Cycle-based Synthesis and Exact Synthesis

PART 1

Pawel KERNTOPF

Warsaw University of Technology and University of Łódź, POLAND

COST Action IC1405, Training School

Toruń, Poland, August 31, 2017

Outline

- Motivation
- General remarks
- Basic notions
- Two papers published in 2002
- Theoretical results
- Evolution of cycle-based approaches to reversible circuit synthesis
- Additional remarks
- Examples of post-synthesis

Motivation

Given a specification of a reversible function *f*

- Synthesis:

  - Find a circuit realizing *f*

- Optimal synthesis:

  - Find a cost minimal circuit realizing *f*

Motivation (cnd.)

- Since 2002 many synthesis algorithms of different features and quality have been developed
- The presentation should be

  - very selective

  - instructive

  - easy to follow
- I have decided to focus on one approach only and to present its progress in a chronological order

Motivation (cnd.)

- A large group of synthesis algorithms consists in consecutive mapping of some items in the specified representation of a given reversible function into subcircuits of a reversible circuit

  Examples:

  - TT (Truth Table)
  - ESOP (Exclusive OR Sum of Products)
  - PPRM (Positive Polarity Reed-Muller)
  - BDD (Binary Decision Diagram)
  - Permutation – sets of cycles

Reversible functions and gates

Definition 1

A completely specified n-input n-output Boolean function is called **reversible** if it maps each input assignment into a unique output assignment.

There are $2^n!$ reversible n-input n-output Boolean functions.

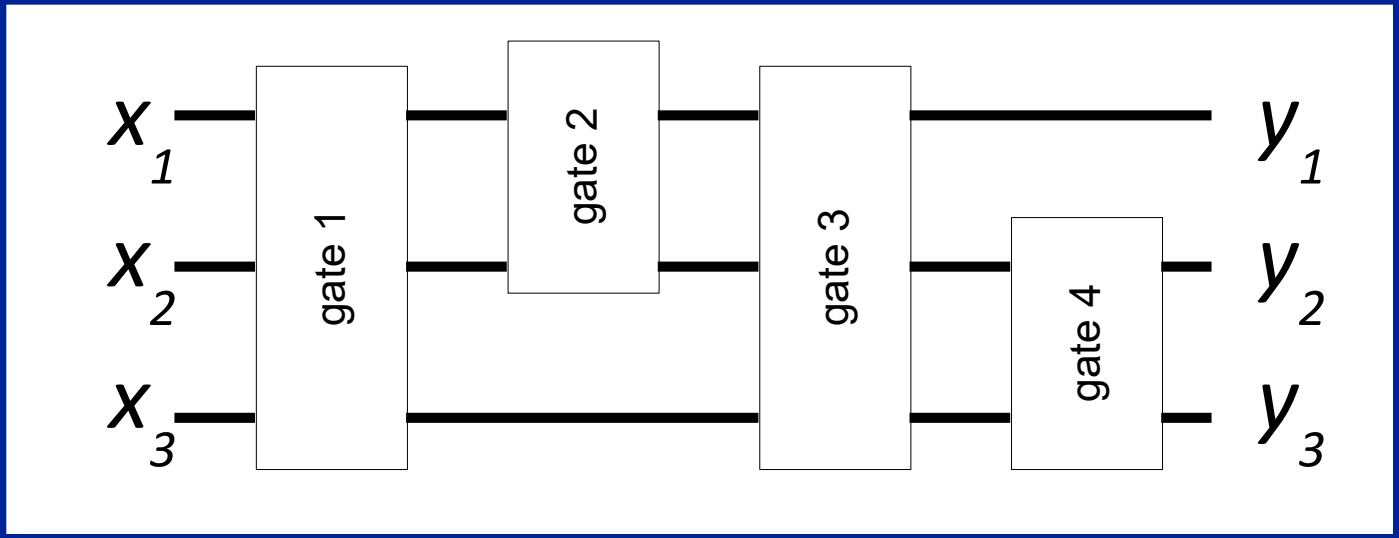
For $n = 3$ this number is **40,320** and for $n = 4$ it is greater than 2×10^{13}

Definition 2

An n-input n-output gate (circuit) is called **reversible** if it realizes an n-input n-output **reversible** function.

