Tools for Cryptanalysis

宋凌

Topics

- What we have known
  - Differential/linear trails
  - Computation of the differential probability or linear correlation

- What we will study
  - Search for good differential/linear trails
  - Aid with various tools

Tools for DC/LC

- Matsui’s algorithm [Mat94]
  - Branch and bound
  - Depth-first search algorithm
- **Mixed-Integer Linear Programming (MILP)[MWGP11]**
- **Boolean Satisfiability (SAT/SMT)**
- Constraint Programming (CP)
- Dedicated tools

MILP-based differential cryptanalysis

Mixed-Integer Linear Program (MILP)

- Linear Programming (LP) is a method to solve optimization problems

$$\min -x_1 + x_2 - 2x_3 + x_4 - x_5$$

Linear objective function

$$\text{subject to} \quad \begin{aligned} x_1 + x_2 &\leq 1 \\ x_1 - 5x_2 + x_3 &\leq 2 \\ 2x_3 + 2x_4 - 4x_5 &\leq 1 \\ x_2 - 2x_4 + x_5 &\leq 0 \\ x &\in \{0,1\}^5 \end{aligned}$$

Constraints in the form of linear inequality

Domain of variables

Mixed-Integer Linear Programming (MILP) allows some of the decision variables to be constrained to **integers** and others to be **non-integers**. Binary variables are common for crypto!

LP vs. MILP

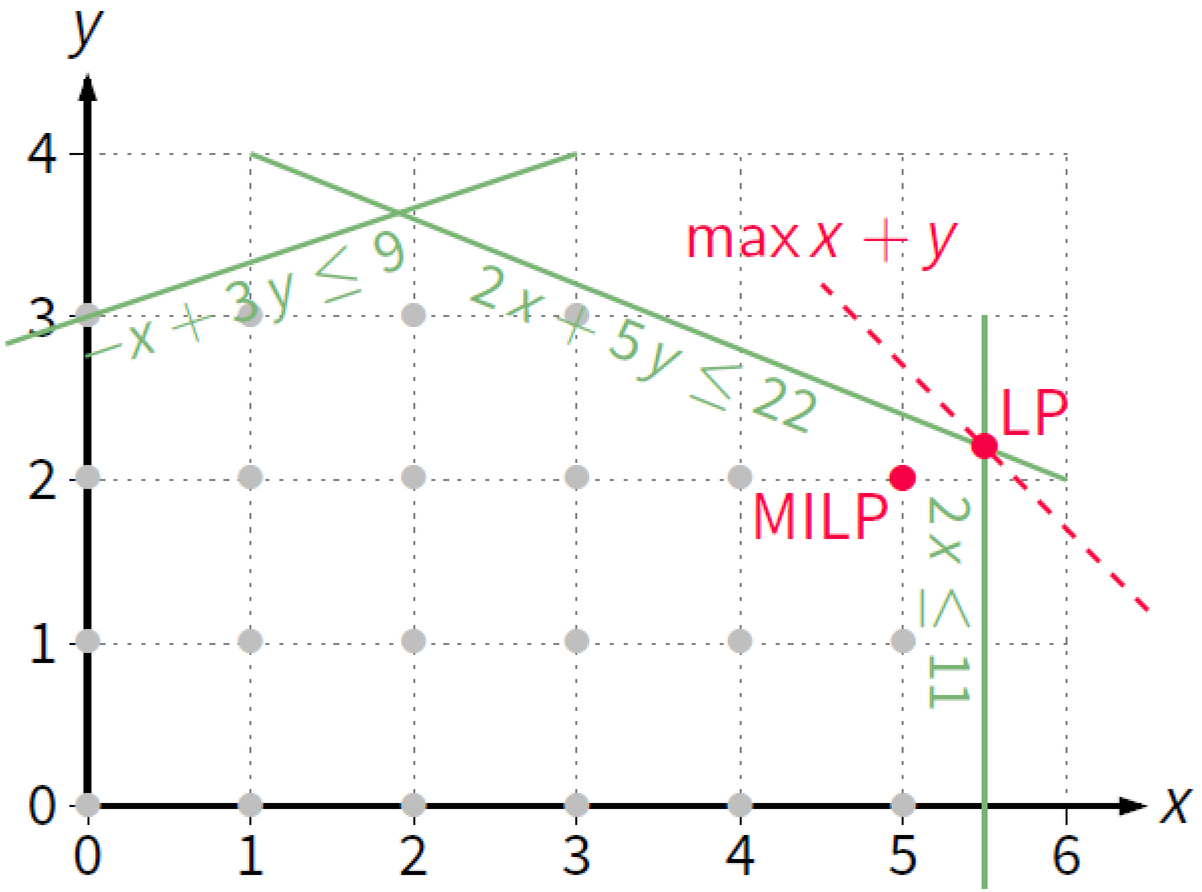


Figure from Maria's slides

MILP solvers

- Hardness of MILP solving:
  - Integer programming is NP-complete.

- Some well-known solver software:
  - CPLEX
  - **Gurobi**

These solvers can be used as stand-alone sotfware (.lp input files) or as libraries with convenient interfaces (C/C++, python Sagemath, ...).

MILP-based differential cryptanalysis

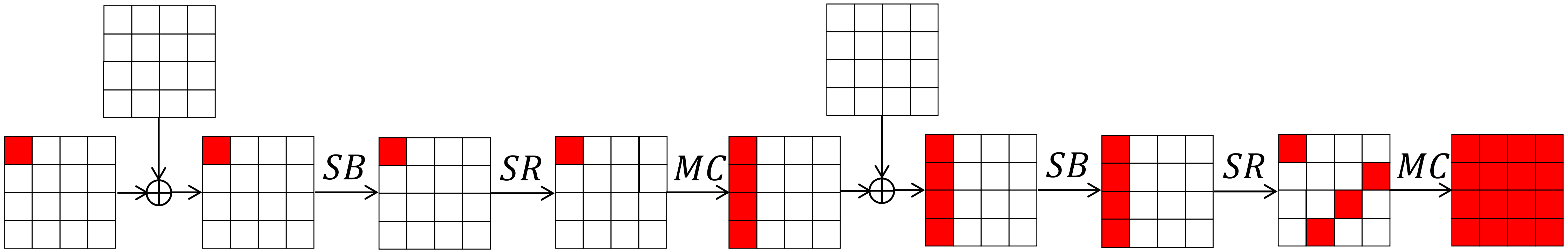
- ☐ **Counting the number of active S-boxes (#AS) of 4-round AES**
- ☐ Search for best differential trails

Counting #AS of 4-round AES

- By the wide trail strategy, it is proved that there are at least 25 active S-boxes in 4-round AES.

- Let us prove it again with MILP.

