# A Very Compact S-box for AES

## *(Advanced Encryption Standard)*

D. Canright

dcanright@nps.edu

Applied Mathematics Dept.

Naval Postgraduate School

Monterey CA 93943, USA

# Advanced Encryption Standard
## (AES)

# Algorithm

- AES is symmetric block cipher
- from 128-bit key, a different round key generated for each of 10 rounds
- each 128-bit block processed by rounds

**round 0 :**

Add Round Key.

**rounds 1-9 :**

S-Box; Shift Rows; Mix Columns; Add Round Key.

**round 10 :**

S-Box; Shift Rows; Add Round Key.

# step1: Add Round Key

for whole 128-bit block:

$$\text{in} \oplus \text{key} \rightarrow \text{out}$$

where $\oplus$ is bitwise exclusive-or (XOR)
(For decryption, inverse operation is identical.)

# step2: S-Box (Byte Substitution)

for each 8-bit byte $\mathbf{a}$:

1. *Inverse*: Let $\mathbf{c} = \mathbf{a}^{-1}$, the inverse in $GF(2^8)$
2. *Affine*: The output $\mathbf{s}$ is $M\,\mathbf{c} \oplus \mathbf{b}$:

$$\begin{pmatrix} s_7 \\ s_6 \\ s_5 \\ s_4 \\ s_3 \\ s_2 \\ s_1 \\ s_0 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} c_7 \\ c_6 \\ c_5 \\ c_4 \\ c_3 \\ c_2 \\ c_1 \\ c_0 \end{pmatrix} \oplus \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}$$

# step3: Shift Rows

for $4 \times 4$ byte matrix, rotate rows 0–3 accordingly:

$$\begin{pmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{pmatrix} \rightarrow \begin{pmatrix} a & b & c & d \\ f & g & h & e \\ k & l & i & j \\ p & m & n & o \end{pmatrix}$$

# step4: Mix Columns

for each 4-byte column $\mathrm{C}$ of $4 \times 4$ byte matrix:

$$\begin{pmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{pmatrix} \begin{pmatrix} C_0 \\ C_1 \\ C_2 \\ C_3 \end{pmatrix} \rightarrow \begin{pmatrix} D_0 \\ D_1 \\ D_2 \\ D_3 \end{pmatrix}$$

where byte multiplication and addition is in $GF(2^8)$

# nonlinearity

- the steps Shift Rows, Mix Columns, & Add Round Key are *linear* operations (and easy)

# nonlinearity

- the steps Shift Rows, Mix Columns, & Add Round Key are *linear* operations (and easy)
- the S-box function is *nonlinear* due to the *inverse* operation in $GF(2^8)$ (not easy to compute)

# Galois Fields

# definition

A *field* is a set with two operations, addition $\oplus$ and multiplication $\otimes$ :

- both satisfy closure
- both associative
- both commutative
- each has identity (0 and 1)
- any element $a$ has additive inverse $-a$
- any nonzero element $a \neq 0$ has multiplicative inverse $a^{-1}$
- multiplication distributive over addition

# finite fields

- A finite field $F$ has $p^n$ elements (prime $p$, integer $n > 0$).
- $F$ has *characteristic* $p$: for any $a \in F$, $a + a + \cdots + a$ ($p$ times) $= 0$.
- Fields with same number of elements are *isomorphic*.
- Over a *subfield* $S \subset F$, of $p^j$ elements with $n = jk$:
  - $F$ is a *vector space* of dimension $k$ over $S$.
  - Each $a \in F$ has *minimal polynomial* of degree $m \leq k$;
  - the $m$ distinct roots $a, a^{p^j}, \ldots$ are *conjugates*, their sum is the *trace* and their product is the *norm* (of $a$).
  - The product of all the minimal polynomials is $x^{p^n} - x$

