分类号：TP309　　单位代码：10422
密　级：公开　　学　　号：201311310

山东大学
SHANDONG UNIVERSITY

硕士学位论文

Thesis for Master Degree

论文题目：针对ARX算法基于MILP的差分线性自动化搜索技术与模减差分的SMT模型

MILP-Based Automatic Search Algorithms for Differential and Linear Trails for ARX Cipher and SMT Model for Additive Differential

作者姓名　付凯
培养单位　数学学院
专业名称　信息安全
指导教师　王美琴 教授
合作导师

2016年 5月 24日

原 创 性 声 明

本人郑重声明：所呈交的学位论文，是本人在导师的指导下，独立进行研究所取得的成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的科研成果。对本文的研究作出重要贡献的个人和集体，均已在文中以明确方式标明。本声明的法律责任由本人承担。

论文作者签名：付凯　　日　期：2016.5.24

关于学位论文使用授权的声明

本人完全了解山东大学有关保留、使用学位论文的规定，同意学校保留或向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅；本人授权山东大学可以将本学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或其他复制手段保存论文和汇编本学位论文。

(保密论文在解密后应遵守此规定)

论文作者签名：付凯　导师签名：　日　期：2016.5.24

目录

中文摘要 I

英文摘要 III

第一章 引言 1

§1.1 课题研究背景 1

§1.2 论文组织结构与研究进展 3

§1.2.1 研究进展 3

§1.2.2 论文组织结构 5

第二章 差分与线性路线的自动化搜索算法简介 9

§2.1 差分分析与线性分析 9

§2.2 差分特征的MILP模型 11

§2.3 线性逼近的MILP模型 12

第三章 基于MILP针对ARX加密算法差分特征的自动化搜索技术 15

§3.1 模加操作的异或差分特征 15

§3.2 模加操作差分特征的MILP模型 16

§3.3 ARX加密算法差分特征的MILP模型 18

§3.4 小结 19

第四章 基于MILP针对ARX加密算法线性逼近的自动化搜索技术 21

§4.1 模加操作的线性逼近 21

§4.2 模加操作线性逼近的MILP模型 24

§4.3 ARX加密算法线性逼近的MILP模型 25

§4.4 小结 26

第五章 对Speck算法的攻击 27
§5.1 Speck算法描述与安全性分析 27
§5.1.1 Speck算法描述 27
§5.1.2 Speck安全性分析 27
§5.2 Speck加密算法的差分特征与线性逼近 29
§5.2.1 构建MILP模型 29
§5.2.2 启发式拼接技术 31
§5.2.3 实验结果 32
§5.3 Speck算法的密钥恢复攻击 34
§5.4 小结 34
第六章 有关模减差分的SMT模型的讨论 43
第七章 总结及未来工作 47
参考文献 49
致谢 53
攻读学位期间完成的学术论文目录 55

CONTENTS

Chinese Abstract I

English Abstract III

Chapter 1 Introduction 1

§1.1 Backgroud 1

§1.2 Organization and Contributions 3

§1.2.1 Contributions 3

§1.2.2 Organization 5

Chapter 2 Preliminary 9

§2.1 The Differential and Linear Attack 9

§2.2 The MILP Model for Differential Characterisics 11

§2.3 The MILP Model for Linear Approximations 12

Chapter 3 MILP-Based Search Algorithm for Differential Trails in ARX .. 15

§3.1 The XOR-Differential Characteristic of Modular Addition 15

§3.2 MILP Model for Differential Characteristic of Addition 16

§3.3 MILP Model for Differential Characteristic of ARX Ciphers 18

§3.4 Summary 19

Chapter 4 MILP-Based Search Algorithm for Linear Trails in ARX 21

§4.1 The Linear Approximation of Modular Addition 21

§4.2 MILP Model for Linear Approximation of Modular Addition 24

§4.3 MILP Model for Linear Approximation of ARX Ciphers 25

§4.4 Summary 26

Chapter 5 Application to Speck 27
§5.1 The Description and Security Analysis of Speck 27
§5.1.1 The Description of Speck 27
§5.1.2 The Security Analysis of Speck 27
§5.2 Differential and Linear Trails of Speck 29
§5.2.1 How to Construct MILP Model 29
§5.2.2 The Heuristic Splicing Technoloy 31
§5.2.3 The Experimental Results 32
§5.3 The Key Recovery Attack on Speck 34
§5.4 Summary 34
Chapter 6 The SMT Model for Additive Differential 43
Chapter 7 Conclusion and Future Work 47
References 49
Acknowlegements 53
Research and Published papers 55

针对ARX算法基于MILP的差分线性自动化搜索技术与模减差分的SMT模型

付 凯

(山东大学数学学院, 济南, 250100)

(山东大学密码技术与信息安全教育部重点实验室)

摘要

随着互联网技术的快速发展，信息安全越来越受到重视。密码学也因此逐渐成为一个热门话题。根据研究方向的不同，密码学主要分为对称密钥密码学与公钥密码学。其中，分组密码是对称密钥密码的重要组成部分。分组密码因其结构简单，便于实现，被广泛应用在各个数据加密场景。因此，分组密码一直是密码学界的研究热点。

针对分组密码算法，密码学家提出许多攻击方案。其中，差分分析与线性分析是两种最经典且最有效的破解手段。对于这两种方法来说，最重要的事情莫过于寻找出一条“好”的差分特征或线性逼近。因此，有关自动化搜索差分特征与线性逼近的工作是一个热门研究方向。

近年来，基于混合整数线性规划（MILP）的搜索技术越来越多的应用到分组算法的差分特征与线性逼近的自动化搜索中，且体现了较为显著的优势，对Simon等分组密码算法取得了较好的结果（Simon分组密码算法是美国国家安全局NSA于2013年设计的一组轻量级加密算法，其特点是消耗硬件面积小）。然而，目前还没有基于MILP针对ARX加密算法进行自动化搜索差分特征与线性逼近的方法。

在本文中，我们提出一种基于MILP自动化搜索ARX算法差分特征与线性逼近的新技术。通过研究ARX算法中模加操作的差分与线性特性，并在假设模加操作的输入与轮与轮之间都是相互独立的前提下，我们提出一种用线性不等式来描述模加操作差分特征与线性逼近特性的新方法。利用该方法，我们可以构建完全刻画ARX算法差分线性特性的MILP模型，并用Gurobi（一种可公开获得的MILP求解器）来搜索ARX 算法的差分特征与线性逼近。作为验证，我们将该技术应用到Speck算法的分析中（Speck加

密算法是美国国家安全局NSA于2013年设计的一组轻量级加密算法，其特点是消耗软件资源小）。我们搜索出优于已有结果的差分特征与线性逼近。利用新搜索出的差分特征以及guess-and-determine技术，我们改进了Speck48，Speck64，Speck96与Speck128的差分攻击。值得注意的是，我们的攻击目前就轮数来说是已知的最优攻击结果。

此外，可满足性模理论（SMT）也引起了密码学界的广泛关注。SMT逐渐被应用到密码算法的搜索工作中。我们在本文中讨论了构建模减差分的SMT模型的可能性，并在假设异或操作的输入与轮与轮之间相互独立的条件下给出了一种新的建模思路。

关键字： 自动化搜索; 差分特征; 线性逼近; ARX; Speck; SMT; STP

MILP-Based Automatic Search Algorithms for Differential and Linear Trails for ARX Cipher and SMT Model for Additive Differential

Kai Fu
(School of mathematics, Shandong University, Jinan, 250100)
(Key Lab of Cryptologic Technology and Information Security, Ministry of Education)

ABSTRACT

With the development of internet, the information security has been more and more important. The cryptography has also been a hot topic. There are two research field in cryptography. One is symmetric cryptography and the other is public key cryptography. The block cipher is an important part of symmetric cryptography. Because of simple structure, the block cipher is widely used in data encryption. The research of block cipher has drawn wide attention.

The cryptographers has proposed many method to attack block cipher. The differential attack and linear attack are two most useful method. The most important thing for two method is to find a good differential trail or linear trail. So, how to automatic search the differential and linear trials has drawn wide attention.

In recent years, Mixed Integer Linear Programming (MILP) has been successfully applied in searching for differential characteristics and linear approximations in block ciphers and has produced the significant results for some ciphers such as SIMON (a family of lightweight and hardware-optimized block ciphers designed by NSA) etc. However, in the literature, the MILP-based automatic search algorithm for differential characteristics and linear approximations is still infeasible for block ciphers such as ARX constructions.

In this paper, we propose an MILP-based method for automatic search

for differential characteristics and linear approximations in ARX ciphers. By researching the properties of differential characteristic and linear approximation of modular addition in ARX ciphers, we present a method to describe the differential characteristic and linear approximation with linear inequalities under the assumptions of independent inputs to the modular addition and independent rounds. We use this representation as an input to the publicly available MILP optimizer Gurobi to search for differential characteristics and linear approximations for ARX ciphers. As an illustration, we apply our method to Speck, a family of lightweight and software-optimized block ciphers designed by NSA, which results in the improved differential characteristics and linear approximations compared with the existing ones. Moreover, we provide the improved differential attacks on Speck48, Speck64, Speck96 and Speck128, which are the best attacks on them in terms of the number of rounds.

In addition, Satisfiability Modulo Theories(SMT) is also interest the cryptographers. SMT can also be used to search the differential trails for ARX ciphers. In this paper, we also talk about how to construct the SMT model to search the additive differential trails for ARX ciphers and give a new method to achieve our goal.

Keywords : Automatic Search; Differential Characteristic; Linear Approximation; ARX; Speck SMT; STP

第一章 引言

§1.1 课题研究背景

随着斯诺登“棱镜门”、苹果公司“泄密门”等一连串泄密事件的发生，信息安全受到前所未有的关注。密码学则因其对数据的保护作用成为信息安全的基石。密码学主要分为对称密码学与公钥密码学。对称密码学又分为分组密码，哈希函数，流密码与认证加密算法等。其中，分组密码因其结构简单，便于实现，且加解密效率高，被广泛的应用到各种数据加密场景。因此，分组密码一直是密码学界研究热点。

针对分组加密算法的破解，差分攻击[3]与线性攻击[15]是最早提出且最有效的两种分析手段。这两种方法不仅被应用到分组密码的研究中，还被广泛的应用到对称密码其他领域。对于这两种分析技术，最重要的事情莫过于寻找一条“好”的差分特征或线性逼近。有了“好”的差分特征或线性逼近，我们的攻击可以事半功倍。因此，有关差分特征与线性逼近的自动化搜索的研究工作一直受到密码学家的广泛关注。近几年的研究思路主要有两大主线：在94年欧密会上，Matsui[16]提出了Branch-and-Bound搜索算法，并且搜索出DES分组加密算法的差分特征与线性逼近。自从问世来，Branch-and-Bound算法被广泛应用到分组密码算法差分特征与线性逼近的搜索工作中。它目前仍是最有效最经典的搜索工具之一。另一条研究路线是则是通过自动化搜索最少活跃S盒个数，来提供抵抗差分分析与线性分析的可证明安全上下界。

近年来，基于混合整数线性规划（MILP）的搜索技术越来越多的应用到分组密码算法差分特征与线性逼近的自动化寻找中。我们首先简单介绍下MILP：MILP问题是从线性规划问题演变过来的一系列求解最优解问题。MILP问题旨在在特定条件下求解出目标函数的最优解。MILP问题要求模型中的变量必须为整数，不能存在出现小数的情况。基于MILP的自动化搜索算法研究进展如下：Mouha[18]与Wu[28]等人率先成功将计算最少差分活跃S盒个数转化为MILP问题。这样就可以很容易的用开源或者商业MILP求解器来进行自动化求解，从而得到最少差分活跃S盒个数。该方法在文献[14, 29]中被用来搜寻特定模式的差分特征与线性逼近。此外，

在文献[21, 27]中，通过加入特定的表示方法，该技术可被用来计算比特级的分组加密算法的最小活跃S 盒个数。随着密码学家的不断努力，基于MILP的方法已经发展成一种自动化搜索真实差分特征的通用工具。在文献[22]中，sun等人通过SAGE工具来产生刻画S盒差分分布特性的线性不等式集合，并用这些不等式来构建搜索差分特征（相关密钥差分特征）的MILP模型。然而为了减少运行时间，加快搜索进程，sun等人只选取了不等式集合中的部分不等式。因此，他们构建的MILP模型不能精确描述所有差分（线性）的真实情况。所以，他们的搜索程序是启发式的，而且得到的结果不一定是准确的或最优的。为了克服这一问题，sun等人在文献[24]中提出一种贪心算法。运用该算法，他们可以在[22]中产生的全部不等式集合中挑选出部分不等式子集。这部分不等式可以完全刻画S盒的全部差分分布情况。这样就可以将[22]中启发式搜索算法转变成准确的可实际应用的搜索方法。此外，他们提出的MILP模型也可以通过加入特定条件来进行差分路径与线性壳的搜索。总结一下，Sun等人在文献[22, 24]提出的方法主要适用于包含比特级异或操作，S 盒操作与比特级线性置换的分组加密算法。值得注意的是，尽管一般的线性层可以被转换成比特级的异或操作，但这样会大大增加MILP问题的求解复杂性。因为异或操作越多引入的变量就会越多，不等式数量就会越多，从而大大加大运行时间。

近年来，随着物联网的飞速发展，业界对加密算法的软件实现效率要求越来越高。由于ARX结构便于软件实现，越来越多的对称加密算法采用ARX结构（A代表模加操作，R代表循环移位操作，X代表异或操作，ARX算法指只包含模加、异或与循环移位操作的加密算法）。值得注意的是，与DES，AES等传统具有S盒操作的加密算法相比，ARX算法中所面临的独立性问题会大大加大其分析难度。尤其，搜索ARX算法的差分特征与线性逼近的算法采用的准则与搜索具有S盒操作加密算法的准则大大不同。自动化搜索ARX算法差分特征与线性逼近逐渐成为密码学界的一个研究重点。在文献[10, 12, 17]中，作者们提出一种自动化搜索ARX算法差分特征的方法。这种方法仅适用于基于ARX结构的HASH函数，而且有一个苛刻的限制条件，即要求密钥是已知的且可以被自由选择。在文献[4] 中，Biryukov与Velichkov利用部分差分分布表技术与Matsui的Branch-and-Bound技术，首次给出了一种自动化搜索ARX差分特征的算法，并

