SAT problem. To increase confidence in the optimality proof, it would be interesting to obtain a *certified* unsatisfiability proof for the $k_{min} - 1$ lines CNF.

* Description of benchmark instances submitted to the *SAT Competition 2011*.

2 Benchmark Instances

In [2], Boyar and Peralta describe a decomposition of the S-box for the symmetric cipher *AES (Advanced Encryption Standard)*. In particular, they provide specifications for two linear straight-line programs which (together with a third program that also includes non-linear operations) can be used to obtain an implementation of this S-box. They call these specifications the *top matrix* and the *bottom matrix*, respectively.

Our benchmark suite contains the encodings of the top matrix for program lengths 20 to 30 and of the bottom matrix for program lengths 12 to 35. They were created using our implementation for [5], which makes use of the SAT framework of the verification environment AProVE [6] and the Tseitin implementation of SAT4J [7].

For the top matrix, satisfiability for the threshold value $k_{min} = 23$ was shown using e.g. the SAT solver MiniSAT [3], as reported in [5]. The unsatisfiability proof for $k_{min} - 1 = 22$ was only achieved later by CryptoMiniSAT [8] (cf. [4]).

The corresponding value of k_{min} for the bottom matrix is currently unknown. In [2], Boyar and Peralta report that they have found linear straight-line programs of 30 lines for the bottom matrix using a heuristic approach. Thus, it would be quite interesting to check whether the CNF for $k = 29$ is unsatisfiable (which would imply optimality of the solutions of [2]). So far, we have obtained a satisfiability proof for $k = 32$ and an unsatisfiability proof for $k = 13$. Thus, especially the instances with $13 < k < 32$ provide an interesting application challenge for the state of the art in SAT solving.

3 Conclusion

Over the last years, the use of SAT solving as a back-end for solving NP-complete problems has expanded to more and more areas of application. This is exemplified by the recent use of SAT solving for automatic synthesis of shortest linear straight-line programs over GF(2) [5,4] (or, equivalently, XOR circuits with minimal number of gates). Prior to this complete approach based on SAT solving, only incomplete heuristics [2] were considered feasible for this application. The resulting SAT instances pose a notable challenge for modern SAT solvers.

References

1. J. Boyar, P. Matthews, and R. Peralta. On the shortest linear straight-line program for computing linear forms. In *Proc. Mathematical Foundations of Computer Science (MFCS '08)*, volume 5162 of *LNCS*, pages 168–179, 2008.
2. J. Boyar and R. Peralta. A new combinational logic minimization technique with applications to cryptology. In *Proc. International Symposium on Experimental Algorithms (SEA '10)*, volume 6049 of *LNCS*, pages 178–189, 2010.
3. N. Eén and N. Sörensson. An extensible SAT-solver. In *Proc. Theory and Applications of Satisfiability Testing (SAT '03)*, volume 2919 of *LNCS*, pages 502–518, 2004.

4. C. Fuhs and P. Schneider-Kamp. Optimizing the AES S-box using SAT. In *Proc. International Workshop on Implementation of Logics (IWIL '10)*, 2010.
5. C. Fuhs and P. Schneider-Kamp. Synthesizing shortest linear straight-line programs over GF(2) using SAT. In *Proc. Theory and Applications of Satisfiability Testing (SAT '10)*, volume 6175 of *LNCS*, pages 71–84, 2010.
6. J. Giesl, P. Schneider-Kamp, and R. Thiemann. AProVE 1.2: Automatic termination proofs in the dependency pair framework. In *Proc. International Joint Conference on Automated Reasoning (IJCAR '06)*, volume 4130 of *LNAI*, pages 281–286, 2006.
7. D. Le Berre and A. Parrain. The SAT4J library, release 2.2. *Journal on Satisfiability, Boolean Modelling and Computation (JSAT)*, 7:59–64, 2010.
8. M. Soos, K. Nohl, and C. Castelluccia. Extending SAT solvers to cryptographic problems. In *Proc. Theory and Applications of Satisfiability Testing (SAT '09)*, volume 5584 of *LNCS*, pages 244–257, 2009.