# $GF(2^8)$ Representation

- *standard*
  - for $GF(2^8)/GF(2)$: $A = a_7x^7 + \cdots + a_1x + a_0$, where $a_i \in \{0, 1\}$ and $x^8 + x^4 + x^3 + x + 1 = 0$.
- *subfield*
  - for $GF(2^8)/GF(2^4)$: $A = a_1x + a_0$ *or* $a_1x_1 + a_0x_0$, where $a_i, T, N \in GF(2^4)$ and $x^2 + Tx + N = 0$;
  - then for $GF(2^4)/GF(2^2)$: $A = a_1x + a_0$ *or* $a_1x_1 + a_0x_0$, where $a_i, T, N \in GF(2^2)$ and $x^2 + Tx + N = 0$;
  - then for $GF(2^2)/GF(2)$: $A = a_1x + a_0$ *or* $a_1x_1 + a_0x_0$, where $a_i \in \{0, 1\}$ and $x^2 + x + 1 = 0$.

(note: $T$ is trace and $N$ is norm, over subfield)

# Implementation

# Implementation Goals

Different applications have different constraints & goals.

**speed :** throughput and/or latency (by parallelism, pipelining)

- Morioka & Satoh, Int'l Conf. Computer Design (2002), *IEEE*
- Weaver & Wawrzynek, (2002)
- Jarvinen et al., FPGA 03 (2003) *ACM*

**low power :** e.g., for smart cards

- Morioka & Satoh, CHES2002 (2003), *LNCS* **2523**

**small size :** for limited ciruitry, e.g., also smart cards

- Rudra et al., CHES2001 (2001), *LNCS* **2162**
- Satoh et al., ASIACRYPT (2001), *LNCS* **2248**
- Wolkerstorfer et al., CT-RSA (2002), *LNCS* **2271**
- Chodowiec & Gaj, CHES2003 (2003), *LNCS* **2779**
- Mentens et al., CT-RSA (2005), *LNCS* **3376**

# small size

- prior smallest: Satoh et al., used nested fields for S-box

# small size

- prior smallest: Satoh et al., used nested fields for S-box
- recent improvement: Mentens et al., considered other isomorphisms (64)

# small size

- prior smallest: Satoh et al., used nested fields for S-box
- recent improvement: Mentens et al., considered other isomorphisms (64)
- current work: more improvement —

# small size

- prior smallest: Satoh et al., used nested fields for S-box
- recent improvement: Mentens et al., considered other isomorphisms (64)
- current work: more improvement —
  - considered more isomorphisms (432), incl. normal bases

