Research Directions on the Complexity of Boolean Circuits for Codes and Cryptography

Luís Brandão, Çağdaş Çalık, Morris Dworkin,
Nathan Dykas, René Peralta, Meltem Sönmez Turan

National Institute of Standards and Technology (Gaithersburg MD, USA)

Presentation at BFA 2019
Boolean Functions and their Applications (BFA)
June 17, 2019 @ Florence, Italy

Outline

1. Introduction

2. Reed-Solomon code

3. Symmetric Boolean functions

4. Binary polynomial multiplication

Outline

Circuit Complexity

Circuit Complexity

We are interested in several measures of circuit complexity:

Circuit Complexity

We are interested in several measures of circuit complexity:

- Multiplicative complexity (MC)
  - SMPC, ZKPs, side-channel protections (threshold)

Circuit Complexity

We are interested in several measures of circuit complexity:

- Multiplicative complexity (MC)
  - SMPC, ZKPs, side-channel protections (threshold)
- Additive complexity (AC)
  - Codes, matrix multiplication

Circuit Complexity

We are interested in several measures of circuit complexity:

- Multiplicative complexity (MC)
  - SMPC, ZKPs, side-channel protections (threshold)
- Additive complexity (AC)
  - Codes, matrix multiplication
- Time, energy, area
  - Actual implementation on a chip

Circuit Complexity

We are interested in several measures of circuit complexity:

- Multiplicative complexity (MC)
  - SMPC, ZKPs, side-channel protections (threshold)
- Additive complexity (AC)
  - Codes, matrix multiplication
- Time, energy, area
  - Actual implementation on a chip

Optimization with respect to any of these metrics is computationally intractable ... we do what we can.

Three topics in this talk:

- Reed-Solomon Codes
- Symmetric Boolean functions
- Recursive relations for binary multiplication

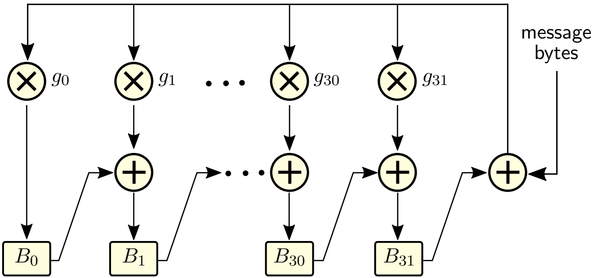
Three topics in this talk:

- Reed-Solomon Codes $\rightarrow$ AC
- Symmetric Boolean functions $\rightarrow$ MC
- Recursive relations for binary multiplication $\rightarrow$ MC and AC

Outline

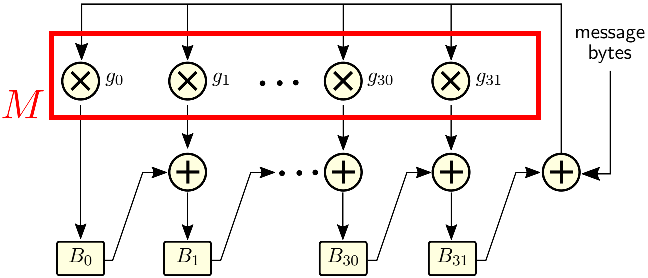
Reed-Solomon codes

RS(255,223) takes 223 message bytes $(m_0, \ldots, m_{222})$ and computes 32 check bytes $B_0, \ldots, B_{31}$ (these are initialized at 0).



Reed-Solomon codes

RS(255,223) takes 223 message bytes $(m_0, \ldots, m_{222})$ and computes 32 check bytes $B_0, \ldots, B_{31}$ (these are initialized at 0).



- Multiplication by a constant $g_i \in GF(2^8)$ is a linear operation.
- Each g_i can be viewed as an 8 by 8 binary matrix. (See M as a 256x8 matrix.)

Heuristics For Linear Circuit Minimization

- In crypto, it seems that everybody uses an algorithm due to Paar.
- In earlier work we showed that Paar's algorithm can do quite poorly.
- We have published two algorithms:
  - An exponential-time algorithm which we can use for systems of dimension up to about 20.
  - An efficient randomized heuristic which we use for larger systems.

Achieved improvement

- Paar (1997) “reduced” the number of XORs for multiplication by M to about 24 gates per finite field multiplication. This yields a circuit with about 760 XORs for multiplication by M.

