Mixed Integer Programming Models for Finite Automaton and Its Application to Additive Differential Patterns of Exclusive-Or

Siwei Sun, Lei Hu, Peng Wang, Meiqin Wang, Danping Shi, Xiaoshuang Ma, Qianqian Yang, Kai Fu

State Key Laboratory of Information Security, Institute of Information Engineering,
Chinese Academy of Sciences, Beijing 100093, China
Data Assurance and Communication Security Research Center,
Chinese Academy of Sciences, Beijing 100093, China
Key Laboratory of Cryptologic Technology and Information Security,
Ministry of Education, Shandong University, Jinan 250100, China
{sunsiwei,hulei,shidanping,maxiaoshuang,yangqianqian}@iie.ac.cn

Abstract. Inspired by Fu *et al.* work [12] on modeling the exclusive-or differential property of the modulo addition as an mixed-integer programming problem, we propose a method with which any finite automaton can be formulated as an mixed-integer programming model. Using this method, we show how to construct a mixed integer programming model whose feasible region is the set of all differential patterns (α, β, γ)'s, such that $\mathrm{adp}^{\oplus}(\alpha, \beta \to \gamma) = \mathrm{Pr}_{x,y}[((x+\alpha) \oplus (y+\beta)) - (x \oplus y) = \gamma] > 0$. We expect that this may be useful in automatic differential analysis with additive difference.

Keywords: Finite automaton, ARX cipher, Modulo addition, Exclusive-or, Additive differential, Integer programming, Automatic cryptanalysis

1 Introduction

An increasing number of symmetric-key cryptographic primitives are constructed with the operations addition modulo 2^n, which are very simple and efficient in software implementation while provide both diffusion and nonlinearity. Examples include the block cipher SPECK [22], the stream cipher Salsa20 [3], as well as the SHA-3 finalists BLAKE [13]. The differential cryptanalysis is one of the best attacks on the ARX constructions which draws great attention of the cryptographic community. Many attacks based on differential analysis have been conducted on ARX ciphers [16, 23, 8, 21, 26, 9, 10, 1, 4].

Without the support of computer-aided (partially) automatic tools, the differential or linear cryptanalysis is an considerably tedious and error-prone process. Hence, there has been a dramatic increase in the developments of automatic methods for automatic differential or (linear) analysis [16, 17, 23, 2, 11, 6, 8, 19, 21, 25, 24, 27, 15, 20, 5, 12]. The mixed-integer programming based method [20, 25, 24, 12] is getting more and more attention as it is easy to implement and highly automatic.

In the MIP based automatic differential cryptanalysis, the most important step is to construct an MIP model such that the feasible region of the model corresponds to the set of all differential (or linear) characteristics of the target cipher. In this paper, we show how to construct a mixed integer programming model whose feasible region is the set of all differential patterns (α, β, γ)'s, such that $\mathrm{adp}^{\oplus}(\alpha, \beta \to \gamma) = \mathrm{Pr}_{x,y}[((x+\alpha) \oplus (y+\beta)) - (x \oplus y) = \gamma] > 0$. Note that almost exclusively all existing differential attacks adopt the *exclusive-or difference*. We expect the work of this paper may be useful in automatic differential analysis with *additive differences*.

Contribution On the theoretical side, we show that any finite automaton (FA) can be modeled by the MIP technique, and therefore any language can be determined by an regular expression can be also modeled by the MIP technique. We think the technique presented in this paper has the potential to be used in modeling those cryptographic problems related to finite automaton. For example, an additive difference patterns $(\alpha, \beta) \to \gamma$ for the exclusive-or operation is possible (i.e. its probability is greater than 0) if and only if a sequence $x_0 \cdots x_{n-1}$ derived from the pattern $(\alpha, \beta) \to \gamma$ is in the set described by a regular expression [18]. It is well known that we can construct an FA such that an input sequence $x_0 \cdots x_{n-1}$ is matched by the regular expression if and only if it is accepted by the FA. Consequently, we can use the technique presented in this paper to modeling this FA and construct a mixed integer programming model whose feasible region is the set of all differential patterns (α, β, γ)'s, such that $\mathrm{adp}^{\oplus}(\alpha, \beta \to \gamma) = \mathrm{Pr}_{x,y}[((x+\alpha) \oplus (y+\beta)) - (x \oplus y) = \gamma] > 0$; On the practical side, we have implemented the technique presented in this paper and integrated it into a framework for automatic differential and linear cryptanalysis.