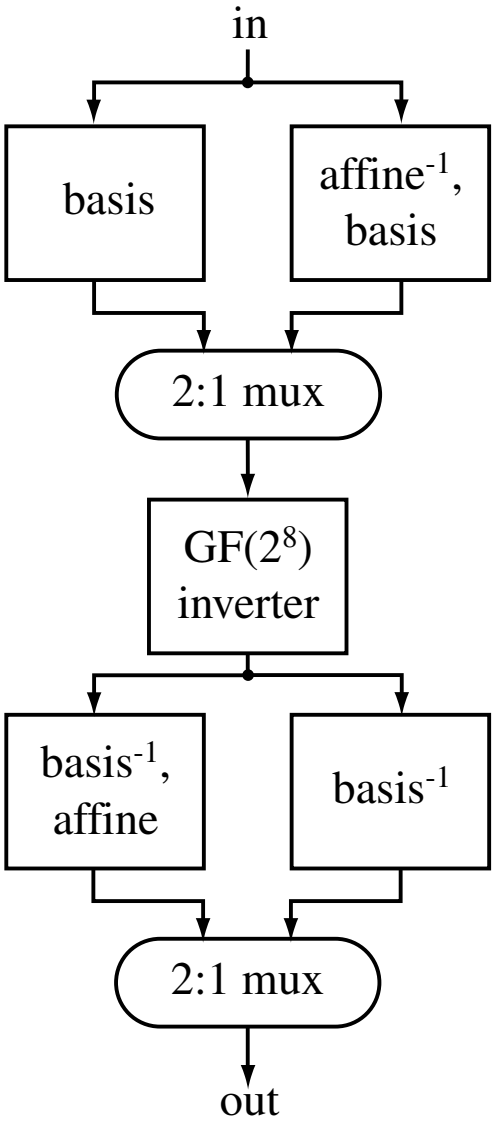
# small size

- prior smallest: Satoh et al., used nested fields for S-box
- recent improvement: Mentens et al., considered other isomorphisms (64)
- current work: more improvement —
  - considered more isomorphisms (432), incl. normal bases
  - fully optimized basis-change matrices

# small size

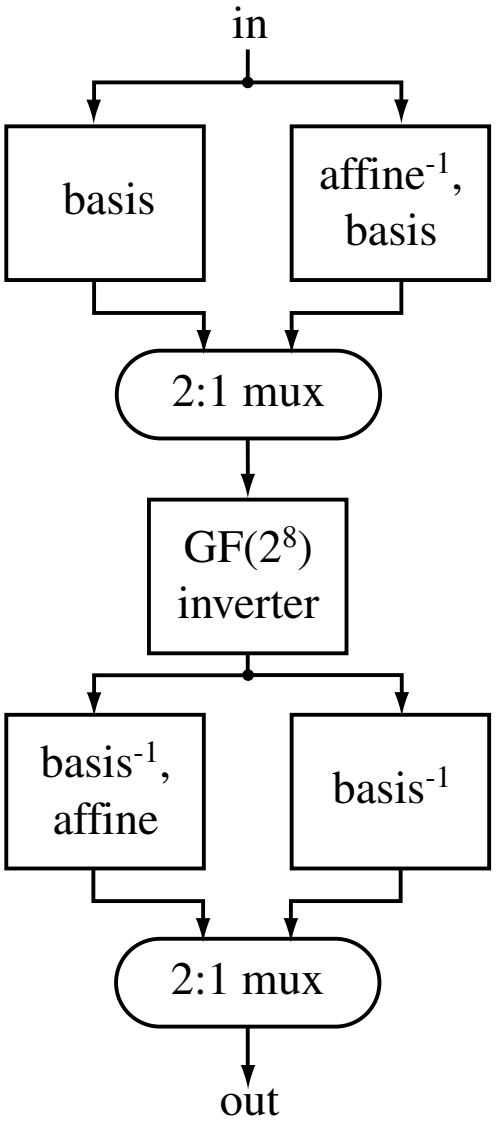
- prior smallest: Satoh et al., used nested fields for S-box
- recent improvement: Mentens et al., considered other isomorphisms (64)
- current work: more improvement —
  - considered more isomorphisms (432), incl. normal bases
  - fully optimized basis-change matrices
  - logic-gate substitution (NOR for NAND and XORs)

# merged S-box, S-box$^{-1}$




- Satoh architecture
  *shares inverter*
  between S-box and S-box$^{-1}$
  (left pathways for encryption
  right pathways for decryption)

# merged S-box, S-box$^{-1}$




- Satoh architecture *shares inverter* between S-box and S-box$^{-1}$ (left pathways for encryption right pathways for decryption)
- This also allows *pairs* of transformations (input and output) to be optimized together

# Main Operations - formulas

using roots of $x^2 + T\,x + N$, where $T$ is trace, $N$ is norm
*polynomial* basis $[x, 1]$ inverse & multiplication:

$$[\Gamma_1, \Gamma_0]^{-1} = (\Gamma_1^2 N + \Gamma_1\Gamma_0 T + \Gamma_0^2)^{-1} \otimes [\ \Gamma_1\ ,\ \Gamma_0 + \Gamma_1 T\ ]$$

$$[\Gamma_1, \Gamma_0] \otimes [\Delta_1, \Delta_0] = [\ \Gamma_1\Delta_0 + \Gamma_0\Delta_1 + \Gamma_1\Delta_1 T\ ,\ \Gamma_0\Delta_0 + \Gamma_1\Delta_1 N\ ]$$