Differential propagation of AES



Assign a binary variable $S_{r,i,j}$ for each state byte:

$$S_{r,i,j} = 1 \text{ if it is active, otherwise } S_{r,i,j} = 0.$$

- **AK**: input = output
- **SB**: input = output, cost = $\sum S_{r,i,j}$
- **SR**: reorder variables
- **MC**: w(input) + w(output) ≥ 5 (the branch number)

Modeling of AES

Variables:

- $S_{r,i,j} \in \{0,1\}$: Is the S-box in row i column j in round r active?
- $M_{r,j} \in \{0,1\}$: Is the column j of MC in round r active?
- #variables: 16*4 + 4*4 = 80, # inequality: **(1 + 8)*4*3 + 1 = 109**

Model:

$$\min \sum_{r,i,j} S_{r,i,j}$$ % Find the minimal #AS

s.t.

$$\sum_i S_{r,i,(i+j)\%4} + \sum_i S_{r+1,i,j} \geq 5 \cdot M_{r,j}$$ % For each MixColumns

$$M_{r,j} \geq S_{r,i,(i+j)\%4}, M_{r,j} \geq S_{r+1,i,j} \text{ for } i \in [0,3]$$

$$\sum_{i,j} S_{0,i,j} \geq 1$$ % At least one active byte in the input

$$S_{r,i,j}, M_{r,j} \in \{0,1\} \text{ for } r, i, j \in [0,3]$$ % Domain

Modeling of AES

Variables:

- $S_{r,i,j} \in \{0,1\}$: Is the S-box in row i column j in round r active?
- $M_{r,j} \in \{0,1\}$: Is the column j of MC in round r active?
- #variables: 16*4 + 4*4 = 80, # inequality: **2*4*3 + 1 = 25**

Model:

$$\min \sum_{r,i,j} S_{r,i,j}$$ % Find the minimal #AS

s.t. $$5 \cdot M_{r,j} \leq \sum_i S_{r,i,(i+j)\%4} + \sum_i S_{r+1,i,j} \leq 8 \cdot M_{r,j}$$ % For each MixColumns

$$\sum_{i,j} S_{0,i,j} \geq 1$$ % At least one active byte in the input

$S_{r,i,j}, M_{r,j} \in \{0,1\}$ for $r, i, j \in [0,3]$ % Domain

A note on how we model a simple crypto problem

- Construct the model for a specific operation by hand.
- The validity can be verified.
- #variables and #inequality may vary when a different modeling is used.

Goal:

Model a problem with a minimal number of variables and inequality.

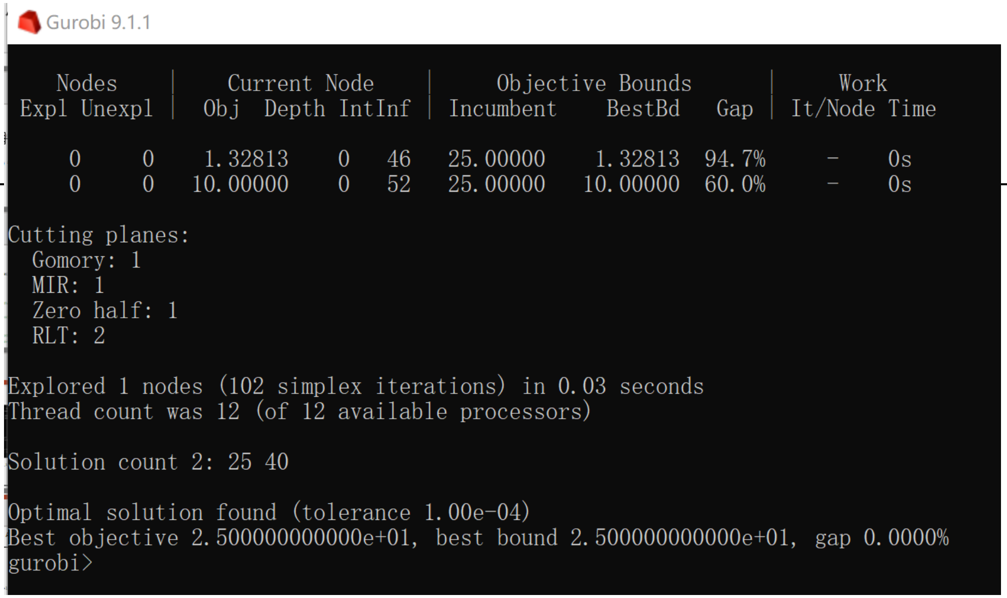
(The solving time is not necessarily reduced when #variables and #inequality are minimal.)

Modeling of AES and solve with Gurobi

AES4r.lp

```
Minimize
S_r0_0_0 + S_r0_0_1 + ...+ S_r3_3_3
Subject To
S_r0_0_0 + S_r0_1_1 + S_r0_2_2 + S_r0_3_3 + S_r1_0_0 + S_r1_1_0 + S_r1_2_0 + S_r1_3_0 - 5 M_r0_0 >= 0
S_r0_0_0 + S_r0_1_1 + S_r0_2_2 + S_r0_3_3 + S_r1_0_0 + S_r1_1_0 + S_r1_2_0 + S_r1_3_0 - 8 M_r0_0 <= 0
......
S_r0_0_0 + S_r0_0_1 + S_r0_0_2 + S_r0_0_3 + S_r0_1_0 + S_r0_1_1 + S_r0_1_2 + S_r0_1_3 + S_r0_2_0 + S_r0_2_1 + S_r0_2_2 +
S_r0_2_3 + S_r0_3_0 + S_r0_3_1 + S_r0_3_2 + S_r0_3_3 >= 1
Binary
S_r0_0_0
......
```

```
gurobi> m = read("AES4r.lp")
gurobi> m.optimize()
gurobi> m.write("filename")
```



```
Gurobi 9.1.1

    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

     0     0    1.32813    0   46   25.00000    1.32813  94.7%     -    0s
     0     0   10.00000    0   52   25.00000   10.00000  60.0%     -    0s

Cutting planes:
  Gomory: 1
  MIR: 1
  Zero half: 1
  RLT: 2

Explored 1 nodes (102 simplex iterations) in 0.03 seconds
Thread count was 12 (of 12 available processors)

Solution count 2: 25 40

Optimal solution found (tolerance 1.00e-04)
Best objective 2.500000000000e+01, best bound 2.500000000000e+01, gap 0.0000%
gurobi>
```