Achieved improvement

- Paar (1997) “reduced” the number of XORs for multiplication by M to about 24 gates per finite field multiplication. This yields a circuit with about 760 XORs for multiplication by M.
- We constructed a circuit to do this using only 159 gates and depth 3.

Achieved improvement

- Paar (1997) "reduced" the number of XORs for multiplication by M to about 24 gates per finite field multiplication. This yields a circuit with about 760 XORs for multiplication by M.
- We constructed a circuit to do this using only 159 gates and depth 3.
- In this case, our solution is basically optimal. **But we have encountered linear maps for which our best methods do a lousy job of minimization.**

Outline

Symmetric Boolean Functions

A Boolean function $f : \{0,1\}^n \rightarrow \{0,1\}$ is said to be symmetric, if the output depends only on the Hamming weight of the input.

- Several useful sub-classes: elementary symmetric (Σ_i^n); exactly-counting (E_i^n); threshold (T_i^n).

This work: Find MC-efficient circuits for symmetric Boolean functions.

Hamming weight method

Since Muller and Preparata :

- A symmetric function is a sum of elementary symmetric functions Σ_i^n;
- Σ_i^n decomposes into a product of $\Sigma_{2^j}^n$;
- If $H = y_k \dots y_0$ is the binary representation of the integer sum $x_1 + \dots + x_n$, then $y_i = \Sigma_{2^i}^n(x_1, \dots, x_n)$.

Hamming weight method

Since Muller and Preparata :

- A symmetric function is a sum of elementary symmetric functions Σ_i^n;
- Σ_i^n decomposes into a product of $\Sigma_{2^j}^n$;
- If $H = y_k \dots y_0$ is the binary representation of the integer sum $x_1 + \dots + x_n$, then $y_i = \Sigma_{2^i}^n(x_1, \dots, x_n)$.

So start by computing H.

Hamming weight method

Since Muller and Preparata :

- A symmetric function is a sum of elementary symmetric functions Σ_i^n;
- Σ_i^n decomposes into a product of $\Sigma_{2^j}^n$;
- If $H = y_k \dots y_0$ is the binary representation of the integer sum $x_1 + \dots + x_n$, then $y_i = \Sigma_{2^i}^n(x_1, \dots, x_n)$.

So start by computing H.

The exact multiplicative complexity of H is known.

A generalization

- Think of the input $x_0 \ldots x_{n-1}$ as n wires of $\mathbf{weight\ 1}$;
- More generally, consider inputs whose **weights are powers of 2**;
- If three wires w_0, w_1, w_2 have weight 2^i you can replace these wires with
  - 1 wire $u = (w_0 + w_1)(w_0 + w_2) + w_0$ of weight 2^{i+1} ; and
  - one wire $v = (w_0 + w_1 + w_2)$ of weight 2^i .

A generalization

- Think of the input $x_0 \dots x_{n-1}$ as n wires of $\mathbf{weight\ 1}$;
- More generally, consider inputs whose **weights are powers of 2**;
- If three wires w_0, w_1, w_2 have weight 2^i you can replace these wires with
  - 1 wire $u = (w_0 + w_1)(w_0 + w_2) + w_0$ of weight 2^{i+1} ; and
  - one wire $v = (w_0 + w_1 + w_2)$ of weight 2^i .

u and v are just the outputs of a full adder. The multiplicative cost of this operation is 1 AND gate.

A generalization

- Think of the input $x_0 \dots x_{n-1}$ as n wires of $\mathbf{weight\ 1}$;
- More generally, consider inputs whose **weights are powers of 2**;
- If three wires w_0, w_1, w_2 have weight 2^i you can replace these wires with
  - 1 wire $u = (w_0 + w_1)(w_0 + w_2) + w_0$ of weight 2^{i+1} ; and
  - one wire $v = (w_0 + w_1 + w_2)$ of weight 2^i .

u and v are just the outputs of a full adder. The multiplicative cost of this operation is 1 AND gate.

For symmetric functions, this reduces the arity of the function to be computed by 1.

Example : E_8^4

Example: find MC-optimal circuit for the exactly-counting E_4^8 (outputs 1 iff the input has four 1's) — posed as open problem in 2008 [BP08].

- 4 applications of the full adder operation reduces the problem from 8 wires of weight 1 to 4 wires (y_3, y_2, y_1, y_0) of weights 4,2,1,1 respectively.

Example : E_8^4

Example: find MC-optimal circuit for the exactly-counting E_4^8 (outputs 1 iff the input has four 1's) — posed as open problem in 2008 [BP08].