*normal* basis $[x_1, x_2]$ inverse & multiplication:

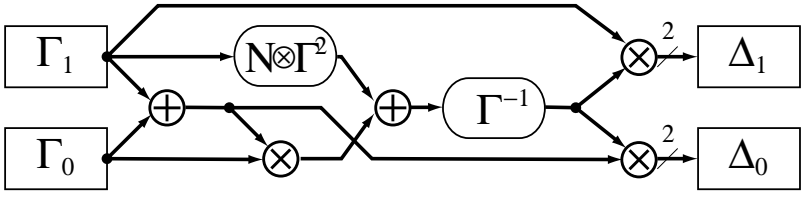
$$[\Gamma_1, \Gamma_0]^{-1} = (\Gamma_1\Gamma_0 T^2 + (\Gamma_1^2 + \Gamma_0^2)N)^{-1} \otimes [\ \Gamma_0,\ \Gamma_1\ ]$$

$$\begin{aligned}[\Gamma_1, \Gamma_0] \otimes [\Delta_1, \Delta_0] \quad = \quad & [\ \Gamma_1\Delta_1 T + (\Gamma_1 + \Gamma_0)(\Delta_1 + \Delta_0)NT^{-1}\ , \\ & \Gamma_0\Delta_0 T + (\Gamma_1 + \Gamma_0)(\Delta_1 + \Delta_0)NT^{-1}\ ]\end{aligned}$$

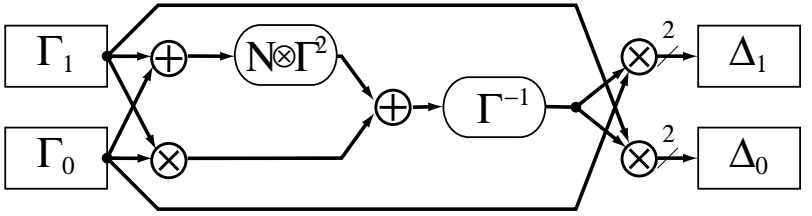
May choose $T = 1$ *or* $N = 1$ *or* $NT^{-1} = 1$; best is $T = 1$.