对Speck，TEA，XTEA等算法进行分析，取得了较好的结果。但该算法所需要的存储复杂度较高，且搜索精度与所选取的部分差分分布表有关。因此该算法是启发式的，搜索出的路线不一定是最优的。在即将发表在FSE2016的文章[5]中，Biryukov等人首次给出了一种Matsui的算法的改进版本，用来寻找ARX密码算法最优差分与线性路线。但这种方法仅对小分组（如Speck32/48等）算法具有优势，对于大分组（如Speck64/96/128等）算法结果相对较差。

与Branch-and-Bounding技术相比较，基于MILP的搜索工具对一些分组算法体现了较为显著的优势。尽管如此，已有的构造MILP模型的方法并不能直接应用到ARX 算法的分析中。举个例子，一个很直接的把MILP模型应用到ARX算法的思路如下：把F_2^n上的模加操作看做$2n \times n$的S盒操作，从而我们可以利用sun等人提出的方法来生成线性不等式并从中挑选出能完全描述模加操作的差分或线性分布情况的部分线性不等式子集。实际上，这种方法在现有计算条件下是不能应用到实际分析工作中。由于真实ARX算法中n一般大于16，生成线性不等式将是一件很难实现的事情。即使生成成功，那么最后筛选出的不等式个数也会很多，这大大增加了MILP 问题的求解复杂度，在现有计算水平下，是无法求解成功的。

基于以上原因，我们进行基于MILP搜索ARX算法的研究工作，希望能提出一种便于实现的搜索技术。

§1.2 论文结构组织与研究进展

1.2.1 研究进展

基于MILP针对ARX差分特征自动化搜索算法： 我们参考已有的对模加操作差分特征与线性逼近的研究，提出了一种构建ARX算法MILP模型的新思路。在文献[13]中，Lipmaa等人对模加操作差分特征进行了详细的研究，并提出一种计算模加操作差分概率的简便算法。他们指出：对于两输入的模加操作（$\alpha, \beta \rightarrow \gamma$）的差分特征，只有第$(i+1)$比特差分与第$i$比特差分满足特定的关系，差分概率才不为0，而且差分概率仅与$(\alpha[i], \beta[i], \gamma[i])$有关。利用该性质，我们发现用少量的线性不等式就可以完全刻画模加操作的差分特性，并且还可以计算其相应的差分概率。于是，我们在文献[13]的研

究基础上，提出一种基于MILP针对ARX算法差分特征搜索的新模型。

基于MILP针对ARX线性逼近自动化搜索算法： 在文献[19, 25]中，Nyberg与wallén等人对模加操作的线性逼近进行了详细的研究，提出一种计算线性逼近相关系数的状态基跳转算法。他们指出对于两输入的模加操作的$(\Gamma, \Lambda_\alpha, \Lambda_\beta)$相关系数，从第$(i+1)$比特掩码跳转到第$i$比特掩码过程中只有满足特定的关系，相关系数才不为0，而且相关系数仅与状态有关。基于以上研究，我们发现利用少量的线性不等式就可以完全刻画模加操作的线性特性，并且还可以计算其相应的线性相关系数。因此，我们可以构建一种新型的针对ARX算法线性逼近自动化搜索的MILP新模型。

值得注意的是， 由于新模型中不等式数量较少（远远少于把模加操作看做S盒操作进行处理所产生的不等式数量），该模型可以实际应用到ARX算法的分析工作中。我们可以把目标函数设置为求解最优差分概率或线性相关系数，从而可以利用Gurobi或者其他MILP求解器来对新模型进行求解。对模型求解就是自动化搜索最优差分特征或线性逼近的过程。

新MILP模型需要解决的问题： 然而，我们的新模型存在以下几个问题：在构建新模型过程中，我们假设模加操作的两个输入与轮与轮之间都是相互独立的。正如参考文献[26]中指出的一样，对于某些ARX算法来说，由于模加操作的输入是由分支操作产生，输入之间是不独立的且相互影响。这样我们的搜索算法也是启发式的，搜索结果可能会存在偏差。因此，对于某些固定密钥来说，通过新技术搜索得到的差分特征（或线性逼近）的概率（或相关系数）可能与实际的概率（或相关系数）有一定的偏差。这种偏差也相应的会对攻击者的成功率产生一定的影响。我们希望在后续研究工作中能解决独立性问题，使之更适用于实际的分析工作。

对Speck算法的差分线性路线搜索： 作为检验，我们将新提出的搜素技术应用到Speck算法的分析中，搜索其所有版本的差分特征与线性逼近。为了让MILP工具在可接受的时间内得到对大分组算法的搜索结果（>48比特），我们采用**启发式拼接技术**，即将分组算法分成两或三部分-最高，（中间），最低。我们独立搜索每一部分的路线，并且保证相邻两部分之间的输入输出差分（掩码）是一致的。在假设模加操作的两个输入与轮与轮之间是相互独立的条件下，我们在Speck48，Speck64，Speck96与Speck128搜索出比已有结果更好的差分特征。利用新搜索的差分特征，我们使用guess-

and-determine技术改进了相应四个版本Speck算法的差分攻击结果。与已有的最好的攻击结果[9]相比较，我们对Speck48与Speck64的攻击提高了1轮，对Speck96的攻击提高了3轮，对Speck128的攻击提高了5轮。在表 1中，我们总结了已有的对Speck的攻击。此外，我们在表 2中将我们搜到的差分特征与线性逼近与已有的结果进行比较。

模减差分的SMT模型： 最后，我们研究异或操作与移位操作的模减差分的特性，并首次给出一种构建ARX算法模减差分特征的SMT模型。在文献 [34]中，Lipmaa等人对两输入的异或操作（$\alpha \oplus \beta = \gamma$）的模减差分进行研究，指出当模减差分概率不为0时，α，β与γ满足特定的形式。我们利用该表达式构建异或操作的SMT模型。此外，在CT-RSA2014上，Biyukov等人给出移位操作的模减差分的分析结果[4]。基于他们的研究，我们给出了移位操作的SMT模型。最后，我们给出衡量模减差分概率的一个指标变量，并设置其为目标函数。利用该模型，我们可以利用SMT求解器STP等进行求解，从而搜索出好的模减差分特征。

SMT模型需要解决的问题： 然而，我们的新模型存在以下几个问题：在构建新模型过程中，我们假设异或操作的两个输入与轮与轮之间都是相互独立的。此外，我们为了便于实现，没有将异或操作以及右移位操作的模减概率加进模型。因此，我们的模型是启发式的，必须检验搜索出来的路线并计算其理论概率。我们希望在后续研究工作中能解决独立性问题与概率问题，使之更适用于实际的分析工作。

1.2.2 论文组织结构

本论文共分为7章，各章节具体内容如下：

第1章为引言，主要介绍了本论文的研究背景与我们的研究进展。

第 2章简要的介绍了已有的基于MILP针对带有S盒分组加密算法差分特征与线性逼近的自动化搜索算法。

第 3章主要介绍了我们提出的基于MILP进行ARX算法差分特征自动化搜索的新技术。

第 4章主要介绍了我们提出的基于MILP进行ARX算法线性逼近自动化搜索的新技术。

第5章主要介绍利用新提出的技术对Speck算法进行搜索，改进了Speck48，Speck64，Speck96与Speck128算法的差分攻击结果。我们的攻击结果就轮数而言是已知的最优结果。

第6章主要讨论了ARX算法模减差分SMT模型的构建，并首次提出一种构建方法。

第7章对本文进行了总结，并在此基础上提出了未来工作中可能的改进。

表 1: Speck算法攻击总结

算法版本$2n/mn$	攻击轮数/ 总轮数	时间复杂度	数据复杂度	存储复杂度	攻击方法	参考文献
48/72	11/22	$2^{67.93}$	$2^{43.727}$	-	线性攻击	[30]
	12/22	$2^{58.8}$	$2^{43.2}$	$2^{45.8}$	矩形攻击	[1]
	10/22	$2^{45.3}$	2^{45}	2^{24}	差分攻击	[1]
	14/22	2^{65}	2^{41}	2^{22}	差分攻击	[9]
	15/22	$\mathbf{2^{70}}$	$\mathbf{2^{46}}$	$\mathbf{2^{22}}$	差分攻击	本文
48/96	12/23	$2^{91.93}$	$2^{43.727}$	-	线性攻击	[30]
	12/23	$2^{58.8}$	$2^{43.2}$	$2^{45.8}$	矩形攻击	[1]
	12/23	$2^{45.3}$	2^{45}	2^{24}	差分攻击	[1]
	15/23	2^{89}	2^{41}	2^{22}	差分攻击	[9]
	16/23	$\mathbf{2^{94}}$	$\mathbf{2^{46}}$	$\mathbf{2^{22}}$	差分攻击	本文
64/96	14/26	$2^{94.9}$	$2^{62.7}$	-	线性攻击	[30]
	14/26	$2^{89.4}$	$2^{63.6}$	$2^{65.6}$	矩形攻击	[1]
	15/26	$2^{61.1}$	2^{61}	2^{32}	差分攻击	[1]
	18/26	2^{93}	2^{61}	2^{22}	差分攻击	[9]
	19/26	$\mathbf{2^{95}}$	$\mathbf{2^{63}}$	$\mathbf{2^{22}}$	差分攻击	本文
64/128	15/27	$2^{126.9}$	$2^{62.7}$	-	线性攻击	[30]
	14/27	$2^{89.4}$	$2^{63.6}$	$2^{65.6}$	矩形攻击	[1]
	15/27	$2^{61.1}$	2^{61}	2^{32}	差分攻击	[1]
	19/27	2^{125}	2^{61}	2^{22}	差分攻击	[9]
	20/27	$\mathbf{2^{127}}$	$\mathbf{2^{63}}$	$\mathbf{2^{22}}$	差分攻击	本文
96/96	8/28	$2^{74.7}$	$2^{27.6}$	-	线性攻击	[30]
	15/28	$2^{89.1}$	2^{89}	2^{48}	差分攻击	[1]
	16/28	2^{85}	2^{85}	2^{22}	差分攻击	[9]
	19/28	$\mathbf{2^{88}}$	$\mathbf{2^{88}}$	$\mathbf{2^{22}}$	差分攻击	本文
96/144	9/29	$2^{122.7}$	$2^{27.6}$	-	线性攻击	[30]
	16/29	$2^{135.9}$	$2^{90.9}$	$2^{94.5}$	矩形攻击	[1]
	15/29	$2^{89.1}$	2^{89}	2^{48}	差分攻击	[1]
	17/29	2^{133}	2^{85}	2^{22}	差分攻击	[9]
	20/29	$\mathbf{2^{136}}$	$\mathbf{2^{88}}$	$\mathbf{2^{22}}$	差分攻击	本文
128/128	8/32	$2^{92.7}$	$2^{28.3}$	-	线性攻击	[30]
	16/32	$2^{111.1}$	2^{116}	2^{64}	差分攻击	[1]
	17/32	2^{113}	2^{113}	2^{22}	差分攻击	[9]
	22/32	$\mathbf{2^{120}}$	$\mathbf{2^{120}}$	$\mathbf{2^{22}}$	差分攻击	本文
128/192	9/33	$2^{156.7}$	$2^{28.3}$	-	线性攻击	[30]
	16/33	$2^{111.1}$	2^{116}	2^{64}	差分攻击	[1]
	18/33	$2^{182.7}$	$2^{125.9}$	$2^{121.9}$	矩形攻击	[1]
	18/33	2^{177}	2^{113}	2^{22}	差分攻击	[9]
	23/33	$\mathbf{2^{184}}$	$\mathbf{2^{120}}$	$\mathbf{2^{22}}$	差分攻击	本文
128/256	7/34	$2^{220.7}$	$2^{28.3}$	-	线性攻击	[30]
	16/34	$2^{111.1}$	2^{116}	2^{64}	差分攻击	[1]
	18/34	$2^{182.7}$	$2^{125.9}$	$2^{121.9}$	矩形攻击	[1]
	19/34	2^{241}	2^{113}	2^{22}	差分攻击	[9]
	24/34	$\mathbf{2^{248}}$	$\mathbf{2^{120}}$	$\mathbf{2^{22}}$	差分攻击	本文

表 2: Speck算法差分特征与线性逼近的总结

	差分特征			线性逼近		
算法	# 轮数	$\log_2 p$	参考文献	# 轮数	$\log_2 c$	参考文献
Speck32	9	-31	[1]	9	-14	[30]
	9	-30	[6]	9	-14	本文
	9	-30	本文			
Speck48	10	-41	[1]	9	-20	[30]
	11	-47	[6]	10	-22	本文
	11	-45	本文			
Speck64	13	-59	[1]	11	-25	[30]
	13	-51	本文	12	-31	[30]
	14	-60	[6]	13	-30	本文
	14	-56	本文			
	15	-62	本文			
Speck96	13	-84	[1]	6	-11	[30]
	13	-67	本文	15	-45	本文
	16	-87	本文			
Speck128	14	-112	[1]	6	-11	[30]
	14	-90	本文	16	-58	本文
	19	-119	本文			

第二章 差分与线性路线的自动化搜索算法简介

在本章中，我们简单介绍已有的对分组密码算法差分特征与线性逼近的自动化搜索算法。由于本文主要基于MILP构建搜索模型，我们着重介绍sun等人在Asiacrypt2014提出的基于MILP进行（相关密钥）差分与线性路线（壳）的自动化搜索算法。如果想要获得有关其算法的更多细节，请参考文献[22, 24]。首先，我们简单介绍下差分分析与线性分析技术。

§2.1 差分分析与线性分析

分组密码的差分分析思想形成于在1990年前后。在CRYPTO1991会议上，Biham与Shamir首次给出了对DES算法的差分攻击[3],从此掀起了密码学研究热潮。差分分析技术属于选择明文攻击。它的基本思想是研究特定差分明文对在加解密过程的概率传播特性，进而区分随机置换与分组算法，并在此基础上恢复部分密钥比特。

在EUROCRYPT1994上，Matsui提出一种对DES算法的一种新型攻击方法-线性分析[15]，并第一次用实验给出了对16轮DES的攻击。与差分分析不同，线性分析属于已知明文攻击。它的基本思想是基于统计思想，通过研究明文，密文以及密钥之间的线性关系来进行密钥恢复攻击。特别指出的是，在某些特定场景下，线性分析也可以用于唯密文攻击。