- 4 applications of the full adder operation reduces the problem from 8 wires of weight 1 to 4 wires (y_3, y_2, y_1, y_0) of weights 4,2,1,1 respectively.
- The function E_8^4 is 1 iff $(y_3, y_2, y_1, y_0) \in \{(1,0,0,0),(0,1,1,1)\}$;

Example : E_8^4

Example: find MC-optimal circuit for the exactly-counting E_4^8 (outputs 1 iff the input has four 1's) — posed as open problem in 2008 [BP08].

- 4 applications of the full adder operation reduces the problem from 8 wires of weight 1 to 4 wires (y_3, y_2, y_1, y_0) of weights 4,2,1,1 respectively.
- The function E_8^4 is 1 iff $(y_3, y_2, y_1, y_0) \in \{(1,0,0,0),(0,1,1,1)\}$;
- Equivalently, $E_8^4 = (y_3 + y_0)(y_3 + y_1)(y_3 + y_2)$;

Example : E_8^4

Example: find MC-optimal circuit for the exactly-counting E_4^8 (outputs 1 iff the input has four 1's) — posed as open problem in 2008 [BP08].

- 4 applications of the full adder operation reduces the problem from 8 wires of weight 1 to 4 wires (y_3, y_2, y_1, y_0) of weights 4,2,1,1 respectively.
- The function E_8^4 is 1 iff $(y_3, y_2, y_1, y_0) \in \{(1,0,0,0),(0,1,1,1)\}$;
- Equivalently, $E_8^4 = (y_3 + y_0)(y_3 + y_1)(y_3 + y_2)$;
- Thus $C_\wedge(E_8^4) \leq 4 + 2 = 6$;

Example : E_8^4

Example: find MC-optimal circuit for the exactly-counting E_4^8 (outputs 1 iff the input has four 1's) — posed as open problem in 2008 [BP08].

- 4 applications of the full adder operation reduces the problem from 8 wires of weight 1 to 4 wires (y_3, y_2, y_1, y_0) of weights 4,2,1,1 respectively.
- The function E_8^4 is 1 iff $(y_3, y_2, y_1, y_0) \in \{(1,0,0,0),(0,1,1,1)\}$;
- Equivalently, $E_8^4 = (y_3 + y_0)(y_3 + y_1)(y_3 + y_2)$;
- Thus $C_\wedge(E_8^4) \leq 4 + 2 = 6$;
- It was known already that $C_\wedge(E_8^4) \geq 6$. So this fully solves the problem.

Results

- For functions of up to 6 variables, we have constructed MC-optimal circuits;

Results

- For functions of up to 6 variables, we have constructed MC-optimal circuits;
- So, if a sequence of full adder operations decrease the number of variables of a target symmetric function to 6 or less, we can construct a pretty good circuit (is it optimal?);

Results

- For functions of up to 6 variables, we have constructed MC-optimal circuits;
- So, if a sequence of full adder operations decrease the number of variables of a target symmetric function to 6 or less, we can construct a pretty good circuit (is it optimal?);
- We generated circuits for all symm. functions with up to 25 vars;

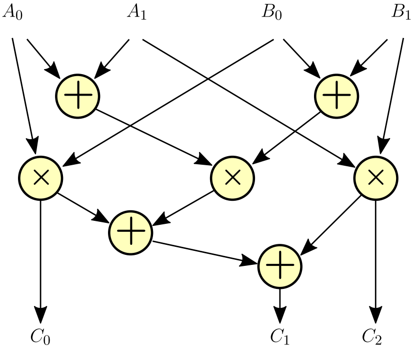
Results

- For functions of up to 6 variables, we have constructed MC-optimal circuits;
- So, if a sequence of full adder operations decrease the number of variables of a target symmetric function to 6 or less, we can construct a pretty good circuit (is it optimal?);
- We generated circuits for all symm. functions with up to 25 vars;
- We believe these circuits are optimal for symmetric functions of 21 or fewer variables.

Outline

Searching for best Karatsuba recurrences

Example: multiplication of two binary polynomials of degree 10



$$(A_0 + A_1x^5) \cdot (B_0 + B_1x^5) = C_0 + C_1x^5 + C_2x^{10}$$

A_0, A_1, B_0, B_1 are polynomials of degree 5.

Karatsuba recurrences

For multiplication of two n-term binary polynomials P and Q.