# Main Operations - diagrams

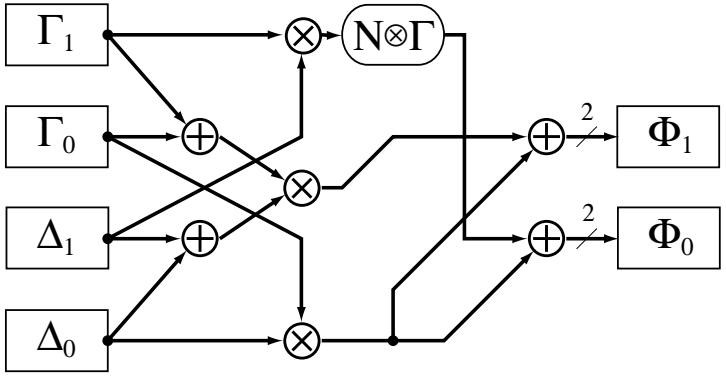
## polynomial inverter



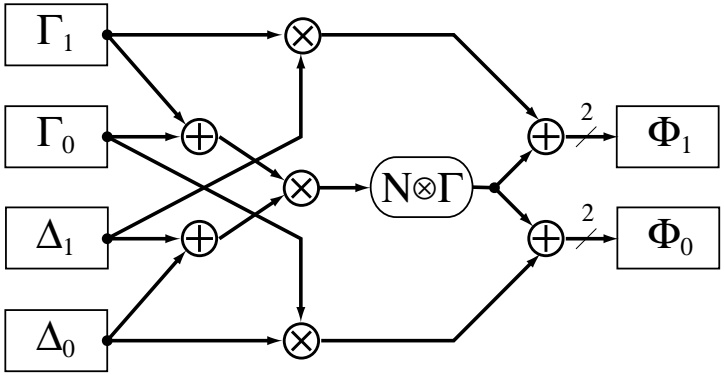

## normal inverter




## polynomial multiplier



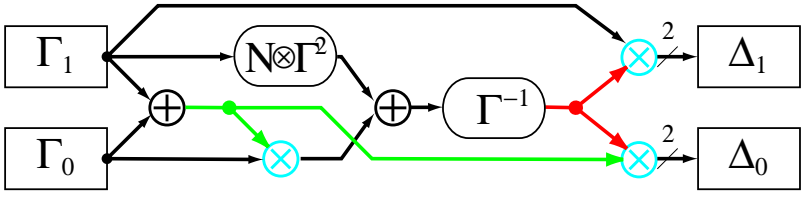

## normal multiplier



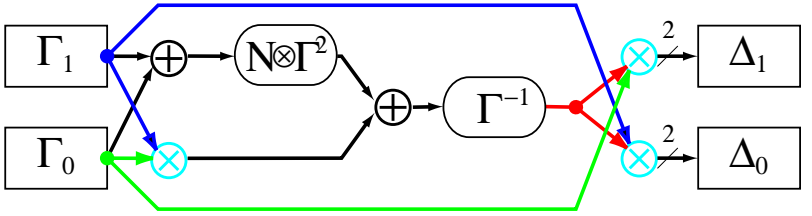

Both polynomial and normal bases require same number and type of subfield operations.

# Main Operations - diagrams

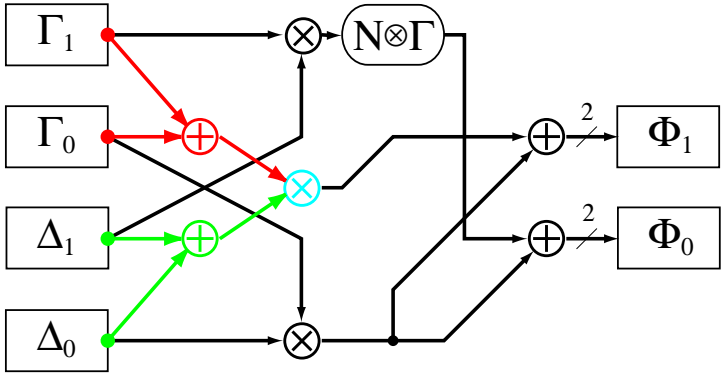
## polynomial inverter



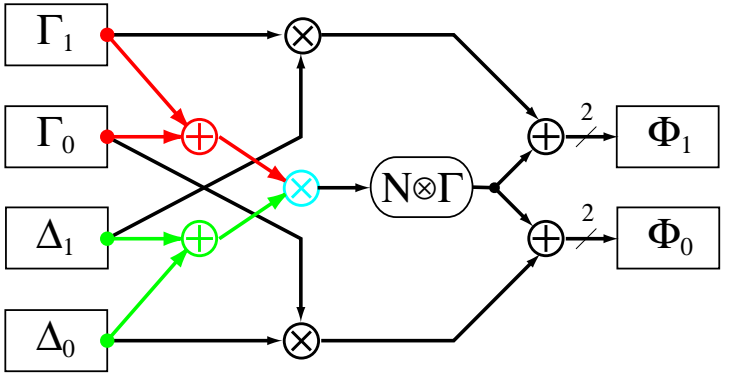

## normal inverter




## polynomial multiplier




## normal multiplier




Both polynomial and normal bases require same number and type of subfield operations.

Note that in normal inverter, each factor to multiplier is shared with another multiplier.