Modeling of AES and solve with Sagemath

```
#!/ usr/bin /env sage
rounds = range (4)
p = MixedIntegerLinearProgram ( maximization = False )
S = p. new_variable ( name ='sbox', binary = True )
M = p. new_variable ( name ='mcol', binary = True )
for r in rounds:
        for j in [0..3]:
                activecells = sum(S[r,i ,(i+j )%4] for i in [0..3]) + sum (S[r+1,i,j] for i in [0..3])
                p. add_constraint (5*M[r,j] <= activecells <= 8*M[r,j])
p.add_constraint(sum(S[0,i,j] for i in [0..3] for j in [0..3]) >= 1)
p.set_objective(=sum(S[r,i,j] for r in rounds for i in [0..3] for j in [0..3]))
p.solve ()
print(p. get_objective_value(), p. get_values(S))
```

https://doc.sagemath.org/html/en/reference/numerical/sage/numerical/mip.html

MILP Example: counting #AS of 4-round AES

```
sage: #!/ usr/bin /env sage
....: rounds = range (4)
....: p = MixedIntegerLinearProgram ( maximization = False )
....: S = p.new_variable( name ='sbox', binary = True )
....: M = p.new_variable( name ='mcol', binary = True )
....: for r in rounds:
....: ^Ifor j in [0..3]:
....: ^I^Iactivecells = sum(S[r,i ,(i+j )%4] for i in [0..3]) + sum (S[r+1,i,j] for i in [0..3])
....: ^I^Ip. add_constraint (5*M[r,j] <= activecells <= 8*M[r,j])
....: p.add_constraint(sum(S[0,i,j] for i in [0..3] for j in [0..3]) >= 1)
....: p.set_objective(sum(S[r,i,j] for r in rounds for i in [0..3] for j in [0..3]))
....: p.solve()
....: print(p.get_objective_value(), p.get_values(S))
....:
25.0
25.0 {(0, 0, 0): 1.0, (0, 1, 1): 1.0, (0, 2, 2): 1.0, (0, 3, 3): 1.0, (1, 0, 0): 0.0, (1, 1, 0): 0.0, (1, 2, 0):
 0.0, (1, 3, 0): 1.0, (0, 0, 1): 0.0, (0, 1, 2): 0.0, (0, 2, 3): 0.0, (0, 3, 0): 0.0, (1, 0, 1): 0.0, (1, 1, 1):
 0.0, (1, 2, 1): 0.0, (1, 3, 1): 0.0, (0, 0, 2): 0.0, (0, 1, 3): 0.0, (0, 2, 0): 0.0, (0, 3, 1): 0.0, (1, 0, 2):
 0.0, (1, 1, 2): 0.0, (1, 2, 2): 0.0, (1, 3, 2): 0.0, (0, 0, 3): 0.0, (0, 1, 0): 0.0, (0, 2, 1): 0.0, (0, 3, 2):
 0.0, (1, 0, 3): 0.0, (1, 1, 3): 0.0, (1, 2, 3): 0.0, (1, 3, 3): 0.0, (2, 0, 0): 0.0, (2, 1, 0): 0.0, (2, 2, 0):
 0.0, (2, 3, 0): 0.0, (2, 0, 1): 1.0, (2, 1, 1): 1.0, (2, 2, 1): 1.0, (2, 3, 1): 1.0, (2, 0, 2): 0.0, (2, 1, 2):
 0.0, (2, 2, 2): 0.0, (2, 3, 2): 0.0, (2, 0, 3): 0.0, (2, 1, 3): 0.0, (2, 2, 3): 0.0, (2, 3, 3): 0.0, (3, 0, 0):
 1.0, (3, 1, 0): 1.0, (3, 2, 0): 1.0, (3, 3, 0): 1.0, (3, 0, 1): 1.0, (3, 1, 1): 1.0, (3, 2, 1): 1.0, (3, 3, 1):
 1.0, (3, 0, 2): 1.0, (3, 1, 2): 1.0, (3, 2, 2): 1.0, (3, 3, 2): 1.0, (3, 0, 3): 1.0, (3, 1, 3): 1.0, (3, 2, 3):
 1.0, (3, 3, 3): 1.0, (4, 0, 0): 1.0, (4, 1, 0): 0.0, (4, 2, 0): 0.0, (4, 3, 0): 0.0, (4, 0, 1): 1.0, (4, 1, 1):
 0.0, (4, 2, 1): 0.0, (4, 3, 1): 0.0, (4, 0, 2): 1.0, (4, 1, 2): 0.0, (4, 2, 2): 0.0, (4, 3, 2): 0.0, (4, 0, 3):
 1.0, (4, 1, 3): 0.0, (4, 2, 3): 0.0, (4, 3, 3): 0.0}
sage:
```

MILP-based differential cryptanalysis

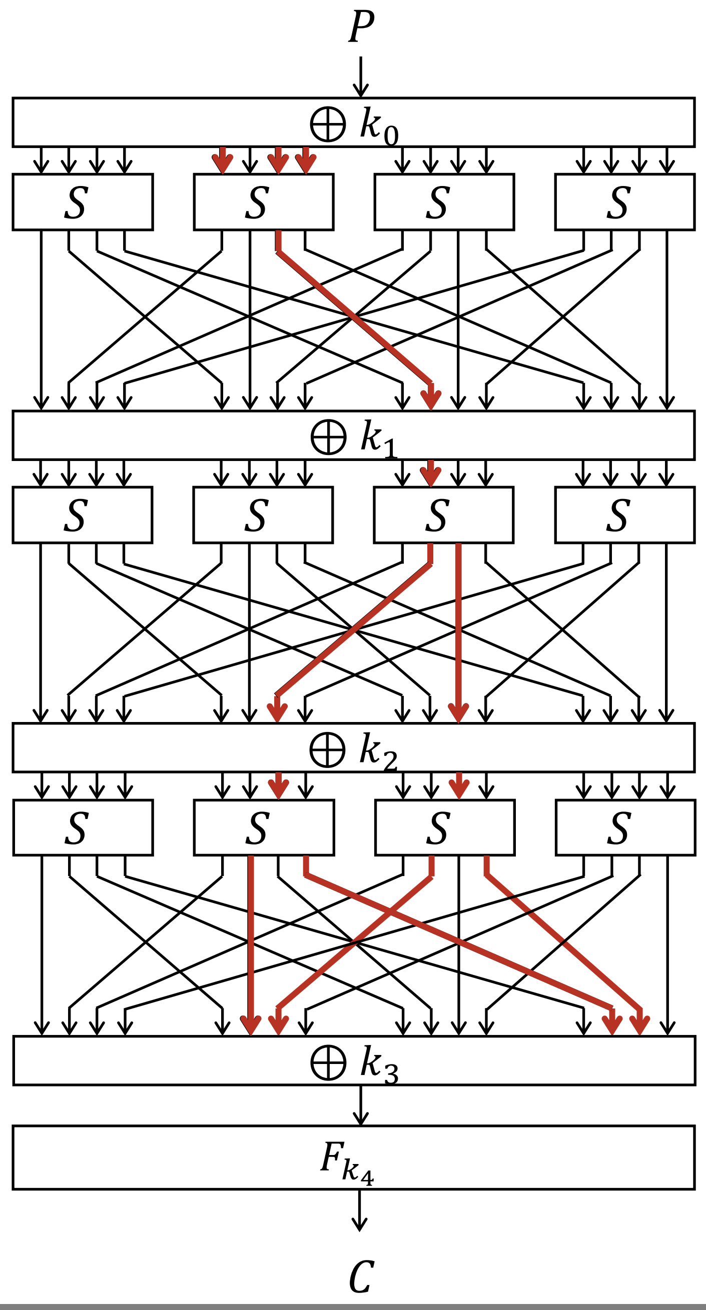
- Counting the number of active S-boxes (#AS) of 4-round AES
- **Search for best differential trails**

