Heuristics to minimize the complexity of digital circuits

Andrea Visconti – Università degli Studi di Milano

CIRCUIT MINIMIZATION

The problem of **gate-efficient implementation** is an hard problem.

We are working on the problem of finding "**good**" circuits over GF(2).

"Good" means small, low-depth, few AND gates, and so on.

Logic minimization techniques with **applications to cryptology**.

Andrea Visconti – Dipartimento di Informatica, Università degli Studi di Milano

CIRCUIT MINIMIZATION

- Hardware
  - **Power consumption** presents a critical issue in computing (e.g. mobile platform);
  - **Costly components** (e.g. chips);
  - Implementation of standard cryptographic algorithms (e.g. RFID, smart cards);

- Software
  - Optimizations implemented in **algebraic attacks** (e.g. symmetric ciphers);
  - **High-speed** software;

CIRCUIT MINIMIZATION

- **Non-linear** functions
- **Linear** functions

- Examples:
  - AES S-Box;
  - Present S-Box;
  - GOST S-Box;
  - Multiplication of polynomials of degree n over GF(2);
  - Camelia;
  - Etc.

CIRCUIT MINIMIZATION

"Good" = small;

PROBLEM: To minimize the total number of gates in the boolean circuit implementation of a given function *f*.

Boolean circuits for linear functions can be represented as linear **straight-line programs** (SLPs).

t1 = x3 + x5;

t2 = x0 + x6;

t3 = x0 + x3;

t4 = t2 + x5;

…

CIRCUIT MINIMIZATION

The shortest SLP problem is to **find the shortest linear program** which computes a set of linear functions over a field.

Solving the shortest SLP problem over GF(2) corresponds to finding a gate-optimal Boolean circuit that computes the linear functions.

This problem is known to be MAX SNP-complete;

Unless P=NP, there is no efficient algorithm that can compute even approximately optimal solutions;

CIRCUIT MINIMIZATION

We apply heuristics…

- **polynomial-time heuristics** do quite poorly on random m×n systems of equations;
- **exponential-time heuristics** do significantly better and are fast enough to be used in many practical situations (e.g. cryptographic functions, matrix multiplication);

Circuit Minimization

An example: AES S-Box.

- To compute the inverse of a number in $GF(2^8)$, i.e. $GF(2)[x]/(x^8 + x^4 + x^3 + x + 1)$;
- To apply an affine transformation: $A\mathbf{x}^{-1}$ + const = y;

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

CIRCUIT MINIMIZATION

Tower fields architecture: to build a circuit for inverses in GF(2^{mn}), given a circuit for inverses in GF(2^m);

- Inversion in GF($(2^4)^2$)
- Inversion in GF($((2^2)^2)^2$)

Itoh e Tsujii, Information and Computation,1988

CIRCUIT MINIMIZATION

There are many representations of GF(2^4).

- Polynomial bases
- Normal bases
- Mixed bases

Satoh et al., ASIACRYPT 2001

Canright, CHES 2005

Boyar and Peralta, SEA 2010

Nogami et al., CHES 2010

Boyar and Peralta, SEC 2012

CIRCUIT MINIMIZATION

We try to **optimize linear components…**

Greedy Algorithm

- To make a locally optimal choice for each decision;
- To lead to a globally optimal solution;

Paar, Int. Symp. Information Theory, 1997

Boyar and Peralta, SEA 2010

CIRCUIT MINIMIZATION

AES's S-box consists of:

- a **linear expansion U** (i.e. 8×22 matrix U)
- a **non-linear contraction F** (from 22 to 18 bits)
- a **linear contraction B** (i.e.18×8 matrix B)

$$B \cdot F(U\mathbf{x}) + \text{const} = y$$

In summary,

- we **reduce** the number of **AND gates** (i.e. reduce the multiplicative complexity);
- we **reduce** the number of **XOR gates** (i.e. optimize linear components).