# Optimizations

- factoring transformation matrices

# Optimizations

- factoring transformation matrices
  - prior work: greedy algorithm

# Optimizations

- factoring transformation matrices
  - prior work: greedy algorithm
  - current: full optimization by tree search

# Optimizations

- factoring transformation matrices
  - prior work: greedy algorithm
  - current: full optimization by tree search
- common subexpressions

# Optimizations

- factoring transformation matrices
  - prior work: greedy algorithm
  - current: full optimization by tree search
- common subexpressions
  - shared factors in inverters

# Optimizations

- factoring transformation matrices
  - prior work: greedy algorithm
  - current: full optimization by tree search
- common subexpressions
  - shared factors in inverters
  - bit sums for square&scale

# Optimizations

- factoring transformation matrices
  - prior work: greedy algorithm
  - current: full optimization by tree search
- common subexpressions
  - shared factors in inverters
  - bit sums for square&scale
- logic gate substitution

# Optimizations

- factoring transformation matrices
  - prior work: greedy algorithm
  - current: full optimization by tree search
- common subexpressions
  - shared factors in inverters
  - bit sums for square&scale
- logic gate substitution
  - XNOR for NOT XOR

# Optimizations

- factoring transformation matrices
  - prior work: greedy algorithm
  - current: full optimization by tree search
- common subexpressions
  - shared factors in inverters
  - bit sums for square&scale
- logic gate substitution
  - XNOR for NOT XOR
  - NAND for AND

# Optimizations

- factoring transformation matrices
  - prior work: greedy algorithm
  - current: full optimization by tree search
- common subexpressions
  - shared factors in inverters
  - bit sums for square&scale
- logic gate substitution
  - XNOR for NOT XOR
  - NAND for AND
  - $a$ NOR $b$ for $a$ XOR $b$ XOR $a$ NAND $b$

# Results

# Best Case Results

our smallest implementation of: merged S-box & inverse (Satoh architecture, with shared inverter); S-box alone; and inverse S-box alone.

| best | XOR | NAND | NOR | NOT | MUX | total gates |
|---|---|---|---|---|---|---|
| merged | 94 | 34 | 6 | 2 | 16 | 234 |
| S-box | 80 | 34 | 6 | 0 | 0 | 180 |
| $(\text{S-box})^{-1}$ | 81 | 34 | 6 | 0 | 0 | 182 |

# Best Case Results

our smallest implementation of: merged S-box & inverse (Satoh architecture, with shared inverter); S-box alone; and inverse S-box alone.

| best | XOR | NAND | NOR | NOT | MUX | total gates |
|---|---|---|---|---|---|---|
| merged | 94 | 34 | 6 | 2 | 16 | 234 |
| S-box | 80 | 34 | 6 | 0 | 0 | 180 |
| $(\text{S-box})^{-1}$ | 81 | 34 | 6 | 0 | 0 | 182 |

- 20% smaller than previous smallest merged S-box of Satoh, at 294 gates.

# Best Case Results

our smallest implementation of: merged S-box & inverse (Satoh architecture, with shared inverter); S-box alone; and inverse S-box alone.

| best | XOR | NAND | NOR | NOT | MUX | total gates |
|---|---|---|---|---|---|---|
| merged | 94 | 34 | 6 | 2 | 16 | 234 |
| S-box | 80 | 34 | 6 | 0 | 0 | 180 |
| $(\text{S-box})^{-1}$ | 81 | 34 | 6 | 0 | 0 | 182 |

- 20% smaller than previous smallest merged S-box of Satoh, at 294 gates.
- same basis that gives smallest merged S-box also gives smallest separate S-box.

# Levels of Optimization

Size of $GF(2^8)$ inverter with increasing levels of optimization:

| inverter | XOR | NAND | NOR | total gates |
|---|---|---|---|---|
| hierarchical | 88 | 36 | 0 | 190 |
| w/ shared oper. | 66 | 36 | 0 | 152 |
| w/ NOR subst. | 56 | 34 | 6 | 138 |

# Levels of Optimization

Size of $GF(2^8)$ inverter with increasing levels of optimization:

| inverter | XOR | NAND | NOR | total gates |
|---|---|---|---|---|
| hierarchical | 88 | 36 | 0 | 190 |
| w/ shared oper. | 66 | 36 | 0 | 152 |
| w/ NOR subst. | 56 | 34 | 6 | 138 |

- sharing operations saves 20%.
- the NOR substitution saves an additional 9%.