Come back to the toy cipher

S = SBox([14, 4, 13, 1, 2, 15, 11, 8, 3, 10, 6, 12, 5, 9, 0, 7])

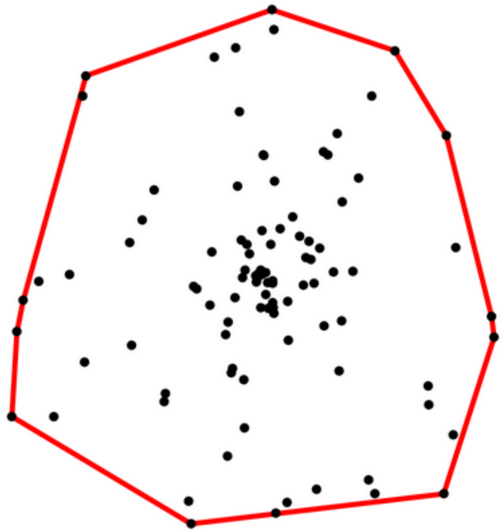
- This cipher uses a bitwise linear layer.
- Cannot treat the S-box as identity.

- How to give a bitwise model for an S-box?
  1. **Convex hull computation [SHW+14]**
  2. **Logical computation [SHW+14, ST17]**



1. Convex hull computation

- Convex hull of a set of points in $\mathbb{R}^n$: the smallest convex set that contains these points.
  - ✓A convex hull can be represented by a set of linear inequalities



- Treat the set of all possible differential patterns of an S-box as a set of points in $\mathbb{R}^n$.

- Then we can compute the linear inequalities representation of the set of differential patterns.

1. Convex hull computation

Collect the set of all possible differentials

$$[x_0, x_1, x_2, x_3, y_0, y_1, y_2, y_3]$$

1. $0 \rightarrow 0 \Rightarrow [0,0,0,0,\ 0,0,0,0]$
2. $1 \rightarrow 3 \Rightarrow [1,0,0,0,\ 1,1,0,0]$
3. $1 \rightarrow 7 \Rightarrow [1,0,0,0,\ 1,1,1,0]$
4. $1 \rightarrow 9 \Rightarrow [1,0,0,0,\ 1,0,0,1]$

……

Input Difference	Output Difference 0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	2	0	0	0	2	0	2	4	0	4	2	0	0
2	0	0	0	2	0	6	2	2	0	2	0	0	0	0	2	0
3	0	0	2	0	2	0	0	0	0	4	2	0	2	0	0	4
4	0	0	0	2	0	0	6	0	0	2	0	4	2	0	0	0
5	0	4	0	0	0	2	2	0	0	0	4	0	2	0	0	2
6	0	0	0	4	0	4	0	0	0	0	0	0	2	2	2	2
7	0	0	2	2	2	0	2	0	0	2	2	0	0	0	0	4
8	0	0	0	0	0	0	2	2	0	0	0	4	0	4	2	2
9	0	2	0	0	2	0	0	4	2	0	2	2	2	0	0	0
A	0	2	2	0	0	0	0	0	6	0	0	2	0	0	4	0
B	0	0	8	0	0	2	0	2	0	0	0	0	0	2	0	2
C	0	2	0	0	2	2	2	0	0	0	0	2	0	6	0	0
D	0	4	0	0	0	0	0	4	2	0	2	0	2	0	2	0
E	0	0	2	4	2	0	0	0	6	0	0	0	0	0	2	0
F	0	2	0	0	6	0	0	0	0	4	0	2	0	0	2	0

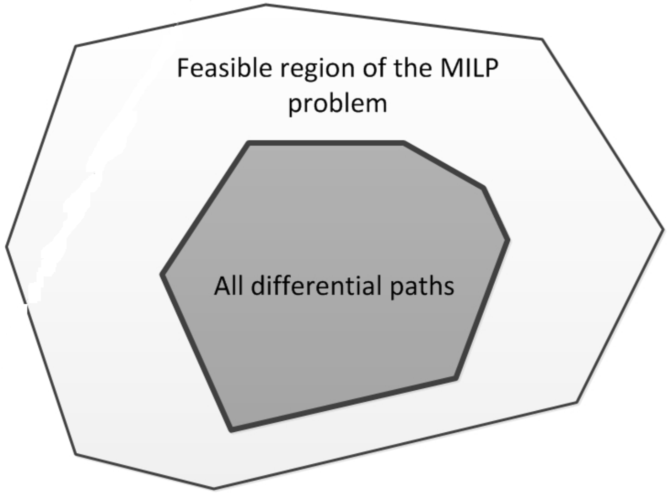
```
sage: Pattern_list = [[0,0,0,0, 0,0,0,0], [0,0,0,1, 0,0,1,1], [0,0,0,1, 0,1,1,1], [0,0,0,1, 1,0,0,1], …]
....: A_polyhedron = Polyhedron(Pattern_list)
....: for v in A_polyhedron.inequality_generator():
....:     print(v)
```

Return 410 inequalities

```
An inequality (0, -1, 0, 0, 0, 0, 0, 0) x + 1 >= 0
An inequality (-1, 0, 0, 0, 0, 0, 0, 0) x + 1 >= 0
An inequality (0, 0, -1, 0, 0, 0, 0, 0) x + 1 >= 0
An inequality (0, -1, -1, -1, -1, 0, -1, 0) x + 4 >= 0
An inequality (0, 0, 0, -1, 0, 0, 0, 0) x + 1 >= 0
An inequality (-1, -1, -1, -1, 1, 0, 1, 0) x + 3 >= 0
An inequality (0, 0, 0, 0, 0, 0, 1, 0) x + 0 >= 0
An inequality (0, 0, 0, 0, -1, 0, 0, 0) x + 1 >= 0
An inequality (-1, 1, -1, -1, -1, 0, 0, -1) x + 4 >= 0
An inequality (0, 0, 0, 0, 0, 0, 0, -1) x + 1 >= 0
An inequality (-1, 1, -1, 0, -1, 0, 1, -1) x + 3 >= 0
An inequality (-1, -1, 0, -1, 1, 0, 1, -1) x + 3 >= 0
An inequality (1, -1, -1, 0, -1, 0, 1, -1) x + 3 >= 0
An inequality (-1, -1, 1, -1, -1, 0, 1, 0) x + 3 >= 0
```

1. Convex hull computation

- Too many inequalities, which will make the MILP problem too difficult to be solved in practical time
  - ✓There are redundant inequalities.
  - ✓Can we use fewer inequalities? **Yes!**