自从问世来，这两种技术就被广泛的应用到分组密码的分析中。经过十几年的发展和完善，差分分析和线性分析已经成为分组密码算法最经典最有效的两种攻击手段。对分组密码算法实现差分攻击或线性攻击，最重要的事情莫过于寻找一条“好”的差分特征或者线性逼近。“好”的定义指的是，差分特征（或者线性逼近）的轮数尽可能的长，或者差分概率（或者线性逼近）尽可能的大。找到一条“好”的差分或线性区分器后，我们可以根据算法特性往前或者往后扩几轮，来进行密钥恢复攻击。如图1所示，假设找到的差分或线性区分器是r轮，我们通过猜测K_m比特密钥或者中间状态信息往前扩展m 轮，并猜测K_n比特密钥或者中间状态信息往后增加n轮，来进行（$r+m+n$）轮的密钥恢复攻击。

因此，自动化搜索差分特征或线性逼近一直是密码学家的研究热点。目前，主要有以下几种自动化搜索技术：

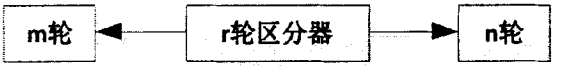

图 1: 差分线性攻击步骤

1. Branch-and-Bounding技术：Matsui在EUROCRYPT1994首次提出Branch-and-Bounding技术，并搜索出DES 算法的差分特征以及线性逼近。**但是该算法仅适用于单差分模式。**

2. 扩展的Branch-and-Bounding技术：Biyukov与Nikolic在FSE2011上改进了Matsui的算法，使之能搜索相关密钥模式下的差分特征[35]。**但是该算法仅适用于具有线性密钥生成算法的分组加密算法。** 此外，Biyukov等人在[4]中，利用Branch-and-Bounding与部分差分分布表技术首次给出了一种自动化搜索ARX差分特征的算法，并在[5]中进行改进，使之用来搜索最优路线。**该算法可以给出安全界，但仅在小分组算法有优势。**

3. 基于整数规划（Integer Programming）的搜索技术：Mouha[18]与Wu[28]等人率先成功将计算最少差分活跃S盒个数转化为整数规划问题。这样就可以很容易的用开源或者商业整数规划求解器来进行自动化求解，从而得到最少差分活跃S盒个数。**这种技术适用于单差分模式与相关密钥模式，但只能给出安全界，不能直接用来搜索“好”的差分特征。**

4. 基于MILP的搜索技术：sun等人在[22, 24]中提出一种可适用于包含比特级异或操作，S 盒操作与比特级线性置换的分组加密算法。该算法相比于以上三种方法，在自动化搜索分组密码算法的差分特征以及线性逼近上具有较大的优势。它在Simon等算法上搜索的路线比已有结果较好。**但该算法不能适用于ARX算法的自动化搜索。**

由于本文介绍构建针对ARX算法差分特征与线性逼近的MILP模型，我们在本章剩余部分主要介绍下sun 等人的工作。我们借鉴其提出的生成不等

式算法以及贪心算法来构建MILP新模型。

§2.2 差分特征的MILP模型

差分模型的目标函数： 令x_i代表第i比特的差分值。那么，如果第i比特没有差分时，$x_i = 0$,否则，$x_i = 1$。另一个比特变量A_j用来表示第j个S盒是否活跃。当第j个S盒是非活跃的，$A_j = 0$，否则$A_j = 1$。那么$\sum_j A_j$即代表加密轮函数与密钥生成算法中活跃S盒的个数。所以sun等人把目标函数设为$\sum_j A_j$，然后求解其最小值。

差分模型的限制条件： 对于比特级的异或操作，输入差分a, b与输出差分c（即$a \oplus b = c$）满足的限制条件如下:

$$\begin{cases} d_\oplus \geq a, d_\oplus \geq b, d_\oplus \geq c \\ a + b + c \geq 2d_\oplus \\ a + b + c \leq 2 \end{cases} \tag{2.1}$$

这里$d_\oplus$ 是一个新引进的比特级变量.

对于S盒操作，sun等人用一个比较精确的方法来刻画满足其差分特性的限制条件。对一个$\omega \times \nu$ 的S盒，它的输入差分与输出差分分别是$(x_0, \ldots, x_{\omega-1})$与$(y_0, \ldots, y_{\nu-1})$，且仍用$A_t$表示其是否活跃。那么下面不等式用来确保非零的输入差分必然导致S盒活跃（即输出差分不为0）以及当输出差分为0 时，输入差分必为0。

$$\begin{cases} A_t - x_k \geq 0, k \in \{0, \ldots, \omega - 1\} \\ -A_t + \sum_{j=0}^{\omega-1} x_j \geq 0 \end{cases} \tag{2.2}$$

他们定义一个$(\omega + \nu)$-维的向量$(x_0, \ldots, x_{\omega-1}, y_0, \ldots, y_{\nu-1}) \in \{0,1\}^{\omega+\nu} \subseteq \mathbb{R}^{\omega+\nu}$ 来表示S盒的输入输出差分模式。这里$\mathbb{R}$ 代表实数域。通过计算S盒所有输入输出差分模式闭包（即上述向量的所有可能情况）的H-表示，我们可以得到许多线性不等式。这些线性不等式都满足S盒所有可能的输入输出差分模式，并可以排除一些S盒的不可能差分模式。为了减少不等式的个数，使模型能更好的应用于实际分析中。他们又在文献[24]提出一种贪心算法来选取闭包H-表示的子集，进而得到数量较少的不等式。这些不等式可

以精确的描述S盒的所有差分模式（既满足S盒所有可能的输入输出差分模式，又可以排除S 盒所有不可能的输入输出差分模式），并且可以被用来构建MILP搜索模型。

对于比特级操作的置换操作，我们可以将其用异或表示出来，并用等式 2.1刻画满足其差分特性的限制条件。最后他们利用各种MILP求解器（譬如Gurobi[11]）求解到好的差分特征。值得注意的是，如果我们把目标函数的值设为N，那么只有当目标函数达到N，才会停止优化进程并输出当前的解。此时得到的解就是恰好有N个活跃S盒的差分特征。

值得注意的是，该方法不能直接用于ARX算法差分特征的自动化搜索。 直接利用该方法，我们可以把模加操作看做S盒，从而求出满足模加操作差分特性的线性不等式。但是，在实际分析工作中，模加操作的分组长度n往往大于16。在现有计算水平下，求解满足48比特（3×16）向量模式的不等式集合是几乎难以实现的。即使不等式成功，最后挑选出的不等式子集的数量也是很大的。这将大大增大了MILP求解复杂度。因为，我们在此动机下，希望寻找一种针对ARX算法差分特征的简单MILP模型。

此外，这种构建MILP模型的思路也可以用于线性逼近的搜索。

§2.3 线性逼近的MILP模型

线性模型的目标函数： 差分模型的许多标记可以直接用于线性模型的构建中。譬如A_j可以代表S盒是否活跃，以及目标函数仍是使和$\sum_j A_j$最小化。

线性模型的限制条件： 对于比特级的异或操作，输入掩码a, b与输出掩码c满足以下的限制条件：

$$a = b = c.$$

对每一个三分支操作，输入掩码与输出掩码满足下的限制关系：

$$\begin{cases} d_\curlywedge \geq a, d_\curlywedge \geq b, d_\curlywedge \geq c \\ a + b + c \geq 2d_\curlywedge \\ a + b + c \leq 2 \end{cases} \tag{2.3}$$

这里$d_\curlywedge$是新引入的比特级变量。

对于一个$\omega \times \nu$S盒,他们仍使用A_t表示其是否活跃。假设输入与输出掩码分别是$(x_0, \ldots, x_{\omega-1})$与$(y_0, \ldots, y_{\nu-1})$。如果输出掩码是非零的，那么$A_t = 1$，否则$A_t = 0$。而且我们可以用下面的限制条件来确保非零的输出掩码必然导致S盒活跃。

$$\begin{cases} A_t - y_k \geq 0, k \in \{0, \ldots, \nu - 1\} \\ -A_t + \sum_{j=0}^{\nu-1} y_j \geq 0 \end{cases} \tag{2.4}$$

与构建差分模型类似，他们用$(\omega+\nu)$-维的向量$(x_0, \ldots, x_{\omega-1}, y_0, \ldots, y_{\nu-1}) \in \{0,1\}^{\omega+\nu} \subseteq \mathbb{R}^{\omega+\nu}$来表示S 盒的所有输入输出线性模式。他们利用类似的方法来计算少量的线性不等式用于准确的描述S盒所有的输入输出线性模式。线性模型中的其余部分与差分模型中相对应的部分是类似的，我们就不一一赘述。

值得注意的是，该方法不能直接用于ARX算法线性逼近的自动化搜索。 直接利用该方法，我们可以把模加操作看做S盒，从而求出满足模加操作线性逼近的线性不等式。但是，在实际分析工作中，模加操作的分组长度n往往大于16。在现有计算水平下，求解满足48比特（3×16）向量模式的不等式集合是几乎难以实现的。即使不等式成功，最后挑选出的不等式子集的数量也是很大的。这将大大增大了MILP求解复杂度。因为，我们在此动机下，希望寻找一种针对ARX算法线性逼近的简单MILP模型。

寻找差分路径与线性壳的算法： 另外，该技术在文献[24]被扩展用来寻找差分路径或者线性壳。通过加入表示给定特性（固定差分或者线性掩码）的限制条件，他们不断更新MILP模型并且求得所有的包含给定差分或者线性掩码的路线。具体的步骤如下：

步骤1 构建描述加密算法（1到r轮）的差分特征或线性逼近的MILP模型M

步骤2 加入表示给定特性（固定的差分或线性）的限制条件

步骤3 利用MILP求解器求解模型M，如果得到一个可行解x，保存x，并且在M中加入不等式l^x，使解x对新模型不是可行解，如果该模型木法仅需求解，转到**步骤4**，否则，重复**步骤3**

步骤4 停止求解，得到所有满足给定特性的路线

第三章 基于MILP针对ARX加密算法差分特征的自动化搜索技术

在本章中，我们参考已有的对模加操作异或差分特征的研究，发现了一个重要的特性。利用该特性，我们提出一种用MILP模型描述模加操作差分特征的新方法。通过新方法，我们可以生成能够精确描述模加操作所有异或差分模式的线性不等式集合。而且不等式的数量相对较小。利用这些不等式集合，我们可以构建针对模加操作异或差分特性的MILP模型，进而可以构建针对ARX算法异或差分特征的MILP模型。

§3.1 模加操作的异或差分特征

首先，我们给出有关模加操作异或差分特征的定义。

定义3.1 令α,β与γ代表n比特的异或差分。那么模2^n加法操作的异或差分概率(DP)xdp^+可以通过如下方法计算：选取所有输入差分为α与β的n比特明文对(x,y)，然后进行加法操作，计算输出差分为γ的概率：

$$xdp^+(\alpha,\beta\rightarrow\gamma)=2^{-2n}\cdot\#\{(x,y):((x\oplus\alpha)+(y\oplus\beta))\oplus(x+y)=\gamma\}.$$

通过查阅文献，我们发现：在文献[13]中，Lipmaa等人提出了用于计算$xdp^+(\alpha,\beta\rightarrow\gamma)$的算法 2。算法 2主要分两步：第一步验证差分特征是否真实存在的；第二步则是计算差分特征的概率xdp^+。为了便于理解，我们把以上两步概括为定理 3.1与定理 3.2。

定理3.1 (参考文献[13]) 差分特征$(\alpha,\beta\rightarrow\gamma)$概率不为0当且仅当$(\alpha[0]\oplus\beta[0]\oplus\gamma[0])=0$与当$\alpha[i-1]=\beta[i-1]=\gamma[i-1]$ $(i\in[1,n-1])$ 时，$\alpha[i-1]=\beta[i-1]=\gamma[i-1]=\alpha[i]\oplus\beta[i]\oplus\gamma[i]$。

定理3.2 (参考文献[13]) 假设$(\alpha,\beta\rightarrow\gamma)$是一条真实存在的差分特征，那么差分概率$xdp^+=2^{-\sum_{i=0}^{n-2}\neg eq(\alpha[i],\beta[i],\gamma[i])}$，这里

$$eq(\alpha[i],\beta[i],\gamma[i])=\begin{cases}1 & \alpha[i]=\beta[i]=\gamma[i]\\ 0 & others\end{cases}.$$

定理 3.1可以用来判断模加操作差分特征$(\alpha, \beta \rightarrow \gamma)$是否真实存在。为了更好的理解，我们举一个例子，差分特征$(\alpha, \beta \rightarrow \gamma) = (11100, 11100 \rightarrow 11110)$是不可能存在的。因为$\alpha[0] = \beta[0] = \gamma[0] \neq \alpha[1] \oplus \beta[1] \oplus \gamma[1]$。利用定理 3.2，我们可以高效快速的计算出模加操作的差分概率。举例，对于差分特征$(\alpha, \beta \rightarrow \gamma) = (11100,\ 00110 \rightarrow 10110)$，差分概率$xdp^+(\alpha, \beta \rightarrow \gamma)$=$2^{-(\neg eq(0,0,0)+\neg eq(0,1,1)+\neg eq(1,1,1)+\neg eq(1,0,0))} = 2^{-2}$。

我们基于定理 3.2，可以得到以下结论：如果n比特差分特征是真实存在的，那么差分概率仅且仅与$(\alpha[i], \beta[i], \gamma[i])$有关，此处$i \in [0, n-2]$。利用这个特性，我们可以构造计算模加操作差分概率$xdp^+$的MILP模型。我们将在以下的章节进行更详细的介绍。

§3.2 模加操作差分特征的MILP模型

根据sun等人提出的模型，我们可以用五个不等式来表示定理 3.1中第一个限制条件：$\alpha[0] \oplus \beta[0] \oplus \gamma[0] = 0$：

$$\begin{cases} d_\oplus \geq \alpha[0], d_\oplus \geq \beta[0], d_\oplus \geq \gamma[0] \\ \alpha[0] + \beta[0] + \gamma[0] - 2d_\oplus \geq 0 \\ \alpha[0] + \beta[0] + \gamma[0] \leq 2 \end{cases} \tag{3.1}$$