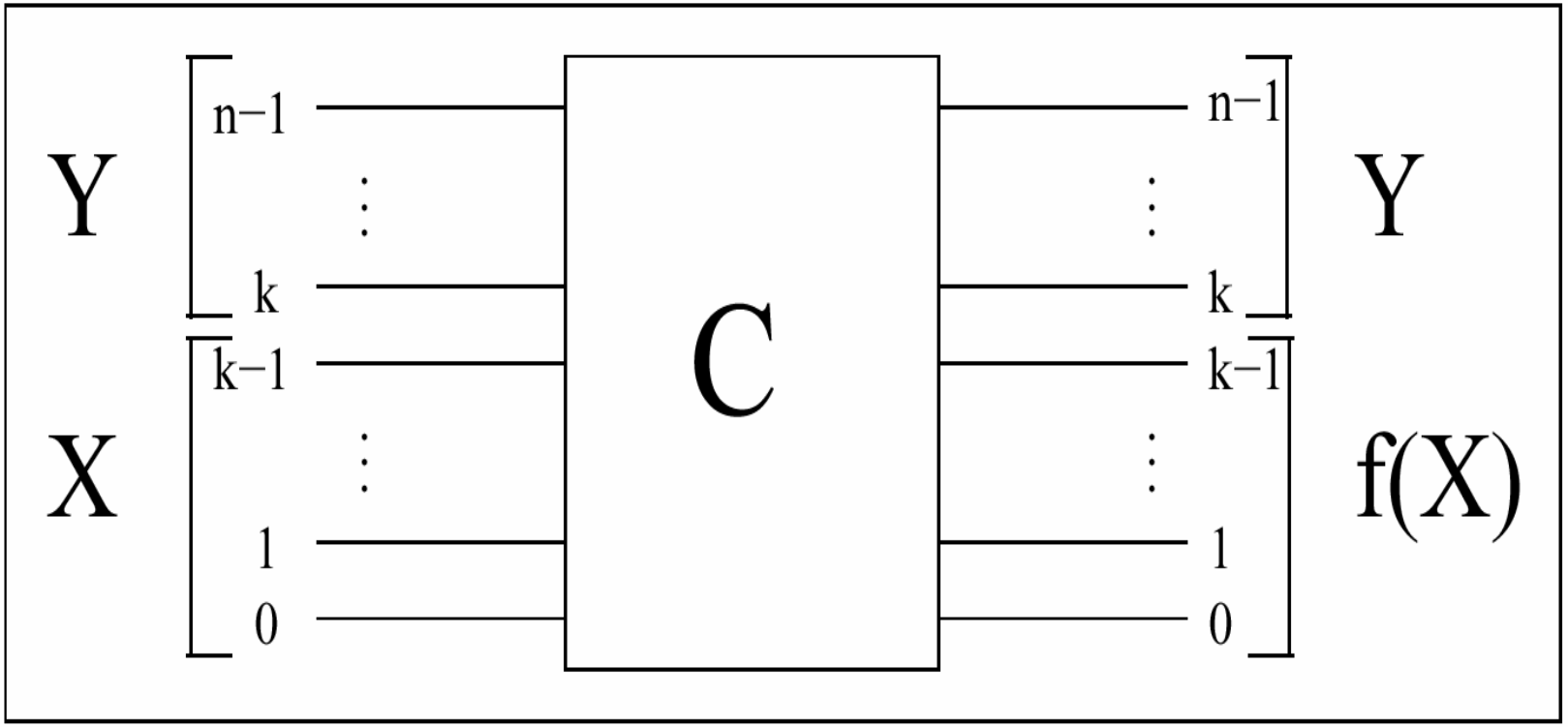
Reversible circuits

- realize reversible functions
- information-lossless
- no fan-out
- built as a cascade of reversible gates:



Reversible circuits

General model of a reversible circuit



NCT gate family (NOT, CNOT, Toffoli)