Return 410 inequalities

```
An inequality (0, -1, 0, 0, 0, 0, 0, 0) x + 1 >= 0
An inequality (-1, 0, 0, 0, 0, 0, 0, 0) x + 1 >= 0
An inequality (0, 0, -1, 0, 0, 0, 0, 0) x + 1 >= 0
An inequality (0, -1, -1, -1, -1, 0, -1, 0) x + 4 >= 0
An inequality (0, 0, 0, -1, 0, 0, 0, 0) x + 1 >= 0
An inequality (-1, -1, -1, -1, 1, 0, 1, 0) x + 3 >= 0
An inequality (0, 0, 0, 0, 0, 0, 1, 0) x + 0 >= 0
An inequality (0, 0, 0, 0, -1, 0, 0, 0) x + 1 >= 0
An inequality (-1, 1, -1, -1, -1, 0, 0, -1) x + 4 >= 0
An inequality (0, 0, 0, 0, 0, 0, 0, -1) x + 1 >= 0
An inequality (-1, 1, -1, 0, -1, 0, 1, -1) x + 3 >= 0
An inequality (-1, -1, 0, -1, 1, 0, 1, -1) x + 3 >= 0
An inequality (1, -1, -1, 0, -1, 0, 1, -1) x + 3 >= 0
An inequality (-1, -1, 1, -1, -1, 0, 1, 0) x + 3 >= 0
```

1. Convex hull computation

Select a smaller set of inequalities via a greedy algorithm

Algorithm 1. Selecting n inequalities from the convex hull $\mathcal{H}$ of an S-box

Input: $\mathcal{H}$: the set of all inequalities in the H-representation of the convex hull of an S-box; $\mathcal{X}$: the set of all impossible differential patterns of an S-box; n: a positive integer.

Output: $\mathcal{O}$: a set of n inequalities selected from $\mathcal{H}$

```
l* := None; X* := X; H* := H; O := ∅;
for i ∈ {0, ..., n − 1} do
    l* := The inequality in H* which maximizes the number of removed
          impossible differential patterns from X* ;
    X* := X* − {removed impossible differential patterns by l*};
    H* := H* − {l*}; O := O ∪ {l*};
end
return O
```

H is the set of inequalities return by SAGE

Select the inequality which excludes the largest number of impossible diffentials

1. Convex hull computation

- A set of 25 inequalities can be selected to model the DDT

$[x_0, x_1, x_2, x_3, y_0, y_1, y_2, y_3]$

(-2, -2, 0, 1, 3, 4, 4, 1, 0) ⇨ $-2\,x_0 - 2\,x_1 + 0\,x_2 + x_3 + 3\,y_0 + 4\,y_1 + 4\,y_2 + y_3 \geq 0$

(4, 1, 0, 1, 4, -2, 3, -2, 0)
(3, 4, 1, 3, -2, 1, -2, 1, 0)
(-1, 4, -1, -1, 3, -2, 4, 5, 0)
(2, -3, -1, -3, -2, 1, -1, -1, 8)
(-3, -1, -4, -2, -3, 2, -4, -1, 14)
(-1, -1, 1, 2, -3, -3, 3, -2, 7)
(1, -1, 2, -1, 1, 2, -2, 0, 2)
(-1, 0, 1, 0, 1, -1, -1, 0, 2)
(1, -2, -2, -1, -1, -2, -3, 2, 8)
(1, 1, 1, -2, -2, 3, 3, -2, 3)
(-2, -2, 2, -1, -3, -1, 3, 1, 6)
(2, -2, -2, 3, 1, 3, 3, 1, 0)

(-1, 1, 2, 1, -1, 2, -1, 2, 1)
(2, 1, 2, 3, -2, -1, -1, -2, 3)
(-3, 2, -1, -2, -3, 1, -1, -2, 9)
(3, 3, -1, -1, 3, 0, 2, -1, 0)
(-1, -1, -1, 1, -2, -1, -2, 2, 6)
(-2, -1, -1, 1, -1, 2, -2, -1, 6)
(-1, -1, 0, -1, 1, 0, 1, -1, 3)
(-1, 0, 0, 1, 1, -1, -1, -1, 3)
(-1, 2, -1, 1, 2, 2, 0, 2, 0)
(2, 2, 2, -1, -1, -1, 3, 3, 0)
(0, 0, -1, -1, -1, -1, -1, -1, 5)
(3, -3, 1, -3, -1, -1, -2, 2, 7)

2. Logical computation

Basic Idea: remove all impossible differentials for the S-box

$$V = [x_0, x_1, x_2, x_3, y_0, y_1, y_2, y_3]$$

Eg. $6 \nrightarrow 4 \Rightarrow [0,1,1,0,\ 0,0,1,0]$ ☒

$$x_0 - x_1 - x_2 + x_3 + y_0 + y_1 - y_2 + y_3 \geq -2$$

Let $q(a) = \begin{cases} 1, & a = 0 \\ -1, & a = 1 \end{cases}$

Then to remove $a = [0,1,1,0,\ 0,0,1,0]$ via

$$\sum q(a[i]) \cdot V[i] \geq 1 - H_w(a)$$

Input Difference	Output Difference 0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	2	0	0	0	2	0	2	4	0	4	2	0	0
2	0	0	0	2	0	6	2	2	0	2	0	0	0	0	2	0
3	0	0	2	0	2	0	0	0	0	4	2	0	2	0	0	4
4	0	0	0	2	0	0	6	0	0	2	0	4	2	0	0	0
5	0	4	0	0	0	2	2	0	0	0	4	0	2	0	0	2
6	0	0	0	4	0	4	0	0	0	0	0	0	2	2	2	2
7	0	0	2	2	2	0	2	0	0	2	2	0	0	0	0	4
8	0	0	0	0	0	0	2	2	0	0	0	4	0	4	2	2
9	0	2	0	0	2	0	0	4	2	0	2	2	2	0	0	0
A	0	2	2	0	0	0	0	0	6	0	0	2	0	0	4	0
B	0	0	8	0	0	2	0	2	0	0	0	0	0	2	0	2
C	0	2	0	0	2	2	2	0	0	0	0	2	0	6	0	0
D	0	4	0	0	0	0	0	4	2	0	2	0	2	0	2	0
E	0	0	2	4	2	0	0	0	6	0	0	0	0	0	2	0
F	0	2	0	0	6	0	0	0	0	4	0	2	0	0	2	0

Changing any bit of a will increase the value of RHS, meaning that any other vector does not violate this inequality.