这里$d_\oplus$ 是一个额外增加的比特级变量。

但是，经过研究，我们发现这五个不等式可以用一个等式来代替：

$$\alpha[0] + \beta[0] + \gamma[0] = 2 \times d_\oplus \tag{3.2}$$

同上，$d_\oplus$是一个额外增加的比特级变量。

接着，我们用向量$(\alpha[i], \beta[i], \gamma[i], \alpha[i+1], \beta[i+1], \gamma[i+1])$来刻画第$i$比特与第$(i+1)$比特差分值之间的关系。那么，根据定理 3.1，我们可以得到以下结论：这个向量只有56种可能的模式。举个例子，差分模式$(0,0,0,1,1,1)$是不可能的。因为$\alpha_i = \beta_i = \gamma_i \neq \alpha_{i+1} \oplus \beta_{i+1} \oplus \gamma_{i+1}$。此外，为了更加高效的计算差分概率，我们把变量$\neg eq(\alpha[i], \beta[i], \gamma[i])$加入到向量中构成新的7维向量。因为差分概率仅与$\neg eq(\alpha[i], \beta[i], \gamma[i])$有关。为了便于理解，我们在下边列出7维向量$(\alpha[i], \beta[i], \gamma[i],\ \alpha[i+1], \beta[i+1], \gamma[i+1], \neg eq(\alpha[i], \beta[i], \gamma[i]))$的全

部56 种可能模式：

$$(0,0,0,0,0,0,0),(0,0,0,0,1,1,0),(0,0,0,1,0,1,0),(0,0,0,1,1,0,0),$$
$$(0,0,1,0,0,0,1),(0,0,1,0,0,1,1),(0,0,1,0,1,0,1),(0,0,1,0,1,1,1),$$
$$(0,0,1,1,0,0,1),(0,0,1,1,0,1,1),(0,0,1,1,1,0,1),(0,0,1,1,1,1,1),$$
$$(0,1,0,0,0,0,1),(0,1,0,0,0,1,1),(0,1,0,0,1,0,1),(0,1,0,0,1,1,1),$$
$$(0,1,0,1,0,0,1),(0,1,0,1,0,1,1),(0,1,0,1,1,0,1),(0,1,0,1,1,1,1),$$
$$(0,1,1,0,0,0,1),(0,1,1,0,0,1,1),(0,1,1,0,1,0,1),(0,1,1,0,1,1,1),$$
$$(0,1,1,1,0,0,1),(0,1,1,1,0,1,1),(0,1,1,1,1,0,1),(0,1,1,1,1,1,1),$$
$$(1,0,0,0,0,0,1),(1,0,0,0,0,1,1),(1,0,0,0,1,0,1),(1,0,0,0,1,1,1),$$
$$(1,0,0,1,0,0,1),(1,0,0,1,0,1,1),(1,0,0,1,1,0,1),(1,0,0,1,1,1,1),$$
$$(1,0,1,0,0,0,1),(1,0,1,0,0,1,1),(1,0,1,0,1,0,1),(1,0,1,0,1,1,1),$$
$$(1,0,1,1,0,0,1),(1,0,1,1,0,1,1),(1,0,1,1,1,0,1),(1,0,1,1,1,1,1),$$
$$(1,1,0,0,0,0,1),(1,1,0,0,0,1,1),(1,1,0,0,1,0,1),(1,1,0,0,1,1,1),$$
$$(1,1,0,1,0,0,1),(1,1,0,1,0,1,1),(1,1,0,1,1,0,1),(1,1,0,1,1,1,1),$$
$$(1,1,1,0,0,1,0),(1,1,1,0,1,0,0),(1,1,1,1,0,0,0),(1,1,1,1,1,1,0).$$

根据文献[23]中描述的方法，我们可以用SAGE Computer Algebra System [20]中**sage. geometry. polyhedron**类中的**inequality_generator()**函数来生成满足以上56种模式的线性不等式。具体的代码与函数，请参考文献[23]。我们可以得到65个线性不等式满足以上56种可能的差分模式。为了加快搜索进程，我们又应用了文献[23]提出的贪心算法来减少不等式的数量。最后我们得到13个线性不等式。这13个线性不等式既满足以上56种差分模式，又可以判断差分模式$(\alpha[i],\beta[i],\gamma[i],\ \alpha[i+1],\beta[i+1],\gamma[i+1])$，$i \in [0,n-2]$是否真实存在。为了便于理解，我们在下边列出这13个线性不

等式：

$$\begin{aligned}
\beta[i]-\gamma[i]+(\neg eq(\alpha[i],\beta[i],\gamma[i]))&\geq 0,\\
\alpha[i]-\beta[i]+(\neg eq(\alpha[i],\beta[i],\gamma[i]))&\geq 0,\\
-\alpha[i]+\gamma[i]+(\neg eq(\alpha[i],\beta[i],\gamma[i]))&\geq 0,\\
-\alpha[i]-\beta[i]-\gamma[i]-(\neg eq(\alpha[i],\beta[i],\gamma[i]))&\geq -3,\\
\alpha[i]+\beta[i]+\gamma[i]-(\neg eq(\alpha[i],\beta[i],\gamma[i]))&\geq 0,\\
-\beta[i]+\alpha[i+1]+\beta[i+1]+\gamma[i+1]+(\neg eq(\alpha[i],\beta[i],\gamma[i]))&\geq 0,\\
\beta[i]+\alpha[i+1]-\beta[i+1]+\gamma[i+1]+(\neg eq(\alpha[i],\beta[i],\gamma[i]))&\geq 0,\\
\beta[i]-\alpha[i+1]+\beta[i+1]+\gamma[i+1]+(\neg eq(\alpha[i],\beta[i],\gamma[i]))&\geq 0,\\
\alpha[i]+\alpha[i+1]+\beta[i+1]-\gamma[i+1]+(\neg eq(\alpha[i],\beta[i],\gamma[i]))&\geq 0,\\
\gamma[i]-\alpha[i+1]-\beta[i+1]-\gamma[i+1]+(\neg eq(\alpha[i],\beta[i],\gamma[i]))&\geq -2,\\
-\beta[i]+\alpha[i+1]-\beta[i+1]-\gamma[i+1]+(\neg eq(\alpha[i],\beta[i],\gamma[i]))&\geq -2,\\
-\beta[i]-\alpha[i+1]+\beta[i+1]-\gamma[i+1]+(\neg eq(\alpha[i],\beta[i],\gamma[i]))&\geq -2,\\
-\beta[i]-\alpha[i+1]-\beta[i+1]+\gamma[i+1]+(\neg eq(\alpha[i],\beta[i],\gamma[i]))&\geq -2.
\end{aligned}\tag{3.3}$$

由于我们可以通过这13个不等式计算出变量$\neg eq(\alpha[i],\beta[i],\gamma[i])$的值。所以，这13个线性不等式也可以被用来计算$(\alpha[i]\|\beta[i]\|\gamma[i]\rightarrow\alpha[i+1]\|\beta[i+1]\|\gamma[i+1])$的差分概率。

然而，这13个线性不等式仅仅刻画了定理 3.1中的第二个限制条件：$\alpha[i]=\beta[i]=\gamma[i]=\alpha[i+1]\oplus\beta[i+1]\oplus\gamma[i+1]$, $i\in[0,n-2]$。因此，我们需要$(13\times(n-1)+1)$ 个线性不等式来完全刻画两个n比特输入模2^n加法操作的差分特性。最后根据定理 3.2，我们可以得到差分概率$xdp^+=2^{-\sum_{i=0}^{n-2}\neg eq(\alpha[i],\beta[i],\gamma[i])}$。

§3.3 ARX加密算法差分特征的MILP模型

除了模加操作，ARX加密算法还包括异或操作，三分支操作以及循环移位操作。下边我们对这三种操作进行MILP建模。对于每一个比特级异或操作，我们可以用不等式(3.2)来刻画其差分特性。对于三分支操作，假设输

入输出差分分别为a，b，c，那么限制条件如下：

$$a = b = c.$$

对于循环移位操作，我们可以对相对应的比特列出等式。目前为止，我们可以用线性不等式或者等式来刻画ARX加密算法中的每一步操作。这些不等式和等式组成了构建ARX加密算法差分特征的MILP模型中的限制条件。

正如章节 3.2所描述的，差分概率$xdp^+ = 2^{-\sum_{i=0}^{n-2} \neg eq(\alpha[i],\beta[i],\gamma[i])}$。那么我们可以把$\sum_{j=1}^{r}\sum_{i=0}^{n-2} \neg eq(\alpha_j[i], \beta_j[i], \gamma_j[i])$设置成$r$轮差分特征的目标函数。这里$\alpha_j$, β_j与γ_j分别代表第j轮模加操作的输入差分与输出差分。我们的目标是寻找$\sum_{j=1}^{r}\sum_{i=0}^{n-2} \neg eq(\alpha_j[i], \beta_j[i], \gamma_j[i])$的最小值。它代表了最好的差分特征的概率。我们可以通过Gurobi求解器[11]求解不等式集合来进行自动化搜索ARX加密算法的差分特征。然而，在现实的密码分析中，得到一条差分特征有时候是远远不够的。我们可以应用sun等人在文献[24]中提出的方法来扩展我们的MILP模型。利用新模型，我们可以搜索ARX算法的差分路径。

值得注意的是，在以上的建模过程中，我们假设模加操作的两个输入与连续的两轮之间都是独立的。然而，对于某些ARX算法来说，他们并不是独立的。这样可能导致在某些特定密钥下，我们搜索得到的结果与理论概率值有一定的偏差。这种偏差相应的会对攻击者的成功率产生一定的影响。

§3.4 小结

本章中，我们利用Lipmaa等人在FSE2002上的研究成果来构建ARX算法差分特征的MILP模型。首先，我们根据其研究构造出满足模加操作差分特征第i比特与第$(i+1)$比特关系的MILP模型，然后又根据其提出的计算差分概率的简便算法，将差分概率加入模型，使搜索更加准确。基于模加操作输入与轮与轮之间相互独立的假设，我们给出了一种基于针对ARX算法差分特征进行自动化搜索的新方法。我们的新算法是切实可行的，并在第 5章对Speck算法进行分析。然而，新模型还存在独立性的问题。对于某些ARX算法来说，模加操作的输入以及轮与轮之间不一定是独立的。因此，在某些特定密钥下，我们搜索得到的结果与理论差分概率有一定的偏差。这种偏差相应的会对攻击者的成功率产生一定的影响。我们希望在后续工作中能克服这一问题。

第四章 基于MILP针对ARX加密算法线性逼近的自动化搜索技术

在本章中，我们重新研究了模加操作线性逼近的特性，并且提出一种基于MILP进行ARX算法线性逼近自动化搜索的新方法。

§4.1 模加操作的线性逼近

令n代表一个非负整数。给定两个向量$x=(a_{n-1},\ldots,a_0)$与$y=(b_{n-1},\ldots,b_0)\in\mathbb{F}_2^n$，定义$x\cdot y$代表标准的内积操作$x\cdot y=a_{n-1}b_{n-1}\oplus\cdots\oplus a_0b_0$。

接下来我们给出模加操作线性相关系数的定义：

定义4.1 令Λ_α, Λ_β与Γ代表给定的n比特线性掩码。那么模2^n加法操作的线性相关系数可以按照以下公式计算：选取所有所有掩码$\Lambda_\alpha,\Lambda_\beta$为$n$比特明文对(x,y)，然后进行加法操作，计算输出掩码为γ时的相关系数：

$$\begin{aligned}&cor_{\boxplus}(\Gamma,\Lambda_\alpha,\Lambda_\beta)=cor(\Gamma\cdot(x+y)\oplus\Lambda_\alpha\cdot x\oplus\Lambda_\beta\cdot y)\\&=2^{-2n}(\#\{x,y\in\mathbb{F}_2^n:\Gamma\cdot(x+y)\oplus\Lambda_\alpha\cdot x\oplus\Lambda_\beta\cdot y=0\}\\&-\#\{x,y\in\mathbb{F}_2^n:\Gamma\cdot(x+y)\oplus\Lambda_\alpha\cdot x\oplus\Lambda_\beta\cdot y=1\}).\end{aligned}$$

基于对进位函数线性逼近的简单分类，Nyberg与Wallén在文献[19, 25]中推导出一种快速计算k输入的模2^n加法操作的线性相关系数的算法。因为只考虑两输入模加操作，所以我们只参考利用针对两输入的方法。为了便于理解，我们把他们提出的算法概括为定理 4.1。

定理4.1 (参考文献[19, 25]) 对于两输入的模2^n的加法操作，假设其输入掩码与输出掩码分别为$\Lambda_\alpha,\Lambda_\beta$ 与Γ, $\Lambda_\alpha,\Lambda_\beta,\Gamma\in\mathbb{F}_2^n$。其中，$\Lambda_\alpha=(\Lambda_\alpha[n-1],\ldots,\Lambda_\alpha[0])$, $\Lambda_\beta=(\Lambda_\beta[n-1],\ldots,\Lambda_\beta[0])$, $\Gamma=(\Gamma[n-1],\ldots,\Gamma[0])$。我们定义一个新的向量$u=(u[n-1],\ldots,u[0])$，这里$u[i]=4\Gamma[i]+2\Lambda_\alpha[i]+\Lambda_\beta[i],0\leq u[i]<8,0\leq i<n$。那么相关系数可由下面的线性表达式计算：

$$cor_{\boxplus}(\Gamma,\Lambda_\alpha,\Lambda_\beta)=LA_{u[n-1]}A_{u[n-2]}\cdots A_{u[1]}A_{u[0]}C, \tag{4.1}$$

其中$A_r, r = 0, \ldots, 7$,代表2×2的矩阵,

$$A_0 = \frac{1}{2}\begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}, A_1 = A_2 = -A_4 = \frac{1}{2}\begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix},$$

$$A_7 = \frac{1}{2}\begin{bmatrix} 0 & 2 \\ 1 & 0 \end{bmatrix}, -A_3 = A_5 = A_6 = \frac{1}{2}\begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix},$$

L是一个行向量$L = (1\ 0)$，C 是一个列向量$C = (1\ 1)^T$.