# Choice of Basis

Comparison of four choices of basis: our best case; best case of Mentens et al.; basis of Satoh et al.; and our worst case.

| basis | merged | S-box | S-box$^{-1}$ |
|---|---|---|---|
| ours | 253 | 195 | 195 |
| Mentens | 271 | 204 | 206 |
| Satoh | 275 | 211 | 209 |
| worst | 293 | 223 | 222 |

- worst-basis merged S-box bigger than best by 16%.

# Matrix Optimization

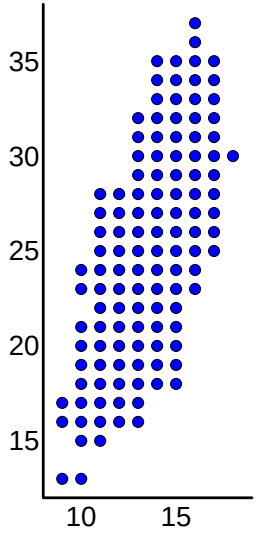
Full optimization of matrices often improves upon the greedy algorithm, but may require much computation.

| matrix size | matrices optimized | # improved by | | | matrices improved |
|---|---|---|---|---|---|
| | | 1 XOR | 2 XORs | 3 XORs | |
| $8 \times 8$ | 1728 | 613 | 138 | 11 | 44% |
| $16 \times 8$ | 55 | 24 | 10 | 6 | 73% |

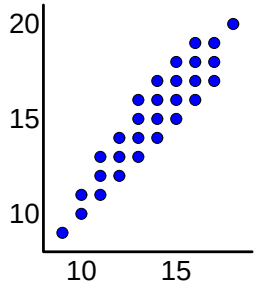
# Matrix Size Predictors

criteria for comparing matrices before optimization:

number of ones vs. opt.




greedy algorithm vs. opt.




comparisons of matrices based on:
‘number of ones’ incorrect for 37% of $8 \times 8$ and 44% of $16 \times 8$
greedy algorithm incorrect for 20% of $8 \times 8$ and 31% of $16 \times 8$

# Conclusions

Several improvements allow the merged S-box architecture of Satoh to be reduced in circuitry:

# Conclusions

Several improvements allow the merged S-box architecture of Satoh to be reduced in circuitry:

- considering other bases for subfields; of 432 cases, best uses all normal bases

# Conclusions

Several improvements allow the merged S-box architecture of Satoh to be reduced in circuitry:

- considering other bases for subfields; of 432 cases, best uses all normal bases
- full matrix optimization improves on the greedy algorithm

# Conclusions

Several improvements allow the merged S-box architecture of Satoh to be reduced in circuitry:

- considering other bases for subfields; of 432 cases, best uses all normal bases
- full matrix optimization improves on the greedy algorithm
- the NOR substitution gives further improvement

# Conclusions

Several improvements allow the merged S-box architecture of Satoh to be reduced in circuitry:

- considering other bases for subfields; of 432 cases, best uses all normal bases
- full matrix optimization improves on the greedy algorithm
- the NOR substitution gives further improvement
- the resulting merged S-box is 20% smaller than that of Satoh

# Conclusions

Several improvements allow the merged S-box architecture of Satoh to be reduced in circuitry:

- considering other bases for subfields; of 432 cases, best uses all normal bases
- full matrix optimization improves on the greedy algorithm
- the NOR substitution gives further improvement
- the resulting merged S-box is 20% smaller than that of Satoh
- this smaller size could save chip area in ASICs, or allow more copies for parallelism and pipelining