2. Logical computation

Basic Idea: remove all impossible differentials for the S-box

$$V = [x_0, x_1, x_2, x_3, y_0, y_1, y_2, y_3]$$

Eg. $6 \nrightarrow 4 \;\Rightarrow [0,1,1,0,\ 0,0,1,0]$ ☒

$$x_0 - x_1 - x_2 + x_3 + y_0 + y_1 - y_2 + y_3 \geq -2$$

$6 \nrightarrow 6 \;\Rightarrow [0,1,1,0,\ 0,1,1,0]$ ☒

$$x_0 - x_1 - x_2 + x_3 + y_0 - y_1 - y_2 + y_3 \geq -3$$

However, these two cases can be represented as

$[0,1,1,0,\ 0,*,1,0]$ ☒

And removed by

$$x_0 - x_1 - x_2 + x_3 + y_0 - y_2 + y_3 \geq -2$$

Input Difference \ Output Difference	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	2	0	0	0	2	0	2	4	0	4	2	0	0
2	0	0	0	2	0	6	2	2	0	2	0	0	0	0	2	0
3	0	0	2	0	2	0	0	0	0	4	2	0	2	0	0	4
4	0	0	0	2	0	0	6	0	0	2	0	4	2	0	0	0
5	0	4	0	0	0	2	2	0	0	0	4	0	2	0	0	2
6	0	0	0	4	0	4	0	0	0	0	0	0	2	2	2	2
7	0	0	2	2	2	0	2	0	0	2	2	0	0	0	0	4
8	0	0	0	0	0	0	2	2	0	0	0	4	0	4	2	2
9	0	2	0	0	2	0	0	4	2	0	2	2	2	0	0	0
A	0	2	2	0	0	0	0	0	6	0	0	2	0	0	4	0
B	0	0	8	0	0	2	0	2	0	0	0	0	0	2	0	2
C	0	2	0	0	2	2	2	0	0	0	0	2	0	6	0	0
D	0	4	0	0	0	0	0	4	2	0	2	0	2	0	2	0
E	0	0	2	4	2	0	0	0	6	0	0	0	0	0	2	0
F	0	2	0	0	6	0	0	0	0	4	0	2	0	0	2	0

Full model

Variables:

- $S_{r,i} \in \{0,1\}$: Is the S-box i in round r active?
- $x_{r,i,j} \in \{0,1\}$: Is the j-th bit of S-box i in round r active?
- #variables: 16*(R+1) + 4*R, # inequality: **2*4*R + 4*R*T, T = 25**

$\min \sum_{r,i} S_{r,i}$ % Find the minimal #AS

s.t. $S_{r,i} \leq \sum_j x_{r,i,j} \leq 4 \cdot S_{r,i}$ % For each S-box

25 inequalities for input-output patterns

$\sum_i S_{0,i} \geq 1$ % At least one active S-box in the input

$S_{r,i}, x_{r,i,j} \in \{0,1\}$ for $r \in [0,R)$, i, j ∈[0,3] % Domain

Find a minimal set [ST17]

- The greedy algorithm does not guarantee a minimal set to be returned.
- Minimize the number of inequalities via MILP

  Introduce variables z_i to denote whether inequality i is selected or not.

$$\text{minimize} \quad \sum_{i=1}^{N} z_i.$$

$$
\begin{aligned}
z_2 + z_8 + z_N &\geq 1, && \text{as the constaint for } R_0,\\
z_2 + z_3 + z_7 &\geq 1, && \text{as the constaint for } R_1,\\
&\vdots && \vdots\\
z_1 + z_3 + z_4 + z_9 &\geq 1, && \text{as the constaint for } R_{|\mathcal{R}|-1}.
\end{aligned}
$$

Impossible patterns which should be removed

	Patterns in $\mathcal{R}$											
	R_0	R_1	R_2	R_3	R_4	R_5	R_6	R_7	R_8	R_9	$\cdots$	$R_{\lvert\mathcal{R}\rvert-1}$
Inequality 1	0	0	0	1	1	0	1	0	0	0	$\cdots$	1
Inequality 2	1	1	0	1	0	0	0	0	0	1	$\cdots$	0
Inequality 3	0	1	1	0	0	1	0	0	1	0	$\cdots$	1
Inequality 4	0	0	0	0	0	0	1	0	0	0	$\cdots$	1
Inequality 5	0	0	0	1	0	0	0	0	1	0	$\cdots$	0
Inequality 6	0	0	1	0	0	0	0	0	0	0	$\cdots$	0
Inequality 7	0	1	0	0	1	1	0	0	0	0	$\cdots$	0
Inequality 8	1	0	0	1	0	0	0	1	0	0	$\cdots$	0
Inequality 9	0	0	1	0	0	0	0	1	0	0	$\cdots$	1
$\vdots$							$\vdots$					
Inequality N	1	0	0	1	0	0	1	0	1	1	$\cdots$	0

Choose from a set with N inequalities

What else MILP can do

- Search for other distinguishers
  - Impossible differential
  - Zero-correlation trail
  - Division trail
  - Boomerang distinguishers
  - Demirci-Selcuk MitM attack
  - Cube attack
  - …

Limitation

- Not so useful for complex bitwise descriptions and characteristics
  - Language of linear inequalities is not so natural for crypto
  - Too many integer variables lead to bad solver performance

SAT-based Cryptanalysis

SAT problem

- The Boolean satisfiability problem (SAT) considers the satisfiability of a given Boolean formula.
- It was shown that the problem is NP-complete. However, modern SAT solvers based on backtracking search can solve problems of practical interest with millions of variables.
- Conjunctive Normal Form (CNF,合取范式)

$$\underbrace{(\overbrace{a}^{\text{literal}})}_{\text{clause}} \wedge \underbrace{(\overbrace{\bar{b}}^{\text{literal}})}_{\text{clause}} \wedge \underbrace{(\overbrace{\bar{a}}^{\text{literal}} \vee \overbrace{b}^{\text{literal}} \vee \overbrace{c}^{\text{literal}})}_{\text{clause}}$$

Product of sums

- EXample solvers：
  - MiniSAT, CryptoMiniSAT, Plingeling,

Model DDT of the S-box

Basic Idea: remove all impossible differentials for the S-box

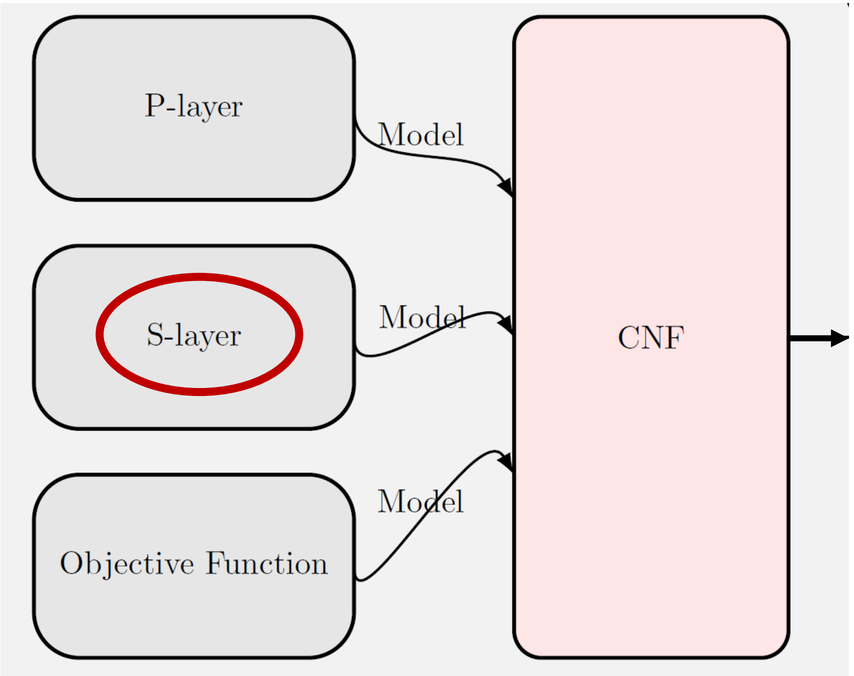
$V = [x_0, x_1, x_2, x_3, y_0, y_1, y_2, y_3]$

Eg. $6 \not\rightarrow 4 \Rightarrow [0,1,1,0,\ 0,0,1,0]$ ☒

$x_0 \vee \bar{x}_1 \vee \bar{x}_2 \vee x_3 \vee y_0 \vee y_1 \vee \bar{y}_2 \vee y_3$ (= true)

That is, remove $a = [0,1,1,0,\ 0,0,1,0]$ via

$$\bigvee (V[i] \oplus a[i])$$ (= true)



Changing any bit of a will make the clause true, meaning that any other vector does not violate this clause.

