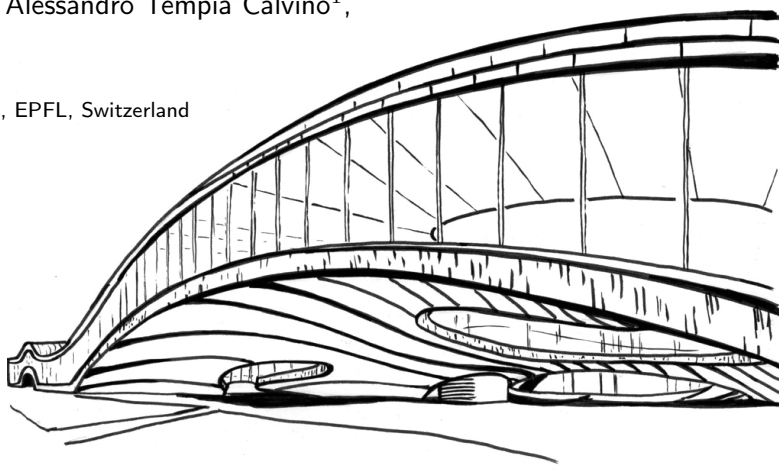
On the Synthesis of High-performance Homomorphic Boolean Circuits

Mingfei Yu[1], Sergiu Carpov[2], Alessandro Tempia Calvino[1], and Giovanni De Micheli[1]

[1]Integrated Systems Laboratory (LSI), EPFL, Switzerland
[2]Arcium, Switzerland

October 14, 2024, at WAHC'24



Outline

- Introduction
- Motivation
- Methodologies
- Experimental Evaluations
- Conclusion and Discussion

Introduction: FHE Families

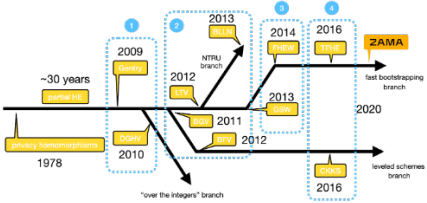


Fig. 1 Development of FHE schemes (courtesy of ZAMA).

Leveled schemes (BGV, BFV, CKKS, etc.):

- Features: Leveled; Slow bootstrapping; Word-level parallelism.
- Applications: Arithmetic functions over large integers or even floating points.

Fast bootstrapping schemes (FHEW, TFHE, etc.):

- Features: Efficient bootstrapping that involves a function evaluation.
- Application: Boolean functions; Arithmetic functions over small integers.

Introduction: Functional Bootstrapping (FBS)

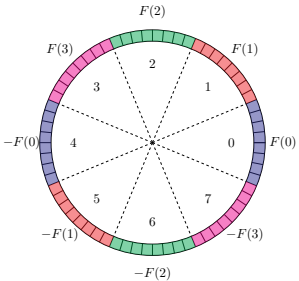


Fig. 2 Visualization of TFHE plaintext space $\mathbb{Z}_4$ and an encoded two-variable Boolean function $f(\boldsymbol{m}) = f(m_0, m_1)$: $\mathbb{B}^2 \to \mathbb{B}$ ($F(\varphi) = F(2m_1 + m_0) = f(\boldsymbol{m})$).

Homomorphically evaluating an n-input Boolean gate (whose function is f):

- Linear combination: Projecting m: $\mathbb{B}^n$ to φ: $\mathbb{Z}_p$ using Φ.
  - Typically, $\varphi = \Phi(\boldsymbol{m}) = \sum_{i=0}^{n-1}(m_i \cdot 2^i)$.
- Blind rotation: Using φ as index to privately evaluate function F.
  - Negacyclicity: If $p \leq \varphi < 2p$, $F(\varphi) = -F(\varphi - p)$.

Introduction: Homomorphic Boolean Circuit / TFHE Circuit

A TFHE circuit is a *directed acyclic graph* (DAG) of k-*input lookup-tables* (k-LUTs):

- Each LUT in the DAG has no more than k inputs.
- Selection of plaintext space size p determines k (by default, $2^k \leq p$).
- Each LUT corresponds to an FBS operation.

TFHE circuit synthesis is an *area-oriented technology mapping* problem:

- Technology mapping: Transforming a graph representation to optimize specific cost metrics.
- Circuit evaluation time correlates to #LUTs (area of the circuit).
- Off-the-shelf technology mappers (such as `YOSYS`) are exploited by TFHE compilers [1].

[1] Charles Gouert and Nektarios Georgios Tsoutsos. "Romeo: conversion and evaluation of HDL designs in the encrypted domain". In: *Design Automation Conference*. 2020

Motivation: How to Achieve More Compact TFHE Circuits?