2 Notations

In this section, we introduce the notations which will be used in the following sections.

2.1 Finite Automaton

A finite automaton (FA) [14] has a set $\mathcal{S} = \{S_0, \ldots, S_{n-1}\}$ of finitely many states, and $\mathcal{S}_A \subseteq \mathcal{S}$ is the set of accepting states of the FA. The input of an FA is a string $x_0x_1 \cdots x_{m-1}$. The finite automaton starts in state S_0, consumes one character of $x_0x_1 \cdots x_{m-1}$ at a time (from x_0 to x_{m-1}), and the FA takes the transition from one state to another with input x_j according to a transition function $\delta : S_i \mapsto \delta(S_i, x_j) \in \mathcal{S}$.

A finite automaton (FA) accepts a string if and only if, starting in S_0, the sequence of characters in the string takes the FA through a series of transitions that leaves it in an accepting state when the entire string has been consumed.

An FA can be represented by a transition diagram, and an example is shown in Fig. 1. Given a string $x_0x_1x_2x_3x_4 = 21256$. The FA starts at state init. After consume the character $x_0 = 2$, it goes to state 1. Then the FA consume the character $x_1 = 1$, and leaves its state unchanged. The state is still unchanged after receiving the character $x_2 = 2$. The FA jumps to state init upon receiving the character $x_3 = 5$. The last character of the string $x_4 = 6$ make the FA stop at state init, an accepting state marked by two circles.

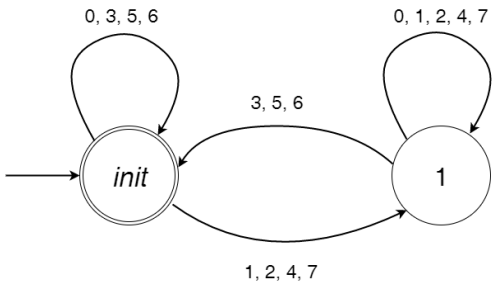

Fig. 1: A finite automaton (used to recognize whether an additive difference pattern with respect to XOR is valid [18])

2.2 Regular Expression

In theoretical computer science and formal language theory, a regular expression is a sequence of characters that define a search pattern, mainly for use in pattern matching with strings. Here, we will not introduce the technical details of a regular expression. But only list some notations which will be used in the following. We refer the reader to [7, 14] for more information.

- *Alternation* The alternation, or union of n characters, $x_0, \cdots, x_{n-1}$, denoted $x_0|x_1|\cdots|x_{n-1}$, is a single character $x \in \{x_0, \cdots, x_{n-1}\}$. For example the single character 3 or 7 is matched by the regular expression 3|5|7, while 8 is not matched by this regular expression.
- *Concatenation* The concatenation of two regular expressions $\mathcal{R}_0$ and $\mathcal{R}_1$, denoted $\mathcal{R}_0\mathcal{R}_1$, matches all strings formed by appending a string matched by $\mathcal{R}_1$ onto one matched by $\mathcal{R}_0$. For example, 98 is matched by the regular expression $(1|2|3|\cdots|9)(0|8)$.
- *Closure* The Kleene closure of a character x, denoted by x^*, matches all strings formed by repeating x zero or more times. For example *aa* and *aaaa* are matched by the regular expression a^*. The strings *bbbc*, *aabbbbbd* are matched by the regular expression $a^*b^*(a|b|c|d)$.

Let $\mathcal{X}_\mathcal{R}$ be the set of all strings which are matched by the regular expression $\mathcal{R}$. It is proved that *we can construct a finite automaton* $FA_\mathcal{R}$ *such that a string is accepted by the finite automaton* $FA_\mathcal{R}$ *if and only if the string belongs to* $\mathcal{X}_\mathcal{R}$ [14].