CIRCUIT MINIMIZATION

Straight-line program for AES S-Box

Inputs: X0, ..., X7
Outputs: S0, ..., S7

$y_{14} = x_3 + x_5$	$y_{13} = x_0 + x_6$	$y_9 = x_0 + x_3$
$y_8 = x_0 + x_5$	$t_0 = x_1 + x_2$	$y_1 = t_0 + x_7$
$y_4 = y_1 + x_3$	$y_{12} = y_{13} + y_{14}$	$y_2 = y_1 + x_0$
$y_5 = y_1 + x_6$	$y_3 = y_5 + y_8$	$t_1 = x_4 + y_{12}$
$y_{15} = t_1 + x_5$	$y_{20} = t_1 + x_1$	$y_6 = y_{15} + x_7$
$y_{10} = y_{15} + t_0$	$y_{11} = y_{20} + y_9$	$y_7 = x_7 + y_{11}$
$y_{17} = y_{10} + y_{11}$	$y_{19} = y_{10} + y_8$	$y_{16} = t_0 + y_{11}$
$y_{21} = y_{13} + y_{16}$	$y_{18} = x_0 + y_{16}$	

$t_{46} = z_{15} + z_{16}$	$t_{47} = z_{10} + z_{11}$	$t_{48} = z_5 + z_{13}$
$t_{49} = z_9 + z_{10}$	$t_{50} = z_2 + z_{12}$	$t_{51} = z_2 + z_5$
$t_{52} = z_7 + z_8$	$t_{53} = z_0 + z_3$	$t_{54} = z_6 + z_7$
$t_{55} = z_{16} + z_{17}$	$t_{56} = z_{12} + t_{48}$	$t_{57} = t_{50} + t_{53}$
$t_{58} = z_4 + t_{46}$	$t_{59} = z_3 + t_{54}$	$t_{60} = t_{46} + t_{57}$
$t_{61} = z_{14} + t_{57}$	$t_{62} = t_{52} + t_{58}$	$t_{63} = t_{49} + t_{58}$
$t_{64} = z_4 + t_{59}$	$t_{65} = t_{61} + t_{62}$	$t_{66} = z_1 + t_{63}$
$s_0 = t_{59} + t_{63}$	$s_6 = t_{56}$ XNOR t_{62}	$s_7 = t_{48}$ XNOR t_{60}
$t_{67} = t_{64} + t_{65}$	$s_3 = t_{53} + t_{66}$	$s_4 = t_{51} + t_{66}$
$s_5 = t_{47} + t_{65}$	$s_1 = t_{64}$ XNOR s_3	$s_2 = t_{55}$ XNOR t_{67}

Andrea Visconti – Dipartimento di Informatica, Università degli Studi di Milano

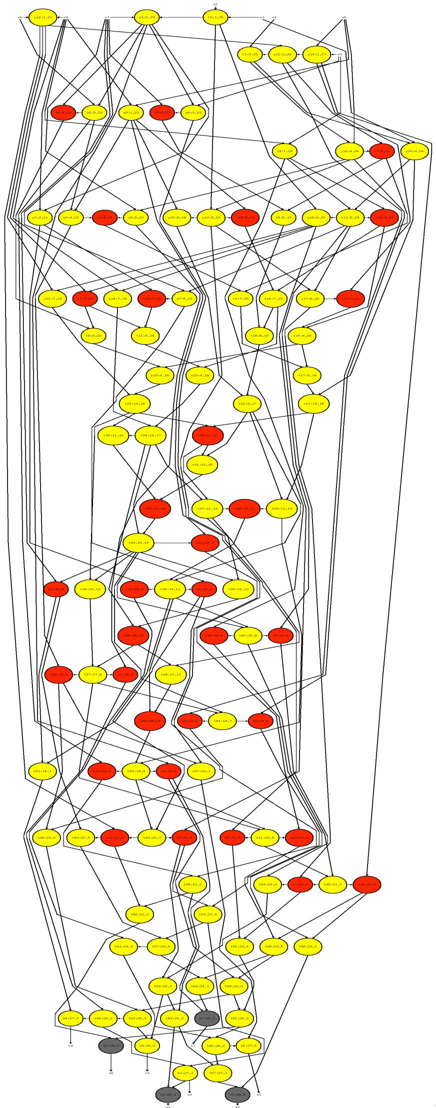
CIRCUIT MINIMIZATION

Inversione in GF(2^4)