举例说明，对于线性逼近($\Gamma = 10100, \Lambda_\alpha = 11110, \Lambda_\beta = 11000$), $u = 73620_8$，那么相关系数$cor_{\boxplus}$=$LA_7A_3A_6A_2A_0C = -2^{-3}$。

为了快速实现定理 4.1，Nyberg and Wallén利用自动状态基从左到右计算相关系数$LA_{u[n-1]}A_{u[n-2]}\cdots A_{u[1]}A_{u[0]}C$。令$e_0 = L = (1\ 0)$ ，$e_1 = (0\ 1)$。那么我们可以发现：

$$e_0A_0 = e_0, \quad e_0A_7 = e_1, \quad e_0A_r = 0, for\ r \in [1, 6], \quad e_1A_3 = -\frac{1}{2}e_1,$$
$$e_1A_0 = e_1A_5 = e_1A_6 = \frac{1}{2}e_1, e_1A_1 = e_1A_2 = e_1A_7 = \frac{1}{2}e_0, e_1A_4 = -\frac{1}{2}e_0. \quad (4.2)$$

我们在图 2中展示了状态跳转过程。当从左到右计算u时，如果自动状态基以状态0结束，那么$LA_{u[n-1]}A_{u[n-2]}\cdots A_{u[1]}A_{u[0]}C = 0$，即说明此时相关系数为0。如果状态基以状态$e_0$或者$e_1$结束，那么$LA_{u[n-1]}A_{u[n-2]}\cdots A_{u[1]}A_{u[0]}C = \pm 2^{-t}$。其中$t$是跳转过程中实线箭头的个数。相关系数的正负号则是由$\{3, 4\}$出现的个数所决定的：

$$LA_{u[n-1]}A_{u[n-2]}\cdots A_{u[1]}A_{u[0]}C > 0$$

当且仅当出现$\{3, 4\}$出现的次数是偶数。举个例子：$u = 73620_8$, $LA_7A_3A_6A_2A_0C = -2^{-3}$.

基于等式 (4.2)中，我们推导得到性质 4.1。性质 4.1主要用于快速计算相关系数$cor_{\boxplus}(\Gamma, \Lambda_\alpha, \Lambda_\beta)$的绝对值。

性质4.1 对于两输入的模2^n加法操作，假设其输入掩码与输出掩码分别为$\Lambda_\alpha, \Lambda_\beta$与$\Gamma$，并且满足图 3中的状态跳转过程。这里$u[i] = 4\Gamma[i] + 2\Lambda_\alpha[i] +$

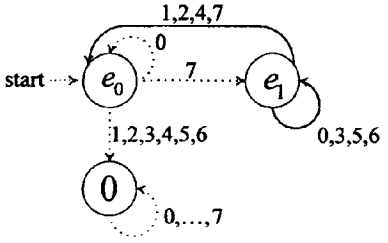

图 2: 对于$u = 73620_8$的状态跳转过程

$\Lambda_\beta[i], 0 \le u[i] < 8, 0 \le i < n$，$\epsilon_j \in \{e_0, e_1\}, 0 \le j < n$。如果线性逼近的相关系数是非零的，那么相关系数的绝对值可以按照下面的式子进行计算：

$$|cor_{\boxplus}(\Gamma, \Lambda_\alpha, \Lambda_\beta)| = 2^{-\#\{0<i<n|\epsilon_i=e_1\}}. \tag{4.3}$$

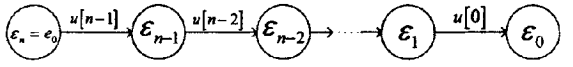

图 3: 模2^n加法操作的状态跳转过程

证明如下：

Proof. 假设线性逼近的相关系数是非零的，那么从等式 (4.2)中，我们可以知道状态转换过程具有以下的性质：$e_0 A_r = 0, r \in [1, 6]$是不可能存在的，其余的状态转换过程如下：

$$\begin{aligned} &e_0 A_0 = e_0, \quad e_0 A_7 = e_1, \quad e_1 A_3 = -\frac{1}{2} e_1, \quad e_1 A_4 = -\frac{1}{2} e_0, \\ &e_1 A_0 = e_1 A_5 = e_1 A_6 = \frac{1}{2} e_1, \quad e_1 A_1 = e_1 A_2 = e_1 A_7 = \frac{1}{2} e_0. \end{aligned} \tag{4.4}$$

当考虑第i个跳转过程即在$u[n-i]$（$1 \le i \le n$）下从状态ϵ_{n+1-i}跳转到ϵ_{n-i}时，我们可以得到$|\epsilon_{n+1-i} A_{u[n-i]}| = 2^{-\tau_{n+1-i}} \epsilon_{n-i}$。其中，如果$\epsilon_{n+1-i} = e_0$，那么$\tau_{n+1-i} = 0$。相反的，当$\epsilon_{n+1-i} = e_1$时，$\tau_{n+1-i} = 1$。那么当从左到右开始

计算$LA_{u[n-1]}A_{u[n-2]}\cdots A_{u[1]}A_{u[0]}C$ 时，我们可以得到：

$$\begin{aligned}|cor_{\boxplus}(\Gamma,\Lambda_\alpha,\Lambda_\beta)| &= \left|e_0A_{u[n-1]}A_{u[n-2]}\cdots A_{u[1]}A_{u[0]}C\right| \\ = \left|2^0\cdot\epsilon_{n-1}A_{u[n-2]}\cdots A_{u[1]}A_{u[0]}C\right| &= \left|2^0\cdot 2^{-\tau_{n-1}}\epsilon_{n-2}A_{u[n-3]}\cdots A_{u[1]}A_{u[0]}C\right| \\ &\vdots \\ = \left|2^{-\sum_{i=1}^{n-1}\tau_i}\epsilon_0C\right| &= 2^{-\sum_{i=1}^{n-1}\tau_i} = 2^{-\#\{0<i<n|\epsilon_i=e_1\}} \quad \square\end{aligned}$$

□

基于性质 4.1的观察，我们可以构造计算两输入模加操作线性逼近相关度绝对值的MILP模型。

§4.2 模加操作线性逼近的MILP模型

在本小节中，我们将介绍一种利用线性不等式来刻画定理 4.1与性质4.1中所描述的模加操作的线性特性的新方法。

为了更好的描述在$u[i]$（$0\le i<n$）下从状态ϵ_{i+1}跳转到状态ϵ_i的过程，我们引入一个新的比特变量s_i。新变量定义如下：如果$\epsilon_i=e_0$，$s_i=0$，相反的，$s_i=1$当$\epsilon_i=e_1$时。我们用5维向量$(s_{i+1},\Gamma[i],\Lambda_\alpha[i],\Lambda_\beta[i],s_i)$来表示状态转移过程$e_{s_{i+1}}A_{u[i]}=e_{s_i}$。对于向量$(s_{i+1},\Gamma[i],\Lambda_\alpha[i],\Lambda_\beta[i],s_i)$，一共会有$2^5$可能的情况。但是根据等式 (4.4)可知，向量一共只有10可能的情况。为了更好的理解，我们在下面列出向量$(s_{i+1},\Gamma[i],\Lambda_\alpha[i],\Lambda_\beta[i],s_i)$的10种可能情况：

$$\begin{aligned}&(0,0,0,0,0),(0,1,1,1,1),(1,0,0,0,1),(1,0,1,1,1),(1,1,0,1,1),\\&(1,1,1,0,1),(1,0,0,1,0),(1,0,1,0,0),(1,1,0,0,0),(1,1,1,1,0).\end{aligned}$$

与第 3.2章描述的方法类似,我们可以用SAGE Computer Algebra System [20]中**sage. geometry. polyhedron** 类中的**inequality_generator()**函数与贪心算法来生成满足上述10种可能模式的线性不等式。我们最后得到8个满

足上述10种可能模式线性不等式。这8个线性不等式如下：

$$
\begin{aligned}
& s_{i+1} - \Gamma[i] - \Lambda_\alpha[i] + \Lambda_\beta[i] + s_i \geq 0, \quad s_{i+1} + \Gamma[i] + \Lambda_\alpha[i] - \Lambda_\beta[i] - s_i \geq 0, \\
& s_{i+1} + \Gamma[i] - \Lambda_\alpha[i] - \Lambda_\beta[i] + s_i \geq 0, \quad s_{i+1} - \Gamma[i] + \Lambda_\alpha[i] - \Lambda_\beta[i] + s_i \geq 0, \\
& s_{i+1} + \Gamma[i] - \Lambda_\alpha[i] + \Lambda_\beta[i] - s_i \geq 0, \quad s_{i+1} - \Gamma[i] + \Lambda_\alpha[i] + \Lambda_\beta[i] - s_i \geq 0, \\
- \quad & s_{i+1} + \Gamma[i] + \Lambda_\alpha[i] + \Lambda_\beta[i] + s_i \geq 0, \quad s_{i+1} + \Gamma[i] + \Lambda_\alpha[i] + \Lambda_\beta[i] + s_i \leq 4.
\end{aligned}
$$

上述8个线性不等式完全刻画了在$u[i]$下从状态ϵ_{i+1}到状态ϵ_i跳转过程的限制条件。在这里，需要特别指出：在图3中，初始状态$\epsilon_n = e_0$，所以我们得需要加一个额外的限制条件：$s_n = 0$。这样，我们就可以用$8 \times n + 1$个线性不等式来刻画两输入模加操作的线性逼近的特性，并且可以通过$|cor_{\boxplus}| = 2^{-\sum_{i=1}^{n-1} s_i}$来计算相关系数的绝对值。

§4.3 ARX加密算法线性逼近的MILP模型

与构建差分模型类型类似，我们对ARX算法的每一步操作进行MILP建模。对于每一个异或操作，三分支操作与循环移位操作，我们可以用 2章中的方法来生成相应的线性不等式或等式。对于每一个三分支操作，我们可以用不等式(3.2)来刻画其差分特性。对于异或操作，假设输入输出差分分别为a，b，c，那么限制条件应该如下：

$$a = b = c.$$

对于循环移位操作，我们可以对相对应的比特列出等式。目前为止，我们可以用线性不等式或者等式来刻画ARX加密算中的每一步操作。这些不等式和等式组成了针对ARX加密算法线性逼近的MILP模型中的限制条件。

正如章节 4.1所描述的，线性逼近相关系数的绝对值$cor_{\boxplus}(\Gamma, \Lambda_\alpha, \Lambda_\beta) = \sum_{j=1}^{r} \sum_{i=1}^{n-1} s_i$。那么我们可以把$\sum_{j=1}^{r} \sum_{i=1}^{n-1} s_i$设置成$r$ 轮线性逼近的目标函数。我们的目标是寻找$\sum_{j=1}^{r} \sum_{i=1}^{n-1} s_i$的最小值。它代表了最好的线性逼近的相关系数的绝对值。我们可以通过Gurobi求解器[11]来求解不等式集合进行自动化搜索ARX加密算法的线性逼近。

正如在构建差分特征模型中描述的一样，我们假设模加操作的两个输入与连续的两轮之间都是独立的。然而，对于某些ARX算法来说，他们并不

是独立的。这样可能导致在某些特定密钥下，我们搜索得到的结果与理论相关系数有一定的偏差。这种偏差相应的会对攻击者的成功率产生一定的影响。

§4.4 小结

本章中，我们利用Nyberg与Wallén在FSE2006上的研究成果来构建ARX算法线性逼近的MILP模型。首先，我们根据其其提出的状态基跳转算法构造出满足模加操作线性逼近从第$(i+1)$比特到第i比特跳转过程的MILP模型，然后其跳转特性，推导出相关系数只与状态有关，并将相关系数加入模型，使之搜索更精确。基于模加操作输入与轮与轮之间相互独立的假设，我们给出了一种基于针对ARX算法线性逼近进行自动化搜索的新方法。我们的新算法是切实可行的，在第 5章对Speck算法进行分析。然而，新模型还存在独立性的问题。对于某些ARX算法来说，模加操作的输入以及轮与轮之间不一定是独立的。因此，在某些特定密钥下，我们搜索得到的结果与理论相关系数有一定的偏差。这种偏差相应的会对攻击者的成功率产生一定的影响。我们希望在后续工作中能克服这一问题。

第五章 对Speck算法的攻击

§5.1 Speck算法描述与安全性分析

5.1.1 Speck算法描述

Speck是美国国家安全局（NSA）于2013年基于ARX结构设计的一系列轻量级分组加密算法[2]。该算法结构简单，由简单的加法，异或，循环移位三种基本运算构成。Speck系列分组密码算法给出了10个应用版本,可以根据具体的安全性要求、性能要求、应用环境等选择合适的分组长度和密钥长度,具有很强的灵活性,应用前景十分广泛。Speck算法的分组长度为$2n$比特（其中n是一个字的长度），密钥长度为mn比特。所以，我们用Speck$2n/mn$来表示分组长度为$2n$比特密钥长度为mn比特的Speck算法。所有版本Speck算法轮函数用到的循环移位常数α, β，算法轮数长度，以及具体的具体的m 与n值都被列在表 3。

该算法的轮函数结构较简单,主要由循环移位操作、异或运算和域上的模加运算组成。算法的密钥扩展函数也调用了轮函数,这使得Speck系列算法的加密效率很高,有利于进行软件实现。如果我们用k_i表示第i轮的子密钥，第i轮输入输出分别为(x_{i-1}, y_{i-1})与(x_i, y_i)，那么轮函数可以用下面等式表示：

$$x_i = ((x_{i-1} \ggg \alpha) \boxplus y_{i-1}) \oplus k_i, \qquad y_i = (y_{i-1} \lll \beta) \oplus x_i,$$

这里α and β代表表 3中列出的循环移位常数。具体的算法结构如图 4.