3 Mixed Integer Programming Model for Finite Automaton

We show how to construct an MIP model whose feasible region is the set of all sequences $x_0 \cdots x_{m-1}$ which are accepted by a given finite automaton.

Step 1. Encoding the states and input characters In this step, we determine how many 0-1 variables are required to represent the states in the automaton, and to assign a specific 0-1 sequence to each state. We also need to determine the number of variables needed to encode the input character set, and to assign a specific 0-1 sequence to each character.

Step 2. Tabular Description of the finite automaton Taking the FA presented in Fig. 1 for example, if the current state of the FA is init (encoded as 0), the FA will go to state 1 (encoded as 1) upon receiving the input character 7 (encoded as 111). This behavior of the FA is recorded as the 8th row in Table. 1. Note that we need to enumerate all such transitions of an FA to form its tabular representation.

Step 3. From tabular Description to mixed integer programming model In this step, we construct an MIP model by using the method of [24] such that its feasible solution corresponds the set of all possible transitions $S_i \to S_{i+1}$ of the FA upon receiving the input character x_i.

A concrete example is given in the following section.

4 MIP Models for Additive Differential Patterns over Exclusive-Or

Definition 1. $\mathrm{adp}^{\oplus}(\alpha, \beta \to \gamma) = \Pr_{x,y}[((x + \alpha) \oplus (y + \beta)) - (x \oplus y) = \gamma]$

let integers $\alpha = \alpha_0\alpha_1 \cdots \alpha_{n-1}$, $\beta = \beta_0\beta_1 \cdots \beta_{n-1}$ and $\gamma = \gamma_0\gamma_1 \cdots \gamma_{n-1}$ are represented in binary form. Let $\omega = \omega_0\omega_1 \cdots, \omega_{n-1}$ such that $\omega_i = 4\alpha_i + 2\beta_i + \gamma_i$, $0 \le i \le n-1$. Then it is obvious that $0 \le \omega_i \le 7$, $0 \le i \le n-1$.

Theorem 1. *[18] The additive differential probability* $\mathrm{adp}^{\oplus}(\alpha, \beta, \gamma)$ *is nonzero if and only if* $\omega = \omega_0\omega_1 \cdots, \omega_{n-1}$ *is in the language defined by the regular expression* $0^*|([0..7])^*(3|5|6)0^*$. *That is,* $\mathrm{adp}^{\oplus} > 0$ *if and only if* ω *is* $00\ldots0$ *or an octoal string starting with any octoal word followed by* 3 *or* 5 *or* 6 *and ending with zero or more* 0*'s.*

Table 1: The tabular description of the finite automaton presented in Fig. 1.

State Transation	S_i	S_{i+1}	α_i	β_i	γ_i	$input : \omega_i = 4\alpha_i + 2\beta_i + \gamma$
$0 \to 0$	0	0	0	0	0	0
	0	0	0	1	1	3
	0	0	1	0	1	5
	0	0	1	1	0	6
$0 \to 1$	0	1	0	0	1	1
	0	1	0	1	0	2
	0	1	1	0	0	4
	0	1	1	1	1	7
$1 \to 0$	1	0	0	1	1	3
	1	0	1	0	1	5
	1	0	1	1	0	6
$1 \to 1$	1	1	0	0	0	0
	1	1	0	0	1	1
	1	1	0	1	0	2
	1	1	1	0	0	4
	1	1	1	1	1	7

The regular expression $\mathcal{R} = 0^*|([0..7])^*(3|5|6)0^*$ corresponds to the FA presented in Fig. 1 [18], whose tabular description is given in Table. 1.