$t_2 = y_{12} \times y_{15}$	$t_3 = y_3 \times y_6$	$t_4 = t_3 + t_2$
$t_5 = y_4 \times x_7$	$t_6 = t_5 + t_2$	$t_7 = y_{13} \times y_{16}$
$t_8 = y_5 \times y_1$	$t_9 = t_8 + t_7$	$t_{10} = y_2 \times y_7$
$t_{11} = t_{10} + t_7$	$t_{12} = y_9 \times y_{11}$	$t_{13} = y_{14} \times y_{17}$
$t_{14} = t_{13} + t_{12}$	$t_{15} = y_8 \times y_{10}$	$t_{16} = t_{15} + t_{12}$
$t_{17} = t_4 + t_{14}$	$t_{18} = t_6 + t_{16}$	$t_{19} = t_9 + t_{14}$
$t_{20} = t_{11} + t_{16}$	$t_{21} = t_{17} + y_{20}$	$t_{22} = t_{18} + y_{19}$
$t_{23} = t_{19} + y_{21}$	$t_{24} = t_{20} + y_{18}$	
$t_{25} = t_{21} + t_{22}$	$t_{26} = t_{21} \times t_{23}$	$t_{27} = t_{24} + t_{26}$
$t_{28} = t_{25} \times t_{27}$	$t_{29} = t_{28} + t_{22}$	$t_{30} = t_{23} + t_{24}$
$t_{31} = t_{22} + t_{26}$	$t_{32} = t_{31} \times t_{30}$	$t_{33} = t_{32} + t_{24}$
$t_{34} = t_{23} + t_{33}$	$t_{35} = t_{27} + t_{33}$	$t_{36} = t_{24} \times t_{35}$
$t_{37} = t_{36} + t_{34}$	$t_{38} = t_{27} + t_{36}$	$t_{39} = t_{29} \times t_{38}$
$t_{40} = t_{25} + t_{39}$		
$t_{41} = t_{40} + t_{37}$	$t_{42} = t_{29} + t_{33}$	$t_{43} = t_{29} + t_{40}$
$t_{44} = t_{33} + t_{37}$	$t_{45} = t_{42} + t_{41}$	$z_0 = t_{44} \times y_{15}$
$z_1 = t_{37} \times y_6$	$z_2 = t_{33} \times x_7$	$z_3 = t_{43} \times y_{16}$
$z_4 = t_{40} \times y_1$	$z_5 = t_{29} \times y_7$	$z_6 = t_{42} \times y_{11}$
$z_7 = t_{45} \times y_{17}$	$z_8 = t_{41} \times y_{10}$	$z_9 = t_{44} \times y_{12}$
$z_{10} = t_{37} \times y_3$	$z_{11} = t_{33} \times y_4$	$z_{12} = t_{43} \times y_{13}$
$z_{13} = t_{40} \times y_5$	$z_{14} = t_{29} \times y_2$	$z_{15} = t_{42} \times y_9$
$z_{16} = t_{45} \times y_{14}$	$z_{17} = t_{41} \times y_8$	

Andrea Visconti – Dipartimento di Informatica, Università degli Studi di Milano

CIRCUIT MINIMIZATION



Andrea Visconti – Dipartimento di Informatica, Università degli Studi di Milano

CIRCUIT MINIMIZATION

An example: Binary Multiplication.

Multiplication of polynomials of degree n over GF(2).

To find the minimum number of AND and XOR gates needed to multiply two polynomials.

Karatsuba-Ofman algorithm.

D.J.Bernstein, High-speed cryptography in characteristic 2;

Circuit Minimization Team page.