Let $M(n)$ be the gate complexity (over $\wedge$ and $\oplus$).
A $k-$way Karatsuba recurrence arises from splitting the polynomials into k pieces. Recurrences are of the form

$$M(n) \leq \alpha M(n/k) + \beta n + \gamma.$$

Karatsuba recurrences

For multiplication of two n-term binary polynomials P and Q.

Let $M(n)$ be the gate complexity (over $\wedge$ and $\oplus$).
A $k-$way Karatsuba recurrence arises from splitting the polynomials into k pieces. Recurrences are of the form

$$M(n) \leq \alpha M(n/k) + \beta n + \gamma.$$

1. α is the multiplicative complexity of multiplying two binary polynomials of degree k.
2. β and γ depend on the additive complexity of certain linear maps generated in the previous step. (FP 2018)

Methodology

Problem is to multiply two binary polynomials
$A = a_0 + a_1 X + \dots a_{n-1} X^{n-1}$, $B = b_0 + b_1 X + \dots b_{n-1} X^{n-1}$.

Targets: the product coefficients $t_k = \sum_{i+j=k} a_i b_j$.

The MC problem is to find a minimum-size set of *generators* of multiplicative complexity 1 which span the set of targets.

Methodology

Problem is to multiply two binary polynomials
$A = a_0 + a_1 X + \ldots a_{n-1} X^{n-1}$, $B = b_0 + b_1 X + \ldots b_{n-1} X^{n-1}$.

Targets: the product coefficients $t_k = \sum_{i+j=k} a_i b_j$.

The MC problem is to find a minimum-size set of *generators* of multiplicative complexity 1 which span the set of targets.

Symmetric Bilinear Generators: for $S \subset \{0, \ldots, n-1\}$, the symmetric bilinear forms $G_S = \big(\sum_{i \in S} a_i\big)\big(\sum_{i \in S} b_i\big)$.

Methodology

Problem is to multiply two binary polynomials
$A = a_0 + a_1X + \dots a_{n-1}X^{n-1}$, $B = b_0 + b_1X + \dots b_{n-1}X^{n-1}$.

Targets: the product coefficients $t_k = \sum_{i+j=k} a_i b_j$.

The MC problem is to find a minimum-size set of *generators* of multiplicative complexity 1 which span the set of targets.

Symmetric Bilinear Generators: for $S \subset \{0, \dots, n-1\}$, the symmetric bilinear forms $G_S = \big(\sum_{i \in S} a_i\big)\big(\sum_{i \in S} b_i\big)$.

Conjecture: For all n, there exists an optimal solution consisting solely of symmetric bilinear generators.

Methodology

1. **Find solutions with minimal sets of generators:**
   1.1 Limit search to subspaces that are expansions of the targets.
   1.2 Determine whether candidate subspaces have a basis of generators.
2. **Reduce # of XOR gates:** For each solution found in the previous step (there may be thousands of them), minimize the linear parts of the resulting circuit.

New results

$$M(6n) \leq 17M(n) + 83n - 26$$
$$M(7n) \leq 22M(n) + 106n - 31$$
$$M(8n) \leq 26M(n) + 147n - 40.$$

These result from exhaustive search of bilinear bases in the case of $n = 6$ and 7.

New results

$$M(6n) \leq 17M(n) + 83n - 26$$
$$M(7n) \leq 22M(n) + 106n - 31$$
$$M(8n) \leq 26M(n) + 147n - 40.$$

These result from exhaustive search of bilinear bases in the case of $n = 6$ and 7.

In the case of $n = 8$ we have not been able to traverse the full space. So we used some divine inspiration to restrict the space.

New results

$$
\begin{aligned}
M(6n) &\leq 17M(n) + 83n - 26\\
M(7n) &\leq 22M(n) + 106n - 31\\
M(8n) &\leq 26M(n) + 147n - 40.
\end{aligned}
$$

These result from exhaustive search of bilinear bases in the case of $n = 6$ and 7.

In the case of $n = 8$ we have not been able to traverse the full space. So we used some divine inspiration to restrict the space.

This yields smallest known circuits for binary polynomial multiplication for many values of n.

Will post circuits for multiplication of polynomials up to 100 or so.

Thank you for your attention

- Project email: circuit_complexity@nist.gov
- Circuit Complexity project at NIST: https://csrc.nist.gov/Projects/Circuit-Complexity
- GitHub webpage: https://github.com/usnistgov/Circuits/

Presentation at the Boolean Functions and their Applications (BFA)

June 17, 2019 @ Florence, Italy