Model DDT of the S-box

Basic Idea: remove all impossible differentials for the S-box

$$V = [x_0, x_1, x_2, x_3, y_0, y_1, y_2, y_3]$$

Eg. $6 \nrightarrow 4 \Rightarrow [0,1,1,0,\ 0,0,1,0]$ ☒

$$x_0 \vee \bar{x}_1 \vee \bar{x}_2 \vee x_3 \vee y_0 \vee y_1 \vee \bar{y}_2 \vee y_3$$

$6 \nrightarrow 6 \Rightarrow [0,1,1,0,\ 0,1,1,0]$ ☒

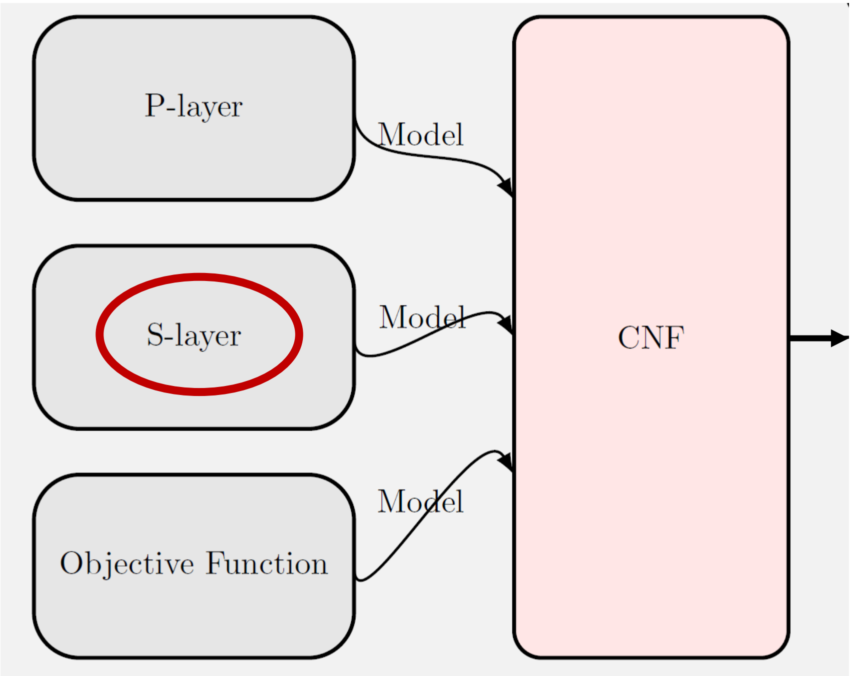
$$x_0 \vee \bar{x}_1 \vee \bar{x}_2 \vee x_3 \vee y_0 \vee \bar{y}_1 \vee \bar{y}_2 \vee y_3$$

However, these two cases can be represented as

$[0,1,1,0,\ 0,*,1,0]$ ☒

And removed by

$$x_0 \vee \bar{x}_1 \vee \bar{x}_2 \vee x_3 \vee y_0 \vee \bar{y}_2 \vee y_3$$



Model DDT of the S-box

Take the activeness into account:

$$V = [x_0, x_1, x_2, x_3, y_0, y_1, y_2, y_3, S_{r,i}]$$

Eg. $6 \nrightarrow 4 \Rightarrow [0,1,1,0,\ 0,0,1,0,\ 0]$ ☒

$[0,1,1,0,\ 0,0,1,0,\ 1]$ ☒

$6 \nrightarrow 6 \Rightarrow [0,1,1,0,\ 0,1,1,0,\ 0]$ ☒

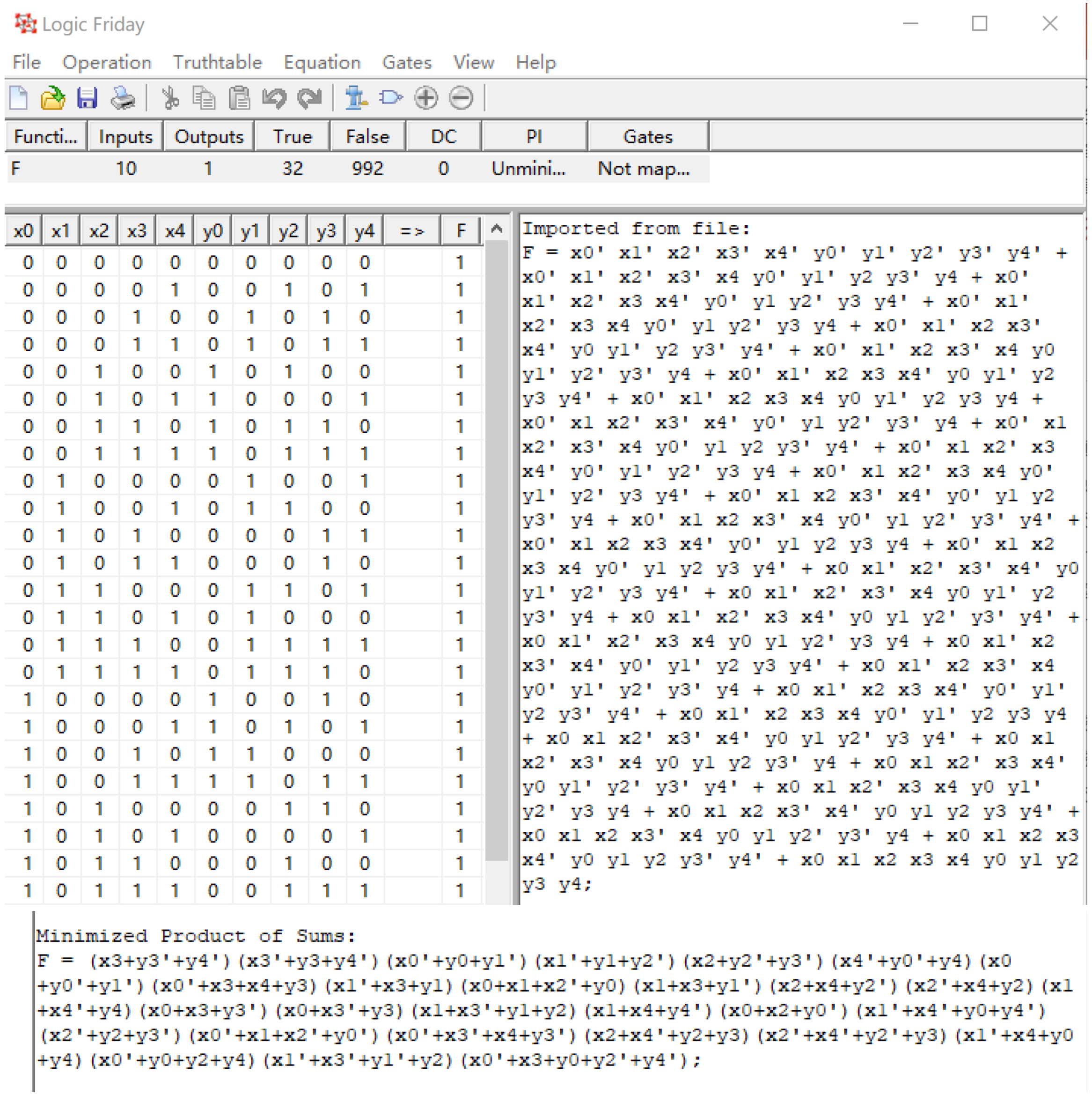
$[0,1,1,0,\ 0,1,1,0,\ 1]$ ☒

Simplify the product of sums

- Quine-McCluskey (QM) algorithm & Espresso algorithm.
- **Software: Logic Friday.**

Use Logic Friday

- Input the truth table of a Boolean function via
  - Typing all the entries by hand
  - Importing from a cvs file
- Click ‘Operation-> minimize’
- Limitation: cannot take more than 16 input variables.



This does not correspond to the S-box of the toy cipher

Convert integer constraints into CNF

- $\sum_{r,i} S_{r,i} \leq w$, i.e., set the number of active S-boxes to $\leq w$.
- Employ the cardinality constraint.
- Cost $2wn + n - 3w - 1$ clauses.
  - when $n = 16, w = 4$ it requires 131 clauses

Cardinality constraint:

$$\sum_{\xi=0}^{\mu-1} p_\xi \leqslant w,\ w \geqslant 1.$$

$$\begin{cases} \overline{p_0} \vee u_{0,0} = 1 \\ \overline{u_{0,j}} = 1 \\ \overline{p_i} \vee u_{i,0} = 1 \\ \overline{u_{i-1,0}} \vee u_{i,0} = 1 \\ \overline{p_i} \vee \overline{u_{i-1,j-1}} \vee u_{i,j} = 1 \\ \overline{u_{i-1,j}} \vee u_{i,j} = 1 \\ \overline{p_i} \vee \overline{u_{i-1,w-1}} = 1 \\ \overline{p_{\mu-1}} \vee \overline{u_{\mu-2,w-1}} = 1 \end{cases}.$$

$u_{i,j}\ (0 \leqslant i \leqslant \mu-2, 0 \leqslant j \leqslant w-1)$

MILP vs. SAT

MILP vs. SAT on modeling DDT

Basic Idea: remove all impossible differentials for the S-box

$V = [x_0, x_1, x_2, x_3, y_0, y_1, y_2, y_3]$

Eg. $6 \nrightarrow 4 \Rightarrow [0,1,1,0,\ 0,0,1,0]$ ☒

$x_0 - x_1 - x_2 + x_3 + y_0 + y_1 - y_2 + y_3 \geq -2$

Changing any bit of a will **increase the value of RHS**, meaning that any other vector does not violate this inequality.

$V = [x_0, x_1, x_2, x_3, y_0, y_1, y_2, y_3]$

Eg. $6 \nrightarrow 4 \Rightarrow [0,1,1,0,\ 0,0,1,0]$ ☒

$x_0 \vee \bar{x}_1 \vee \bar{x}_2 \vee x_3 \vee y_0 \vee y_1 \vee \bar{y}_2 \vee y_3 = 1$

Changing any bit of a will **make the clause true**, meaning that any other vector does not violate this clause.

Obtain minimized MILP models for DDT via the minimized product-of-sums representation.

赛题解读

题目二

Z_2^n上 非空子集线性不等式完全刻画问题

例如，n=3, A={(000),(101),(011),(110)}。我们可以构造一组线性不等式组 L:

$$\begin{cases} x_1 + x_2 \geq x_3 \\ x_1 + x_3 \geq x_2 \\ x_2 + x_3 \geq x_1 \\ x_1 + x_2 + x_3 \leq 2 \end{cases},$$

其由 4 个不等式组成。容易验证，上述线性不等式组 L 关于(x_3,x_2,x_1)的解集恰好为 A。

8个小题

$$Score_{i,j} = \delta_j c_j / l_{i,j}, \qquad c_j = \min\{l_{1,j}, l_{2,j}, \dots, l_{m,j}, r_j\},$$

表 1 每道小题的参数设置

小题序号	n	元素个数
1	6	29
2	8	97
3	10	317
4	12	2017
5	14	6361
6	16	32386
7	20	491144
8	24	8115092

表 2 每道小题权重分值δ_j和不等式个数参考值r_j的取值

小题序号j	δ_j	r_j
1	100	8
2	200	16
3	200	30
4	200	180
5	400	800
6	600	2300
7	800	36000
8	1000	576000

要求：完整刻画，使用的不等式尽可能少。

举例：第一小题

- $n = 6$, 29个元素
- 采用两种方法
  - Convex hull computation + greedy algorithm
  - Logic Friday

q_1_n=6.txt

```
00 09 0B 0D 0F 12 13 16
17 19 1A 1D 1E 24 25 26
27 29 2B 2C 2E 32 33 34
35 39 3A 3C 3F
```

- The number of inequalities is 14. Minimized.

Summary

- MILP-based cryptanalysis
- SAT-based cryptanalysis
- 应用于密码数学挑战赛题目二

- **Questions?**

Reference

[Mat94] Mitsuru Matsui. On Correlation Between the Order of S-boxes and the Strength of DES. Advances in Cryptology – EUROCRYPT ’94. Vol. 950. LNCS. Springer, 1994, pp. 366–375.

[MWGP11] Nicky Mouha, Qingju Wang, Dawu Gu, Bart Preneel: Differential and Linear Cryptanalysis Using Mixed-Integer Linear Programming. Inscrypt 2011: 57-76

[ST17] Yu Sasaki, Yosuke Todo: New Algorithm for Modeling S-box in MILP Based Differential and Division Trail Search. SECITC 2017: 150-165

[SHW+14] Siwei Sun, Lei Hu, Peng Wang, Kexin Qiao, Xiaoshuang Ma, Ling Song: Automatic Security Evaluation and (Related-key) Differential Characteristic Search: Application to SIMON, PRESENT, LBlock, DES(L) and Other Bit-Oriented Block Ciphers. ASIACRYPT (1) 2014: 158-178