Accommodating *larger* LUTs:

- With a fixed plaintext space p, LUTs with more than $\lfloor \log_2 p \rfloor$ inputs might be valid.
- A k-LUT is valid if there exists $\boldsymbol{w}$ such that $\max(\boldsymbol{m} \cdot \boldsymbol{w}) - \min(\boldsymbol{m} \cdot \boldsymbol{w}) < p$.
  - $\boldsymbol{w} = [w_0, \ldots, w_{k-1}]$ is a *weight assignment* and each w_i is an integer.
  - Formulate $\boldsymbol{w}$'s existence as an *integer linear programming* (ILP) problem [2].
- Existing approaches to leveraging larger LUTs are greedy.

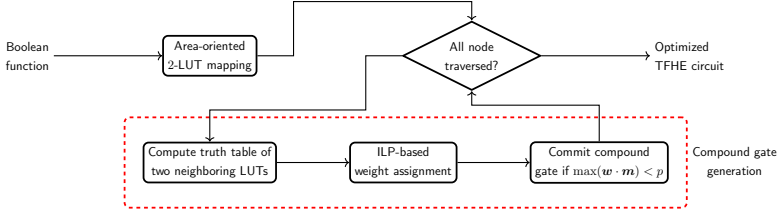


Fig. 3 AutoHoG's approach to leveraging large LUTs.

[2] Zhenyu Guan et al. "AutoHoG: Automating Homomorphic Gate Design for Large-Scale Logic Circuit Evaluation". In: *IEEE TCAD* 43.7 (2024), pp. 1971–1983

Motivation: How to Achieve More Compact TFHE Circuits

Supporting the *multi-value FBS* (MVFBS) technique [3]:

- Enables evaluating multiple functions over the same encrypted input.
  - Factoring the test polynomials into (1) a common factor and (2) parts specific to each LUT.
  - Allowing to share one blind rotation among evaluation of multiple LUTs.
  - Obtaining each evaluation result by multiplying the blind-rotated common accumulator with each specific part.
- Support for this technique in existing TFHE circuit synthesis approaches is passive.

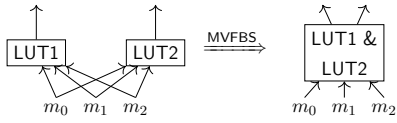


Fig. 4 Multi-value FBS consolidates evaluation of LUTs that share inputs.

[3] Sergiu Carpov, Malika Izabachène, and Victor Mollimard. "New Techniques for Multi-value Input Homomorphic Evaluation and Applications". In: *CT-RSA*. 2019, pp. 106–126

Motivation: How to Achieve More Compact TFHE Circuits

The state-of-the-art (SoTA) TFHE circuit synthesis approach [2] v.s. what is desired:

	SoTA	Desired
Exploiting large LUTs	Greedy, susceptible to local optima	Comprehensive exploration of design space, but efficient
Supporting MVFBS	Passive, post-synthesis LUT merging	Proactive, integrating supporting MVFBS into synthesis

What we provide are:

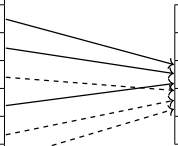
- An MVFBS-aware technology mapping approach.
- A detailed study of its application to plaintext space $\mathbb{Z}_4$.

[2] Zhenyu Guan et al. "AutoHoG: Automating Homomorphic Gate Design for Large-Scale Logic Circuit Evaluation". In: *IEEE TCAD* 43.7 (2024), pp. 1971–1983

Methodologies: Exploiting Larger LUTs

$\varphi = \sum_{i=0}^{2}(m_i \cdot 2^i)$	$\{m_2, m_1, m_0\}$	$F(\varphi)$
0	$\{0,0,0\}$	f_0
1	$\{0,0,1\}$	f_1
2	$\{0,1,0\}$	$f_2 = f_1$
3	$\{0,1,1\}$	f_3
4	$\{1,0,0\}$	$f_4 = f_1$
5	$\{1,0,1\}$	$f_5 = f_3$
6	$\{1,1,0\}$	$f_6 = f_3$
7	$\{1,1,1\}$	f_7

Correspondence between $F(\varphi)$ and $f(\boldsymbol{m})$
with **default weight assignment**



$\varphi = \sum_{i=0}^{2} m_i$	$\{m_2, m_1, m_0\}$	$F(\varphi)$
0	$\{0,0,0\}$	f_0
1	$\{0,0,1\}$, $\{0,1,0\}$, $\{1,0,0\}$	f_1
2	$\{0,1,1\}$, $\{1,0,1\}$, $\{1,1,0\}$	f_3
3	$\{1,1,1\}$	f_7

Correspondence between $F(\varphi)$ and $f(\boldsymbol{m})$
with **unit weight assignment**

Symmetry enables projection to a smaller plaintext space.