We use a the tuple $(S_i, \alpha_i, \beta_i, \gamma_i, S_{i+1})$ with 5 variables to denote the transition pattern of the FA. From the Table. 1, there are exactly 16 possible patterns for $(S_i, \alpha_i, \beta_i, \gamma_i, S_{i+1})$, which are listed in the following.

```
(0, 0, 0, 0, 0)  (0, 0, 1, 1, 0)  (0, 1, 0, 1, 0)  (0, 1, 1, 0, 0)
(0, 0, 0, 1, 1)  (0, 0, 1, 0, 1)  (0, 1, 0, 0, 1)  (0, 1, 1, 1, 1)
(1, 0, 1, 1, 0)  (1, 1, 0, 1, 0)  (1, 1, 1, 0, 0)  (1, 0, 0, 0, 1)
(1, 0, 0, 1, 1)  (1, 0, 1, 0, 1)  (1, 1, 0, 0, 1)  (1, 1, 1, 1, 1)
```

$$\begin{cases} -2S_i + \alpha_i + \beta_i + \gamma_i + 2S_{i+1} \geq 0 \\ -\alpha_i + \beta_i - \gamma_i - S_{i+1} \geq -2 \\ \alpha_i - \beta_i - \gamma_i - S_{i+1} \geq -2 \\ -\alpha_i - \beta_i - \gamma_i + S_{i+1} \geq -2 \\ -\alpha_i - \beta_i + \gamma_i - S_{i+1} \geq -2 \\ -\alpha_i + \beta_i + \gamma_i + S_{i+1} \geq 0 \\ \alpha_i - \beta_i + \gamma_i + S_{i+1} \geq 0 \\ \alpha_i + \beta_i - \gamma_i + S_{i+1} \geq 0 \\ S_i + \alpha_i + \beta_i + \gamma_i - S_{i+1} \geq 0 \end{cases} \tag{1}$$

An addition requirement is that the automaton starts at the state encoded as 0 and ends at the state encoded as 1, which equivalent to $S_0 = 0$, and $S_n = 1$.

5 Conclusion and Discussion

In this paper, we show how to construct a mixed integer programming model whose feasible region is the set of all differential patterns (α, β, γ)'s, such that $\mathrm{adp}^{\oplus}(\alpha, \beta \to \gamma) = \Pr_{x,y}[((x+\alpha) \oplus (y+\beta)) - (x \oplus y) = \gamma] > 0$. Construction of such MIP models is the first step towards an MIP based automatic differential analysis with additive differences.

However, the current work cannot be used directly to search for good differential characteristics since we do not know what objective function should be used to find high probability characteristics. A possible heuristic choice of the objective function is to minimize the overall Hamming weight of the difference patterns. The investigation of whether this choice of objective function is effective in practice is left as future work, and we welcome any discussion and cooperation.

References

1. Abed, F., List, E., Lucks, S., Wenzel, J.: Differential cryptanalysis of round-reduced simon and speck. In: Fast Software Encryption - 21st International Workshop, FSE 2014, London, UK, March 3-5, 2014. Revised Selected Papers. pp. 525–545 (2014)
2. Alex Biryukov and Vesselin Velichkov: Automatic search for differential trails in ARX ciphers. In: Topics in Cryptology–CT-RSA 2014, pp. 227–250. Springer (2014)
3. Bernstein, D.J.: The salsa20 family of stream ciphers. In: New Stream Cipher Designs - The eSTREAM Finalists, pp. 84–97 (2008)
4. Biryukov, A., Roy, A., Velichkov, V.: Differential analysis of block ciphers SIMON and SPECK. In: Fast Software Encryption - 21st International Workshop, FSE 2014, London, UK, March 3-5, 2014. Revised Selected Papers. pp. 546–570 (2014)
5. Biryukov, A., Velichkov, V., Le Corre, Y.: Automatic search for the best trails in arx: Application to block cipher speck. In: Fast Software Encryption – FSE 2016. Springer (2016)
6. Brier, E., Khazaei, S., Meier, W., Peyrin, T.: Linearization framework for collision attacks: Application to cubehash and MD6. In: Advances in Cryptology - ASIACRYPT 2009, 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009. Proceedings. pp. 560–577 (2009)
7. By Jeffrey E.F. Friedl: Mastering Regular Expressions. O'Reilly (2006)
8. Cannière, C.D., Rechberger, C.: Finding SHA-1 characteristics: General results and applications. In: Advances in Cryptology - ASIACRYPT 2006, 12th International Conference on the Theory and Application of Cryptology and Information Security, Shanghai, China, December 3-7, 2006, Proceedings. pp. 1–20 (2006)
9. Crowley, P.: Truncated differential cryptanalysis of five rounds of salsa20. IACR Cryptology ePrint Archive 2005, 375 (2005), http://eprint.iacr.org/2005/375