5.1.2 Speck安全性分析

自从问世来，Speck就受到密码学界的极大关注。目前，密码学家给出很多Speck算法的攻击结果。在文献[1]中，Abed等人给出了几乎所有版本Speck的差分攻击与矩形攻击。在FSE2014会议文章[6] 中，Birukov等人搜索出9轮Speck32,11轮Speck48与14轮Speck64的差分特征。他们的结果比文献[1]中提供的差分特征要好一些。在文献[9]中，Dinur提出一种sub-cipher攻击方法，并改进了所有版本Speck算法的密钥恢复攻击结果。这里说明一

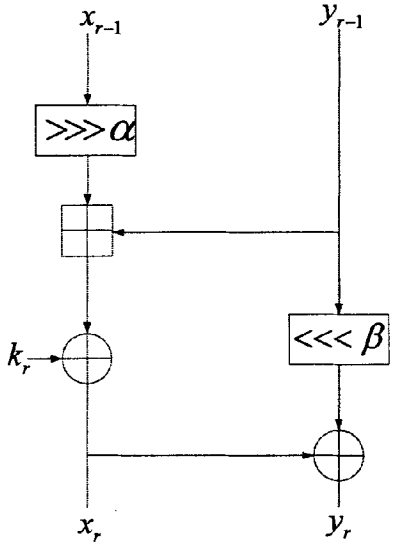

图 4: Speck算法轮函数

表 3: Speck算法的参数设置

版本$2n/mn$	分组长度$2n$	字长度n	密钥长度mn	密钥字长度m	轮数r	α	β
32/64	32	16	64	4	22	7	2
48/72	48	24	72	3	22	8	3
48/96	48	24	96	4	23	8	3
64/96	64	32	96	3	26	8	3
64/128	64	32	128	4	27	8	3
96/96	96	48	96	2	28	8	3
96/144	96	48	144	3	29	8	3
128/128	128	64	128	2	32	8	3
128/192	128	64	192	3	33	8	3
128/256	128	64	256	4	34	8	3

下，Dinur的攻击使用了文献[6]中的差分特征。在文献[5]中，Biryukov等人给出了10轮Speck32, 9轮Speck48,8轮Speck64,7轮Speck96与6轮SPeck128的最优差分路线，并且在马尔科夫独立假设下评估了Speck算法抵抗单密钥差分分析的安全界。关于线性分析的结果，Yao等人在文献[27]中搜索出9 轮Speck32,9轮Speck48, 12轮Speck64,6 轮Speck96与6轮Speck128的线性路线，并给出了相应的恢复密钥攻击。具体的结果请参考表 1。

§5.2 Speck加密算法的差分特征与线性逼近

在本小节中，我们详细解释如何利用第 3章与第 4章中介绍的方法来自动化搜索Speck加密算法差分特征与线性逼近。

首先，我们基于第 3章与第 4章中介绍的方法生成线性不等式集合。这些不等式集合被用来构建自动化搜索差分特征和线性逼近的MILP模型。接着我们采用文献[21, 22, 23, 24]中的方法，利用Gurobi求解器[11]对我们的MILP模型就行求解。实际上，其他MILP求解器，譬如CPLEX[8]等，也可以被用来进行不等式的求解。我们的方法主要分为以下两步：

步骤1 将描述r轮Speck算法差分特征或线性的不等式集合转换成Gurobi可以识别的格式。

步骤2 利用Gurobi进行求解**步骤 1**中产生的线性不等式集合，从而获得差分或线性路线。

首先，我们介绍如何生成不等式集合以及构建MILP模型。

5.2.1 构建MILP模型

为了不失一般性，我们以搜索r轮Speck32算法差分特征为例介绍如何生成不等式集合以及如何构建MILP模型。

为了便于理解，我们先构造第i轮的差分传递模型。如图 5所示，我们用Δz_{i-1}与Δz_i分别代表第i轮的输入差分与输出差分。其中，$\Delta z_i = (\Delta z_i^{31}, \ldots, \Delta z_i^0), 0 < i \leq r$。

如果用α_i，β_i与γ_i分别代表第i轮轮函数中模加操作的两个输入差分与输出差分，那么我们可以得到：$\alpha_i = (\Delta z_{i-1}^{22}, \ldots, \Delta z_{i-1}^{16}, \Delta z_{i-1}^{31}, \ldots, \Delta z_{i-1}^{23}), \beta_i =$

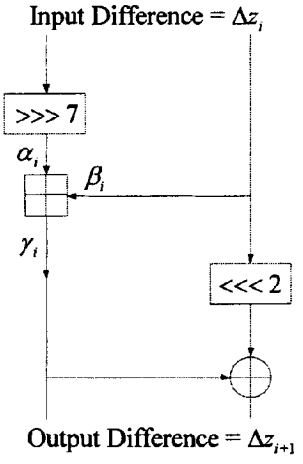

图 5: Speck32算法第i轮差分传递特性

$(\Delta z_{i-1}^{15}, \ldots, \Delta z_{i-1}^{0}), \gamma_i = (\Delta z_i^{31}, \ldots, \Delta z_i^{16})$。接着，我们把$\alpha_i, \beta_i$与$\gamma_i$带入不等式 (2.1)与(3.3)中，可以得到$13 \times (16-1)+1 = 196$个线性不等式。这196个线性不等式可以表示第$i$轮轮函数中模加操作的$(\alpha_i, \beta_i \rightarrow \gamma_i)$的差分传递特性。

对于第i轮轮函数中的异或操作来说，两个输入差分分别是$(\Delta z_{i-1}^{13}, \ldots, \Delta z_{i-1}^{0}, \Delta z_{i-1}^{15}, \Delta z_{i-1}^{14})$与$(\Delta z_i^{31}, \ldots, \Delta z_i^{16})$，输出差分则是$(\Delta z_i^{15}, \ldots, \Delta z_i^{0})$。那么我们有以下的关系：

$$\begin{aligned} \Delta z_{i-1}^{13} \oplus \Delta z_i^{31} &= \Delta z_i^{15}, \\ &\vdots \\ \Delta z_{i-1}^{14} \oplus \Delta z_i^{16} &= \Delta z_i^{0}. \end{aligned}$$

接着，我们根据不等式 (3.2)可以生成$1 \times 16 = 16$个线性不等式。这16个线性不等式可以完全刻画第i 轮中两输入异或操作的差分特性。综上所述，我们可以利用生成的$196 + 16 = 212$个线性不等式来完全刻画$(\Delta z_{i-1} \rightarrow \Delta z_i)$的差分传递特性。

同样的，我们可以用类似的方法生成$212 \cdot r$个线性不等式来刻画r轮的差分传递特性：$(\Delta z_0 \rightarrow \Delta z_1 \rightarrow \ldots \rightarrow \Delta z_r)$。此外，我们还需要加一个额外的

条件来确保明文不会出现零差分：$\sum_{j=0}^{31} \Delta z_0^j > 0$。

至于目标函数，我们把求解$\sum_{i=1}^{r} \sum_{j=0}^{32-2} \neg eq(\alpha_i[j], \beta_i[j], \gamma_i[j])$的最小值设为目标函数。并把上述生成的$212 \cdot r + 1$个线性不等式作为限制条件，将其转换成Gurobi可读的LP格式。最后，我们可以用Gurobi求解器来寻找r轮Speck32算法最好的差分特征。

基于第4章提出的模型，我们可以用类似的方法来构建自动化搜索Speck算法线性逼近的MILP模型。限于篇幅限制，我们就不一一赘述。

我们把详细的代码公布在https://github.com/fukai6/milp_speck.git上。利用提供的代码，我们可以生成描述r轮Speck算法差分特征或线性逼近的线性不等式集合，并将其转成Gurobi可读的LP格式。然后，我们可以利用Gurobi求解器来进行r轮Speck算法差分特征或线性逼近的自动化搜索。

5.2.2 启发式拼接技术

当我们搜索分组长度大于48比特的Speck算法的差分特征或线性逼近时，由于计算资源的限制，我们采用一种启发式拼接技术来加快搜索过程。启发式拼接搜索技术就是独立的搜索若干个短轮数的路线，然将这几条路线拼成一条长路线。举个例子，如图6所示，我们可以通过搜索输出差分为δ的r_1轮的差分特征与输入差分为δ的r_2 轮的差分特征来拼接出一条（$r_1 + r_2$）轮的差分特征。

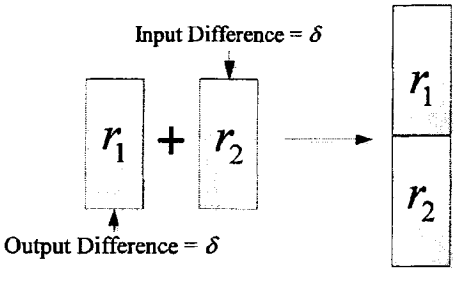

图 6: 启发式拼接技术

基于对实验过程中搜索出的Speck算法短轮数差分特征的观察，以及参考文献[4]提供的差分特征，我们发现当δ的左半部分为$0x80$，右半部分为0时，差分概率可能会比较好一些。于是，我们手动选择不同的r_1与r_2的参数值，以及设置$\delta = 0x80\|0$分别作为两条短路线的输入差分与输出差分，来拼接一条（$r_1 + r_2$）轮的差分特征。在实际搜索过程中，我们设置的参数具体如下：进行Speck64差分特征自动化搜索时，设置δ=0x80 || 0x00000000，$r_1 = 11, r_2 = 4$。进行Speck96差分特征自动化搜索时，设置δ=0x80 || 0x000000000000，$r_1 = 11, r_2 = 5$。对Speck128差分特征自动化搜索时，情况比较特殊。限于计算资源，我们设置$r_1 = 9, r_2 = 9$得到一条概率为2^{-126}的18轮差分特征。为了寻找到更好的差分特征，我们首先设置$r_1 = 9, r_2 = 4$搜索到一条13轮差分特征。然后在9轮前扩展了6轮，得到表6中概率为2^{-119}的（$6 + 9 + 4 = 19$）轮差分特征。同理对于线性逼近，基于实验观察，我们感觉当δ的左半部分为$0x1$，右半部分为0时，线性相关系数可能会比较好一些。在实际的自动化搜索中，我们设置的参数如下：对于Speck64线性逼近自动化搜索，设置δ=0x1 || 0x00000000，r_1=10，r_2=3。对于Speck96线性逼近自动化搜索，设置δ=0x1 || 0x000000000000，r_1=3，r_2=12。对于Speck128线性逼近自动化搜索，设置δ=0x1 || 0x0000000000000000，r_1=6，r_2=10。

接下来，我们解释如何在模型加入限制条件来设置输入差分（掩码）或者输出差分（掩码）为δ。为了不是一般性，我们以在r_2轮Speck32算法加入输入差分$\delta = 0x80\|0x0$为例。假设明文差分$\Delta z_0 = (\Delta z_0^{31}, \Delta z_0^{30}, ..., \Delta z_0^1, \Delta z_0^0)$。为加入限制条件输入差分$\delta = 0x80\|0x0$，只在模型中需加入$\sum_{j=0}^{31} \Delta z_0^j = 1$与$\Delta z_0^{23} = 1$即可。加入输出差分（掩码）步骤类似，我们就不一一赘述。接着，我们可以用Gurobi等MILP求解器来进行ARX算法差分特征与线性逼近的自动化搜索。

5.2.3 实验结果

利用新提出的算法，我们搜索了Speck所有版本算法的差分特征与线性逼近，并把搜索得到的实验过程中最好的差分特征与线性逼近分别列在表 4, 表 5, 表 6, 表 7, 表 8 与表 9 中。这里$\sum_{i=1}^{r} \log_2 p_i$与$\sum_{i=1}^{r} \log_2 c_i$分别代表每一轮的差分特征的概率与线性逼近的相关系数的绝对值。

为了更好的与已有的搜索算法相比较，我们在表2中，总结了Speck算法差分特征与线性逼近的结果。我们改进了Speck48，Speck64，Speck96与Speck-128算法的差分特征与线性逼近。至于搜索程序的运行时间，对于Speck32加密算法，我们利用个人电脑（配置为4核，Intel(R) Core(TM) i5 CPU 650 @3.20GHz）进行搜索，运行时间大约为几个小时。对Speck算法其他版本，我们用实验室IBM服务器（配置为64核，Intel(R)Xeon(R) CPU E7330, 2.40GHz）进行搜索，运行时间大约为1天。我们在这里特别指出，我们完全搜索了Speck32算法的差分特征与线性逼近。对于Speck其他版本算法，我们只是尽力搜索出优于已知结果的路线，并不能保证搜索结果是最好的。通过搜索结果以及运行时间相比较，我们可以得出新提出的技术在基于独立的条件下搜索大分组ARX算法差分特征与线性逼近具有一定的优势。

此外，为了检验假设输入独立与轮与轮之间独立是否对Speck算法差分特征与线性逼近产生影响以及验证新模型是不是适用于实际攻击工作，我们对表 4, 5, 6 中列出的差分特征进行验证。因为差分路线轮数过长，完全验证是不切实际的。于是，我们根据差分概率把路线分成几部分概率较大的差分片段。我们分别对每一片段进行验证。具体的验证过程如下：首先随机生成大量密钥，然后随机产生足够多的明文对，加密相对应的轮数，计算差分特征的实际概率，最后与理论值进行比较并进行分析。通过以上方法，我们计算了每一个短轮数的差分片段，并与理论结果进行比较。具体的结果列在表 10中。在表 10中，第一列代表测试的密码算法，第二轮代表差分片段的轮数，第三列则是理论上的差分片段的差分概率，第四列则是用于验证程序所需要的明文对个数，第五列则是测试密钥量，最后一列则代表着满足实际概率大于等于理论概率的密钥个数。特此说明，对于Speck128，由于前几轮的差分概率太低，无法进行实验测试，我们只能验证从12轮到19轮这7轮差分特征。通过表 10，我们可以看出对于大部分算法的差分特征“好的”密钥个数所占比例都在50%以上，这说明我们的新模型是具有实际参考价值的，是可以应用于实际分析的。但是对于某些差分特征“好的”密钥个数略低于了平均值。这是假设Speck算法中模加操作输入与轮与轮之间相互独立的原因所导致的。这些偏差也会相应的对攻击者的成功率产生一定的影响。

§5.3 Speck算法的密钥恢复攻击

在SAC2014上，Dinur提出一种针对Speck算法密钥恢复攻击的通用技术[9]。与传统密钥恢复攻击不同，这种新技术并没有使用计数技术，而是采用guess-and-determine技术。新技术不仅可以大大提高差分攻击的轮数，还可以大大降低对Speck算法的攻击的时间复杂度。

新技术的主要思路如下：对于Speck$2n/mn$算法，如果找到一条概率为p的$r-1$轮的差分特征。那么攻击者可以在差分路线头部往前扩展一轮，差分路线尾巴往后增加m轮。这样，我们总共可以攻击$r+m$轮。可以往前扩一轮的原因是，第一轮的子密钥以异或的形式介入轮函数，对输入差分不产生任何影响。所以，我们不需要猜测第一轮的子密钥。往后加m轮的原因是每一轮介入n比特密钥，一共只有$m \times n$比特主密钥。