- Use **complete symmetry** to reduce required plaintext space: $2^n \to n+1$ ($n = 3$ in the above example).
- With plaintext space $\mathbb{Z}_4$, any 3-input LUTs implementing symmetric functions are allowed.

Methodologies: Exploiting Larger LUTs

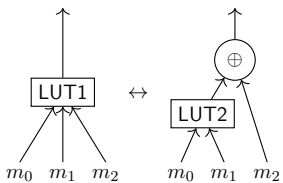


Fig. 5 A LUT is negacyclic if its function is *top-XOR-decomposible*.

$\varphi = \sum_{i=0}^{2}(m_i \cdot 2^i)$	$\{m_2, m_1, m_0\}$	$F(\varphi)$
0	$\{0,0,0\}$	f_0
1	$\{0,0,1\}$	f_1
2	$\{0,1,0\}$	f_2
3	$\{0,1,1\}$	f_3
4	$\{1,0,0\}$	$f_4 = -f_0$
5	$\{1,0,1\}$	$f_5 = -f_1$
6	$\{1,1,0\}$	$f_6 = -f_2$
7	$\{1,1,1\}$	$f_7 = -f_3$

When negacyclic functions meet negacyclic polynomials:

- Use **negacyclicity** to reduce required plaintext space: $2^n \rightarrow 2^{n-1}$ ($n = 3$ in the above example).
- With plaintext space $\mathbb{Z}_4$, any 3-input LUTs implementing negacyclic functions are allowed.

Methodologies: MVFBS-aware Technology Mapping

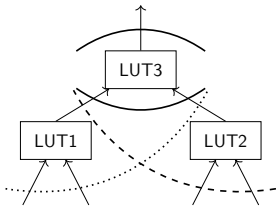


Fig. 6 Three qualified cuts rooted at LUT3, indicating three choices:
(1) solid curve: the current implementation;
(2) dotted curve: implementing LUT3 and LUT1 as a 3-input LUT;
(3) dashed curve: implementing LUT3 and LUT2 as a 3-input LUT.

(0) *Cut enumeration*:

- (On the subject graph) Collect sub-graphs (*cuts*) rooted at each node and with no more than 3 inputs (*leaves*).

(1) *Matching*:

- Check the function of 3-input cuts and abandon those neither symmetric nor negacyclic ones.

MVFBS-aware Technology Mapping

(2) *Selection*: Go through the graph in topological order and update the *representative cut* for each node by estimating the corresponding cost.

- Recursive cost estimation to reflect each change in a node's representative cut.
- Each cost estimation heuristic has its merits and limits.
- *Area flow* [4] and *exact area* [5] are adopted in our implementation.
  - Area flow estimates a cut's cost from a global perspective, maximizing logic sharing.
  - Exact area estimates a cut's cost from a local perspective, finding the most efficient implementation of each node.

(3) Deriving the TFHE circuit: Traversing the subject graph in reverse-topological order; For each reached node, its representative cut indicates its new implementation.

---

[4] Valavan Manohararajah, Stephen D. Brown, and Zvonko G. Vranesic. "Heuristics for Area Minimization in LUT-Based FPGA Technology Mapping". In: *IEEE TCAD* 25.11 (2006), pp. 2331–2340

[5] Alan Mishchenko et al. "Combinational and Sequential Mapping with Priority Cuts". In: *ICCAD*. 2007, pp. 354–361

Methodologies: MVFBS-aware Technology Mapping

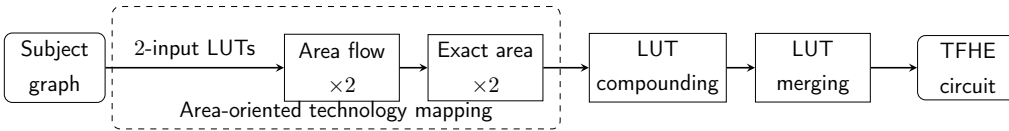
Enhanced exact area:

- Equipping technology mapping with MVFBS-awareness by maintaining a *label monitor.*
  - *Label* of a cut consists of its (1) input signals and (2) function type.
  - MVFBS applies to selected cuts that share the label.
  - The label monitor keeps track of currently selected representative cuts.
- Deriving each cut's label at the matching stage.
- Referring to the label monitor when computing each cut's exact area.
  - If the label of a cut is contained in the monitor, its exact area is 0.
  - Update the monitor every time a representative cut change is made.