10. Dinur, I.: Improved differential cryptanalysis of round-reduced speck. In: Selected Areas in Cryptography - SAC 2014 - 21st International Conference, Montreal, QC, Canada, August 14-15, 2014, Revised Selected Papers. pp. 147–164 (2014)
11. Dobraunig, C., Eichlseder, M., Mendel, F.: Heuristic tool for linear cryptanalysis with applications to CAESAR candidates. In: Advances in Cryptology - ASIACRYPT 2015 - 21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29 - December 3, 2015, Proceedings, Part II. pp. 490–509 (2015)
12. Fu, K., Wang, M., Guo, Y., Sun, S., Hu, L.: Milp-based automatic search algorithms for differential and linear trails for speck. In: Fast Software Encryption – FSE 2016. Springer (2016)
13. Jean-Philippe Aumasson, Luca Henzen, Willi Meier, Raphael C.-W. Phan: SHA-3 proposal BLAKE (2010), `https://131002.net/blake/blake.pdf`
14. Keith D. Cooper, Linda Torczon: Engineering a Compiler. ELSEVIER (2012)
15. Kölbl, S., Leander, G., Tiessen, T.: Observations on the SIMON block cipher family. In: Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I. pp. 161–185 (2015)
16. Leurent, G.: Construction of differential characteristics in ARX designs application to skein. In: Advances in Cryptology–CRYPTO 2013, pp. 241–258. Springer (2013)
17. Lipmaa, H., Moriai, S.: Efficient algorithms for computing differential properties of addition. In: Fast Software Encryption – FSE 2002. pp. 336–350. Springer (2002)
18. Lipmaa, H., Wallén, J., Dumas, P.: On the additive differential probability of exclusive-or. In: Fast Software Encryption – FSE 2004. pp. 317–331. Springer (2004)
19. Mendel, F., Nad, T., Schläffer, M.: Finding SHA-2 characteristics: Searching through a minefield of contradictions. In: Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings. pp. 288–307 (2011)
20. Mouha, N., Wang, Q., Gu, D., Preneel, B.: Differential and linear cryptanalysis using mixed-integer linear programming. In: Information Security and Cryptology. pp. 57–76. Springer (2012)
21. Nicky Mouha and Bart Preneel: Towards finding optimal differential characteristics for ARX: Application to Salsa20. IACR Cryptology ePrint Archive, Report 2013/328 (2013), `http://eprint.iacr.org/2013/328`
22. Ray Beaulieu, Douglas Shors, Jason Smith, Stefan Treatman-Clark, Bryan Weeks and Louis Wingers: The SIMON and SPECK families of lightweight block ciphers. IACR Cryptology ePrint Archive, Report 2013/404 (2013), `http://eprint.iacr.org/2013/404`
23. Schläffer, M., Oswald, E.: Searching for differential paths in MD4. In: Fast Software Encryption, 13th International Workshop, FSE 2006, Graz, Austria, March 15-17, 2006, Revised Selected Papers. pp. 242–261 (2006)
24. Sun, S., Hu, L., Wang, M., Wang, P., Qiao, K., Ma, X., Shi, D., Song, L.: Automatic enumeration of (related-key) differential and linear characteristics with predefined properties and its applications. IACR Cryptology ePrint Archive 2014, 747 (2014)
25. Sun, S., Hu, L., Wang, P., Qiao, K., Ma, X., Song, L.: Automatic security evaluation and (related-key) differential characteristic search: Application to simon, present, lblock, DES(L) and other bit-oriented block ciphers. In: Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014. Proceedings, Part I. pp. 158–178 (2014)
26. Wang, X., Yu, H.: How to break MD5 and other hash functions. In: Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings. pp. 19–35 (2005)
27. Yao, Y., Zhang, B., Wu, W.: Automatic search for linear trails of the SPECK family. In: Information Security - 18th International Conference, ISC 2015, Trondheim, Norway, September 9-11, 2015, Proceedings. pp. 158–176 (2015)