在最后，攻击者可以通过边猜边筛除错误密钥来降低时间复杂度，最后用$2 \cdot p^{-1}$选择明文来恢复$r+m$ 轮Speck$2n/mn$ 算法的mn 比特密钥。攻击所需要的时间复杂度是$2 \cdot p^{-1} \cdot 2^{(m-2)n}$，存储复杂度是$2^{22}$字节。

利用新搜索出的差分特征，我们可以改进已有的密钥恢复攻击。因为攻击步骤与文献[9]中相同，限于篇幅限制，我们就省去了细节部分。我们在表1总结了我们的攻击结果，并已有结果进行对比。通过对比发现，我们改进了Speck48，Speck64，Speck96与Speck128的攻击结果，而且就轮数而言是已有的最优结果。

§5.4 小结

本章中，我们将新提出基于MILP的自动化搜索技术应用到Speck算法。在假设模加操作的输入以及轮与轮之间相互独立的条件下，我们给出了Speck算法的MILP模型，并利用Gurobi搜索了其所有版本的差分特征与线性逼近（具体的代码挂在https://github.com/fukai6/milp_speck.git）。为了缩短搜索时间，我们提出一种启发式拼接技术，即将分组密码分成若干部分，独立搜索这几部分的差分特征概率（线性逼近相关系数），然后将这几部分组合成一条长的差分（线性）路线。我们把具体结果总结在表2中。我们改进了Speck48，Speck64，Speck96与Speck128算法的差分特征与线性逼近。换言之，我们提出的新模型在大分组ARX算法差分特征与线性逼近的

自动化搜索中占有一定的优势。此外，我们还应用Dinur在SAC2014提出的针对Speck算法的攻击技术，提高了Speck48/64/96/128的攻击结果。我们的攻击就轮数而言是已知最优的结果。

表 4: Speck32与Speck48的差分特征

	Speck32			Speck48		
i	Δ_L	Δ_R	$\log_2 p_i$	Δ_L	Δ_R	$\log_2 p_i$
0	0211	0A04		001202	020002	
1	2800	0010	-4	000010	100000	-3
2	0040	0000	-2	000000	800000	-1
3	8000	8000	0	800000	800004	-0
4	8100	8102	-1	808004	808020	-2
5	8004	840E	-3	8400A0	8001A4	-4
6	8532	9508	-8	608DA4	608080	-9
7	5002	0420	-7	042003	002400	-11
8	0080	1000	-3	012020	000020	-5
9	1001	5001	-2	200100	200000	-3
10				202001	202000	-3
11				210020	200021	-4
$\sum_{i=1}^{r} \log_2 p_i$			-30			-45

表 5: Speck64与Speck96的差分特征

	Speck64			Speck96		
i	Δ_L	Δ_R	$\log_2 p_i$	Δ_L	Δ_R	$\log_2 p_i$
0	04092400	20040104		240004000009	010420040000	
1	20000820	20200001	-6	082020000000	000120200000	-6
2	00000009	01000000	-4	000900000000	000001000000	-4
3	08000000	00000000	-2	000008000000	000000000000	-2
4	00080000	00080000	-1	000000080000	000000080000	-1
5	00080800	00480800	-2	000000080800	000000480800	-2
6	00480008	02084008	-4	000000480008	000002084008	-4
7	06080808	164A0848	-7	0800FE080808	0800EE4A0848	-12
8	F2400040	40104200	-13	000772400040	400000104200	-21
9	00820200	00001202	-8	000000820200	000000001202	-11
10	00009000	00000010	-4	000000009000	000000000010	-4
11	00000080	00000000	-2	000000000080	000000000000	-2
12	80000000	80000000	0	800000000000	800000000000	0
13	80800000	80800004	-1	808000000000	808000000004	-1
14	80008004	84008020	-3	800080000004	840080000020	-3
15	808080A0	A08481A4	-5	808080800020	A08480800124	-5
16				800400008124	842004008801	-9
$\sum_r \log_2 p_r$			-62			-87

表 6: Speck128的差分特征

i	Δ_L	Δ_R	$\log_2 p_i$
0	0124000400000000	0801042004000000	
1	0800202000000000	4808012020000000	-6
2	4800010000000000	0840080100000002	-6
3	0808080000000006	4A08480800000016	-7
4	4000400000000032	1042004000000080	-12
5	0202000000000080	8012020000000480	-7
6	0010000000000480	0080100000002084	-5
7	8080000000002080	84808000000124A0	-5
8	0400000000012440	2004000000080144	-9
9	2000000000080220	2020000000480801	-9
10	0000000000480001	0100000002084008	-7
11	000000000E080808	080000001E4A0848	-8
12	00000000F2400040	4000000000104200	-15
13	0000000000820200	0000000000001202	-8
14	0000000000009000	0000000000000010	-4
15	0000000000000080	0000000000000000	-2
16	8000000000000000	8000000000000000	0
17	8080000000000000	8080000000000004	-1
18	8000800000000004	8400800000000020	-3
19	8080808000000020	A084808000000124	-5
$\sum_{i=1}^{r} \log_2 p_i$			-119

表 7: Speck32与Speck48的线性逼近

	Speck32			Speck48		
i	Γ_L	Γ_R	$\log_2 c_i$	Γ_L	Γ_R	$\log_2 c_i$
0	0380	5224		000131	050021	
1	4880	4885	-1	018100	200101	-2
2	20A0	2071	-2	000100	000001	-1
3	40A0	00C1	-2	000001	000000	0
4	0080	4001	-3	0D0000	0C0000	-1
5	0000	0001	0	606100	606C00	-2
6	0004	0004	0	00024A	00620B	-2
7	3810	3010	-1	181040	731042	-4
8	2180	01C0	-3	D812C0	9802D0	-3
9	066A	0608	-2	040600	C4961A	-5
10				2484F2	2480F6	-2
$\sum_{i=1}^{r} \log_2 c_i$	2^{-14}			2^{-22}		

表 8: Speck64与Speck96的线性逼近

	Speck64			Speck96		
i	Γ_L	Γ_R	$\log_2 c_i$	Γ_L	Γ_R	$\log_2 c_i$
0	18600010	10724800		000001000130	040000010021	
1	1B000000	03104000	-3	000000018100	200000000101	-2
2	18000000	18120000	-2	000000000100	000000000001	-1
3	C0000000	C0100000	-1	000000000001	000000000000	0
4	04000006	04800006	-1	0D0000000000	0C0000000000	-1
5	00260030	04200030	-2	604500000000	604C00000000	-2
6	01010501	21013181	-5	00224D000003	006228000003	-4
7	01800126	00018021	-4	181070001018	1B105A680018	-12
8	00018100	20000101	-5	001200000010	180210400000	-6
9	00000100	00000001	-1	101000000000	00101A000000	-3
10	00000001	00000000	0	001800000000	000010000000	-2
11	09800000	08000000	-1	000010000000	000000000000	-1
12	40610000	40680000	-2	000000D00000	000000C00000	-1
13	00024982	00420802	-3	000006041800	000006048000	-2
14				000030003490	000030043080	-3
15				180181910500	800181A10526	-5
$\sum_{i=1}^{r} \log_2 c_i$	2^{-30}			2^{-45}		

表 9: Speck128的线性逼近

i	Γ_L	Γ_R	$\log_2 c_i$
0	0001010000018798	6A800101300601C1	
1	0000018000300720	9400000180300625	-7
2	0000000181818100	200000000181B105	-4
3	0000000001800120	0000000000018021	-3
4	0000000000018100	2000000000000101	-4
5	0000000000000100	0000000000000001	-1
6	0000000000000001	0000000000000000	0
7	0980000000000000	0800000000000000	-1
8	4045000000000000	4048000000000000	-2
9	00224D0000000002	0042280000000002	-4
10	1810600000000010	1A10536C00000010	-9
11	0012000000000080	1002186000000080	-4
12	0010000000000406	8010130000000406	-3
13	3680000000002000	3080180000002004	-3
14	8500000000010181	8524C000000101A1	-4
15	8002000000080001	2106000000080100	-6
16	01301A0000404401	0030180000404801	-3
$\sum_{i=1}^{r} \log_2 c_i$		2^{-58}	

表 10: 对表 4, 5, 6差分特征验证结果

加密算法	分片轮数长度	差分概率	明文对数	密钥总数	“好的”密钥数
Speck32/64	0-9	2^{-30}	2^{32}	7040	3456
Speck48/96	0-6	2^{-19}	2^{22}	10000	4093
Speck48/96	6-11	2^{-26}	2^{30}	6400	2989
Speck64/128	0-6	2^{-19}	2^{22}	10000	5513
Speck64/128	6-8	2^{-20}	2^{24}	10000	4887
Speck64/128	8-15	2^{-23}	2^{26}	10000	4918
Spec96/144	0-5	2^{-15}	2^{18}	10000	5123
Spec96/144	5-7	2^{-16}	2^{20}	10000	5039
Spec96/144	7-8	2^{-21}	2^{24}	10000	5454
Spec96/144	8-11	2^{-17}	2^{20}	10000	5020
Spec96/144	11-16	2^{-18}	2^{22}	10000	4645
Spec128/256	12-19	2^{-23}	2^{26}	10000	4876

第六章 有关模减差分的SMT模型的讨论

近年来，可满足性模理论SMT（satisfiability modulo theories）也逐渐成为自动化搜索领域的焦点。在文献[32]中，Mouha与Preneel提出一种利用SMT求解器STP[31]来搜索ARX算法最优差分路线的新方法，并将此方法应用到Salsa20算法。此外，在CRYPTO2015文章[33]中，Kölbl等人也利用STP等对Simon等算法进行搜索，取得了不错的结果。

由于STP支持多比特的运算，我们因此希望构建ARX算法中模减差分的SMT模型。在文献[34]中，Lipmaa等人提出了计算模减差分概率的简便方法。他们指出对于异或操作：$\alpha \oplus \beta = \gamma$（其中$\alpha, \beta$ 与γ 都是n比特的差分），当模减差分概率$adp^{\oplus}$不为0时，必满足$w = 0^*$或者$w = w'(3+5+6)0^*$的形式。这里w的长度为n，且$w[i] = 4 \times \alpha[i] + 2 \times \beta[i] + \gamma[i]$。$w'$则是任意长度的八进制字符串。

为了应用以上性质，我们进行研究，发现：

异或操作： 当异或差分概率不为0时，$\alpha[0] \oplus \beta[0] \oplus \gamma[0] = 0$，且$w[i]$出现3或5或6后，$w[i+1]$则可以为0到7之间的任意数字。为了更好的刻画这一特性，我们引入$n+1$比特的变量d。定义如下：$d[0] = 0$，$d[i+1] = \alpha[i]|\beta[i]|\gamma[i]|d[i]$，其中$0 \le i < n$。那么我们可以得到以下结论，当$adp^{\oplus}$不为0时，$\alpha, \beta, \gamma$与$d$满足以下特性：

$$\alpha[i] \oplus \beta[i] \oplus \gamma[i] \le d[i], \quad 0 \le i < n$$

由于STP支持向量的与运算，或运算与异或运算，我们希望将上述式子简单化。经过化简，我们发现上述关系可以用下面的等式来表示：

$$(\alpha \oplus \beta \oplus \gamma) \& (\neg(d \gg 1)) = 0 \tag{6.1}$$

因此，我们可以用CVC语句ASSERT($h = ((h[n-1:1]@0\text{bin}0)|\alpha|\beta|\gamma)$)来刻画变量$d$（这里$h = d \gg 1$），用ASSERT(BVXOR($\alpha$,BVXOR($\beta, \gamma$)) & $(\sim h) = 0\text{bin}\underbrace{00...0}_{n}$) 来刻画等式6.1。CVC语言为STP可读输入语言，具体可参考http://stp.github.io/cvc-input-language/。

至于移位操作，我们可以参考Biyukov等人在RT-RSA14的文章[4]中的工作:

左移t位:

$$adp^{\ll t}(\alpha \to \beta) = \begin{cases} 1 & if(\beta = \alpha \ll t) \\ 0 & others \end{cases}.$$

那么，我们直接用ASSERT($\beta = \alpha \ll t$)表示$\beta = \alpha \ll t$即可。

右移t位:

$$adp^{\gg t}(\alpha \to \beta) = \begin{cases} 2^{-n}(2^{n-t} - \alpha_L)(2^t - \alpha_R) & \beta = (\alpha \gg t) \\ 2^{-n}\alpha_L(2^t - \alpha_R) & \beta = (\alpha \gg t) - 2^{n-t} \\ 2^{-n}\alpha_R(2^{n-t} - \alpha_L - 1) & \beta = (\alpha \gg t) + 1 \\ 2^{-n}(\alpha_L + 1)\alpha_R & \beta = (\alpha \gg t) - 2^{n-t} + 1 \end{cases}.$$

其中，$\alpha = \alpha_L \times 2^t + \alpha_R$。由于STP支持加法，减法与乘法运算，我们引入两个新的比特变量a与b。然后用等式

$$\begin{cases} a \leq 1 \\ b \leq 1 \\ \beta = (\alpha \gg t) - a \times 2^{n-t} + b \end{cases} \tag{6.2}$$

来刻画右移操作特性。用CVC语言表示即：

- a : BITVECTOR(1);
- b : BITVECTOR(1);
- ASSERT(β = BVSUB(n, BVPLUS(n, (α >> t), 0bin$\underbrace{0..0}_{n-1}$ @b)), (BVMULT(n, 0bin$\underbrace{0..0}_{t-1}$ 1 $\underbrace{0..0}_{n-t}$, 0bin$\underbrace{0..0}_{n-1}$ @a)));

模减差分概率： 对于异或操作的模减差分概率，在文献[34]中，Lipmaa等人提出一种类似异或差分矩阵乘的算法。其中共有8个8×8矩阵，具体请参考文献[34]。我们暂时没有发现与第4章类似的“好”规律，因此暂时无法构建异或操作模减差分概率的SMT模型。但是，他们研究发现，对于异或操作来说，模减差分概率满足：

$$adp^{\oplus}(w) = adp^{\oplus}(w0*).$$

因此为了便于实现，我们暂时用变量d的汉明重量作为衡量异或操作模减差分概率的一个指标。d的汉明重量越小，模减差分概率可能会越大。对于左移移位操作，我们无需考虑其概率。对于右移移位操作，由于其情况较多，模减差分概率较复杂，我们暂时无法给出一个较好的SMT模型。为了简便实现，我们忽略右移位的模减差分概率，只考虑异或操作的模减差分概率。因此，我们目的在于搜索汉明重量较小的d。