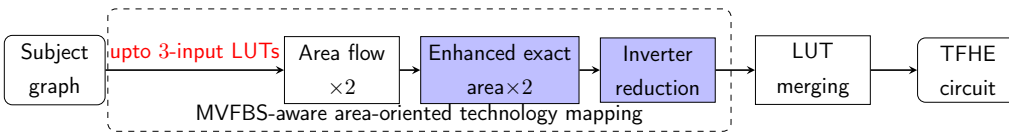
Inverter reduction:

- Excessive inverters reduce opportunities for MVFBS application.

Experimental Evaluations: Setups



(a) SoTA



(b) Ours

Fig. 7 Evaluated TFHE circuit synthesis approaches.

Experimental Evaluations: Profiling Optimized TFHE Circuits

Benchmark	*SoTA*			*TechMap*			*Enhanced TechMap*		
	#LUTs	Merge[%]	Time[s]	#LUTs	Merge[%]	Time[s]	#LUTs	Merge[%]	Time[s]
round-robin arbiter	11 434	0.00	0.06	11 605	0.00	0.07	11 605	0.00	0.08
coding-cavlc	554	9.62	<0.01	545	8.86	<0.01	542	11.15	<0.01
ALU control unit	97	11.82	<0.01	94	8.74	<0.01	94	10.48	<0.01
decoder	291	3.96	<0.01	292	2.34	<0.01	292	3.95	<0.01
i2c controller	1 109	2.89	<0.01	1 032	2.46	<0.01	1 029	3.38	<0.01
int to float converter	175	8.85	<0.01	171	8.56	<0.01	170	11.46	<0.01
memory controller	36 446	2.70	0.28	35 510	2.96	0.12	35 016	5.45	0.13
priority encoder	833	0.12	<0.01	818	0.12	<0.01	818	0.24	<0.01
look-ahead XY router	174	1.14	<0.01	134	4.96	<0.01	126	11.27	<0.01
voter	5 170	12.55	0.03	3 306	23.98	0.01	2 936	33.15	0.01
Average	13 611.30	5.08	0.059	9 925.55	7.99	0.039	9 508.15	12.26	0.435
Norm.	**1**	**1**	**1**	**0.729**	**1.572**	**0.661**	**0.699**	**2.413**	**0.737**

- *Merging rate* ('merge'): reduction in #LUTs after applying LUT merging.
- (Interestingly) *TechMap* outperforms the SoTA in merging rate.

Experimental Evaluations: Estimating Execution Cost

Benchmark	*SoTA*	*TechMap*	*Enhanced TechMap*
	Exec. cost	Exec. cost	Exec. cost
round-robin arbiter	457 360	464 200	464 200
coding-cavlc	22 160	21 800	21 680
ALU control unit	3 880	3 760	3 760
decoder	11 640	11 680	11 680
i2c controller	44 360	41 280	41 160
int to float converter	7 000	6 840	6 800
memory controller	1 603 624	1 562 440	1 540 704
priority encoder	33 320	32 720	32 720
look-ahead XY router	6 960	5 360	5 040
voter	227 480	145 464	117 440
Average	553 490.80	405 266.50	387 796.00
Norm.	**1**	**0.732**	**0.701**

- Exploit `Concrete` for precisely estimating the execution cost per FBS operation.
- The estimated speedup correlates with the #LUTs reduction in the circuit model.

Conclusion and Discussion

Goal: Synthesizing more compact TFHE circuits.

- Contribution 1: a systematic analysis of Boolean properties resulting in valid large LUTs, facilitating technology mapping.
- Contribution 2: Maximizing the usage of MVFBS during technology mapping.
- Effectiveness: Achieved a 29.94% reduction in circuit execution time.

Open problems:

- More dedicated heuristics to increase the application of MVFBS.
- Do we need to consider all valid large LUTs?
  - After a certain threshold on p, accommodating larger LUTs may not pay off [7].
  - A subset of large LUTs can be sufficient (effectiveness and efficiency).

[7] Sergiu Carpov. *A Fast Heuristic for Mapping Boolean Circuits to Functional Bootstrapping*. Cryptology ePrint Archive, Paper 2024/1204. 2024

Conclusion and Discussion

Feel free to take a look at our open-source tools:

TFHE Circuit Optimizer

Execution Cost Estimator

- The circuit optimizer is currently implemented as a part of the logic synthesis library `mockturtle`.
- Ongoing development aims to extract the optimizer as a standalone project, incorporating the execution cost estimator.

On the Synthesis of High-performance Homomorphic Boolean Circuits

Mingfei Yu[1], Sergiu Carpov[2], Alessandro Tempia Calvino[1], and Giovanni De Micheli[1]

[1]Integrated Systems Laboratory (LSI), EPFL, Switzerland
[2]Arcium, Switzerland

October 14, 2024, at WAHC'24