至此，我们构建出ARX算法的模减差分的SMT模型，并可以用求解工具STP进行搜索求解。

值得注意的是，我们依旧假设ARX算法异或操作输入独立，并假设轮与轮之间相互独立。然而，对于某些ARX算法来说，他们并不是独立的。这样可能导致在某些特定密钥下，我们搜索得到的结果与理论值有一定的偏差。这种偏差相应的会对攻击者的成功率产生一定的影响。而且由于忽略了右移的概率，我们的模型是启发式的。我们必须验证搜索出的路线，并重新计算其模减差分概率。

第七章 总结及未来的工作

在本文中，我们通过研究模加操作差分与线性的特性，在假设输入与轮与轮之间相互独立的情况下，提出了一种基于MILP模型自动化搜索ARX加密算法差分特征与线性逼近的方法。接着我们利用新方法对Speck算法进行差分特征与线性逼近的搜索，并得到了较好的结果。与已有的最好的差分特征相比较，我们的结果如下：Speck64路线提高一轮，Speck96路线提高三轮，Speck128路线提高五轮，Speck48概率更优。利用找到的差分特征，我们改进了已有的对着四个版本Speck算法的攻击结果。此外，我们还搜索了Speck算法的线性逼近。与已有结果相比较，我们改进了Speck48，Speck64，Speck96与Speck128的线性逼近。此外，我们还研究了异或操作与移位操作的模减差分特性，在假设输入与轮与轮直接相互独立的情况下，提出一种基于SMT模型自动化搜索ARX加密算法差分特征的新方法。

接下来的研究工作可以侧重于以下几个方向：

1. 尝试解决MILP新建模型中的独立性问题，使之更符合实际分析。
2. 尝试建立多输入模加操作的MILP模型，使之成为一个通用的搜索工具。
3. 尝试使用其他的MILP求解器，以更好的缩短求解时间。
4. 尝试研究如何构建模减差分概率的SMT模型。
5. 尝试研究如何构建多输入模加操作异或差分概率的SMT模型。

参考文献

[1] Abed, F., List, E., Lucks, S., Wenzel, J.: Differential Cryptanalysis of Round-Reduced SIMON and SPECK. FSE 2014. LNCS, vol. 8540, pp. 525-545. Springer, 2015.

[2] Beaulieu, R., Shors, D., Smith, J., Treatman-Clark, S., Weeks, B., Wingers, L.: The SIMON and SPECK Famillies of Lightweight Block Ciphers. Cryptology ePrint archive, Report 2013/543 (2013). http://eprint.iacr.org/

[3] Biham, E., Shamir, A.: Differential Cryptanalysis of DES-like Cryptosystems. Journal of Cryptology, 4(1):3-72, 1991.

[4] Biryukov, A., Velichkov, V.: Automatic Search for Differential Trails in ARX Ciphers. CT-RSA 2014. LNCS, vol. 8366, pp. 227-250. Springer, 2014.

[5] Biryukov, A., Velichkov, V., Corre, Y.: Automatic Search for the Best Trails in ARX: Application to Block Cipher Speck. FSE 2016, Bochum, Germany, March 20-23(to appear), 2016

[6] Biryukov, A., Roy, A., Velichkov, V.: Differential Analysis of Block Ciphers SIMON and SPECK. FSE 2014. LNCS, vol. 8540, pp. 546-570. Springer, 2015.

[7] Biryukov, A., Velichkov, V., Corre, Y. L.: Automatic Search for the Best Trails in ARX: Application to Block Cipher Speck. FSE 2016. LNCS. Springer, to Appear.

[8] CPLEX, I, I.: IBM software group. User-Manual CPLEX. 2011

[9] Dinur, I.: Improved Differential Cryptanalysis of Round-Reduced Speck. SAC 2014. LNCS, vol. 8781, pp. 147-164. Springer, 2014.

[10] De Cannière, C., Rechberger, C.: Finding SHA-1 Characteristics: General Results and Applications. ASIACRYPT 2006. LNCS, vol. 4284, pp. 1 - 20. Springer, 2006.

[11] Gurobi Optimization. Gurobi Optimizer Reference Manual. 2013. http://www.gurobi.com

[12] Leurent, G.: Construction of Differential Characteristics in ARX Designs - Application to Skein. Cryptology ePrint Archive, 2012/668(2012). https://eprint.iacr.org/

[13] Lipmaa, H., Moriai, S.: Efficient Algorithms for Computing Differential Properties of Addition. FSE 2002. LNCS, vol. 2355, pp. 336-350. Springer, 2002.

[14] Liu, M., Chen, J.: Improved Linear Attacks on the Chinese Block Cipher Standard. JCST, vol. 29, Issue 6, pp. 1123-1133. Springer, 2014.

[15] Matsui, M.: Linear Cryptanalysis Method for DES Cipher. EUROCRYPT 1993. LNCS, vol.765, pp. 386-397. Springer, 2001.

[16] Matsui, M.: On Correlation between the Order of S-boxes and the Strength of DES. EUROCRYPT 94. LNCS, vol. 950, pp. 366-375. Springer, 2006.

[17] Mendel, F., Nad, T., Schläffer, M.: Finding SHA-2 Characteristics: Searching through a Minefield of Contradictions. ASIACRYPT 2011. LNCS, vol. 7073, pp. 288 - 307. Springer, 2011.

[18] Mouha, N., Wang, Q., Gu, D., Preneel, B.: Differential and Linear Cryptanalysis Using Mixed-Integer Linear Programming. Inscrypt 2011. LNCS, vol. 7537, pp. 57-76. Springer, 2012.

[19] Nyberg, K., Wallen, J.: Improved Linear Distinguishers for SNOW 2.0. FSE 2006. LNCS, vol. 4047, pp. 144-162. Springer, 2006.

[20] Stein, W.,et al.: Sage: Open Source Mathematical Software(2008).

[21] Sun, S., Hu, L., Song, L., Xie, Y., Wang, P.: Automatic Security Evaluation of Block Ciphers with S-bP Structures Against Related-Key Differential Attacks. Inscrypt 2013. LNCS, vol.8567, pp. 39-51. Springer, 2013.

[22] Sun, S., Hu, L., 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. ASIACRYPT 2014. LNCS, vol. 8873, pp. 158-178. Springer, 2014.

[23] Sun, S., Hu, L., 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. Cryptology ePrint Archive, Report 2013/676 (2013). https://eprint.iacr.org/

[24] Sun, S., Hu, L., Wang, M., Wang, P., Qiao, K., Ma, X., Shi, D., Song, L., Fu, K.: Towards Finding the Best Characteristics of Some Bit-oriented Block Ciphers and Automatic Enumeration of (Related-key) Differential and Linear Characteristics with Predefined Properties. Cryptology ePrint Archive, Report 2014/747 (2014). https://eprint.iacr.org/

[25] Wallén, J.: Linear Approximations of Addition Modulo 2^n. FSE 2003. LNCS, vol. 2887, pp. 261-273. Springer, 2003.

[26] Wang, G., Keller, N., Dunkelman, O.: The Delicate Issues of Addition with Respect to XOR Differences. SAC 2007. LNCS, vol. 4786, pp. 212-231. Springer, 2007.

[27] Winnen, L.: Sage S-box Milp Toolkit. http://www.ecrypt.eu.org/tools/sage-s-box-milp-toolkit

[28] Wu, S., Wang, M.: Security Evaluation against Differential Cryptanalysis for Block Cipher Structures. Cryptology ePrint Archive, Report 2011/551 (2011). https://eprint.iacr.org/

[29] Wu, S., Wu, H., Huang, T., Wang, M., Wu, W.: Leaked-state-forgery attack against the authenticated encryption algorithm ALE. ASIACRYPT 2013. LNCS, vol. 8269, pp. 377-404. Springer, 2013.

[30] Yao, Y., Zhang, B., Wu, W.: Automatic Search for Linear Trails of the SPECK Family. ISC 2015. LNCS, vol. 9290, pp. 158 - 176. Springer, 2015.

[31] Ganesh, V., Dill, D.: A decision procedure for bit-vectors and arrays. CAV 2007, LNCS, vol. 4590, pp. 519 - 531, Springer, 2007.

[32] Mouha, N., Preneel, B.: Towards finding optimal differential characteristics for arx: Application to salsa20. Cryptology ePrint Archive, Report 2013/328 (2013). https://eprint.iacr.org/

[33] Kölbl, S., Leander, G., Tiessen, T.: Observations on the SIMON Block Cipher Family. CRYPTO 2015. LNCS, vol. 9215, pp. 161-185. Springer, 2015

[34] Lipmaa, H., Wallén, J., Dumas, P.: On the Additive Differential Probability of Exclusive-Or FSE 2004, LNCS, vol. 3017, pp. 317-331. Springer, 2004

[35] Birukov, A., Nikolic, I.: Search for Related-key Differential Characteristics in DES-line Ciphers FSE 2011, LNCS, vol. 6733, pp. 18-34. Springer, 2011

致谢

时光荏苒，岁月如梭，三年的研究生生活就要结束了。这三年的生活给我留下了很多美好的记忆。在这里，我想向每一位关心我和帮助我的人表达最诚挚的感谢。

首先，我要感谢导师王美琴教授，王老师科学严谨的工作态度和忘我的工作热情让我由衷的敬佩。同时，王老师对信息安全有着深刻的理解，尤其在密码分析领域一直走在国际的前沿。她不仅让我们有机会参加著名密码学会议，还让我们参与到密码前沿课题研究工作中，让我们受益匪浅。三年的学习中，王老师悉心教导，让我学到了许多以前从未接触到的知识。王老师言传身教，让我改掉了懒惰拖延的坏习惯。在本论文撰写期间，王老师不仅为我指明了课题研究方向，而且在一些细节方面也给与了很多的指导，让我能够顺利的完成毕业论文的写作。王老师在生活方面也给予了我很多的帮助，尤其在最后找工作期间全力的支持和帮助让我心怀感激。再次对王美琴老师表示由衷的感谢。

其次，我还要感谢王小云老师，王老师执着的科研精神令我敬佩。另外，王老师为我们提供了优越的软硬件条件，让我们能够在一个舒适的环境中学习和工作，再次向王小云老师表示深深的谢意。同时还要感谢实验室的江守礼老师，王明强老师，孙秋梅老师，魏普文老师，王薇老师，张国艳老师和徐进老师，他们在我的三年研究生生活中给予了很多的指导和帮助。

感谢实验室的温隆学长，陈怀凤学长，李艳斌学姐和刘瑜学姐对我学习与生活上的帮助。感谢与我一同度过三年研究生生活的同学们，毕文泉，崔婷婷，丁召杰，郭英华，刘付山，刘文卿，皮若男，滕达，袁龙，宗瑞以及各位学弟学妹，是你们让这三年变得更加丰富多彩；特别感谢孙玲学妹，王绍梅学妹和尤瑞英学妹，感谢你们在百忙之中帮我整理各种材料和处理琐碎杂事。还要感谢宿舍舍友刘骏与田英良，感谢你们在我困难的时候给予帮助。

感谢我的父母，感谢他们一直以来无私的支持和帮助。

最后，再一次向关心我，帮助我的所有老师、同学、亲人、朋友们表示衷心的感谢！

攻读学位期间完成的学术论文目录

1. 第一作者 Kai Fu, Meiqin Wang, Yinghua Guo, Siwei Sun, Lei Hu：MILP-Based Automatic Search Algorithms for Differential and Linear Trails for Speck. FSE 2016, Bochum, Germany, March 20-23(to appear), 2016

2. 第一作者 Kai Fu, Ling Sun, Meiqin Wang: New Integral Attacks on SIMON. Submitted to IET Information Security

3. 第二作者 Ling Sun, Kai Fu, Meiqin Wang: Improved Zero-Correlation Cryptanalysis on SIMON. INSCRYPT 2015. LNCS. vol, 9589, pp 125-143, Springer, 2016

4. 第二作者 Yu Liu, Kai Fu, Wei Wang, Ling Sun, Meiqin Wang: Linear cryptanalysis of reduced-round SPECK. Inf. Process. Lett. 116(3): 259-266 (2016)

5. 合作作者 Siwei Sun, Lei Hu, Meiqin Wang, Peng Wang, Kexin Qiao, Xiaoshuang Ma, Danping Shi, Ling Song, Kai Fu: Constructing Mixed-integer Programming Models whose Feasible Region is Exactly the Set of All Valid Differential Characteristics of SIMON. Cryptology ePrint Archive, Report 2015/122(2015). https://eprint.iacr.org/

6. 合作作者 Siwei Sun, Lei Hu, Meiqin Wang, Peng Wang, Kexin Qiao, Xiaoshuang Ma, Danping Shi, Ling Song, Kai Fu: Towards Finding the Best Characteristics of Some Bit-oriented Block Ciphers and Automatic Enumeration of (Related-key) Differential and Linear Characteristics with Predefined Properties.Cryptology ePrint Archive, Report 2014/747(2014). https://eprint.iacr.org/

7. 合作作者 Siwei Sun, Lei Hu, Peng Wang, Meiqin Wang, Danping Shi, Xiaoshuang Ma, Qianqian Yang, Kai Fu: Mixed Integer Programming Models for Finite Automaton and Its Application to Additive Differential Patterns of Exclusive-Or. Cryptology ePrint Archive, Report 2016/338(2016). https://eprint.iacr.org/

学位论文评阅及答辩情况表

	姓　名	专业技术职　务	是否博导（硕导）	所在单位	总体评价※
论文评阅人	王薇	讲师	硕导	山东大学	优秀
	王高雨	副教授	硕导	华东师范大学	优秀

	姓　名		专业技术职　务	是否博导（硕导）	所在单位
答辩委员会成员	主席	林东岱	教授	博导	中国科学院软件研究所
	委员	王小云	教授	博导	清华大学
		江宇祉	教授	博导	山东大学
		孟宪萌	教授	硕导	山东财经大学
		王明强	教授	博导	数学学院

答辩委员会对论文的总体评价※	优秀	答辩秘书	魏普文	答辩日期	2016.5.23
备注					

※ 优秀为“A”；良好为“B”；合格为“C”；不合格为“D”。