NOT

$y = x \oplus 1$

CNOT

$y_1 = x_1$

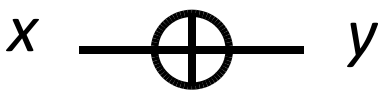
$y_2 = x_2 \oplus x_1$

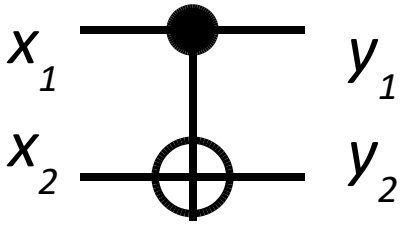
Toffoli

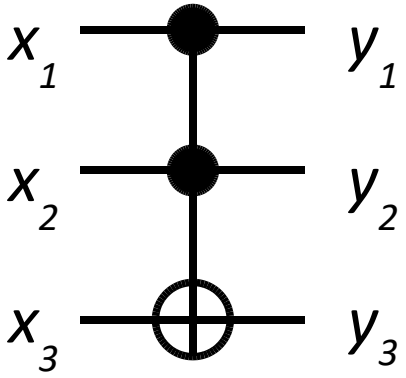
$y_1 = x_1$

$y_2 = x_2$

$y_3 = x_3 \oplus x_1 x_2$

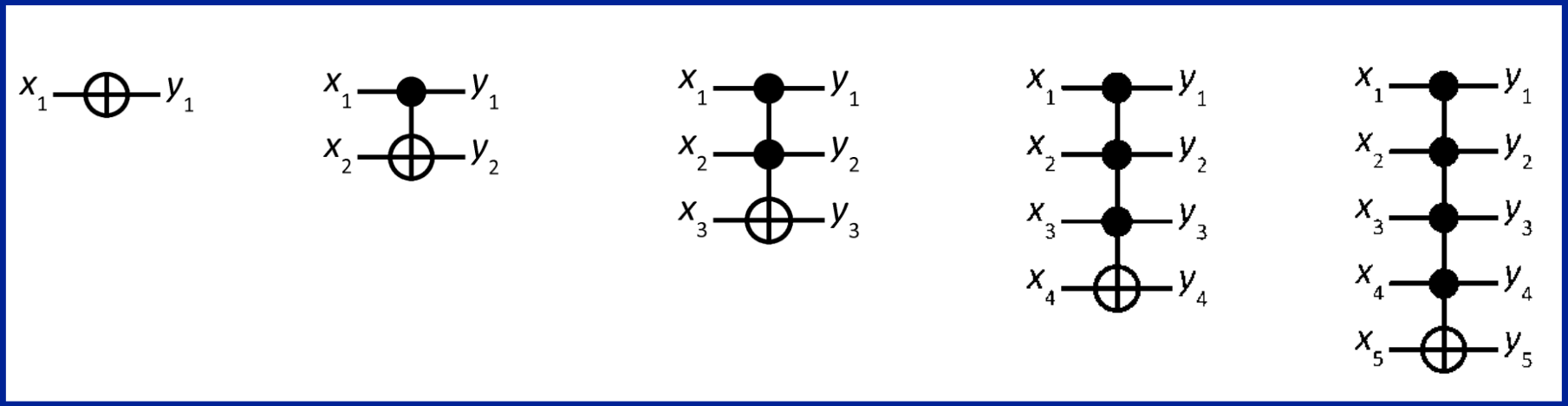






NCT gate family - generalization

NOT	CNOT	Toffoli3	Toffoli4	Toffoli5
$y_1 = x_1 \oplus 1$	$y_1 = x_1$	$y_1 = x_1$	$y_1 = x_1$	$y_1 = x_1$
	$y_2 = x_2 \oplus x_1$	$y_2 = x_2$	$y_2 = x_2$	$y_2 = x_2$
		$y_3 = x_3 \oplus x_1x_2$	$y_3 = x_3$	$y_3 = x_3$
			$y_4 = x_4 \oplus x_1x_2x_3$	$y_4 = x_4$
				$y_5 = x_5 \oplus x_1x_2x_3x_4$



A gate line with ⊕ sign is called the target

Inverse functions and circuits

If a circuit

$$C = G_1 G_2 \ldots G_k$$

realizes a reversible function f then the circuit

$$C^{-1} = G_k^{-1} \ldots G_2^{-1} G_1^{-1}$$

realizes the inverse function f^{-1}

- All gates belonging to NCT family of gates are self-inverse, i.e. $G^{-1} = G$
- Thus C^{-1}, an inverse of a circuit C built from NCT gates, is a mirror image of C

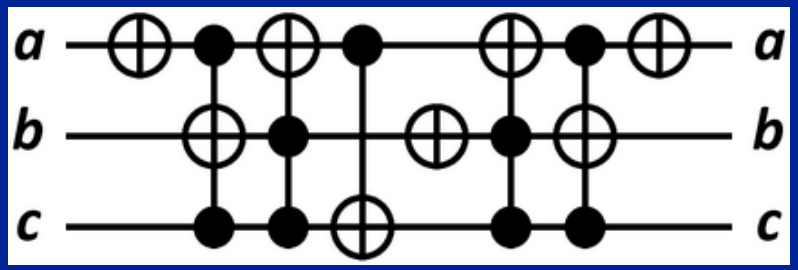
Cost functions

- **Gate count (GC)** = number of gates in a circuit
- **Quantum cost (QC)** = sum of costs of elementary quantum gates (i.e. NOT, CNOT, CV, CV^+) implementing given reversible gates
- NOT gate QC = 1
- CNOT gate QC = 1
- Toffoli 3-input/output gate QC = 5
- Toffoli 4-input/output gate QC = 13
- **Number of lines in a circuit (#L)** very important

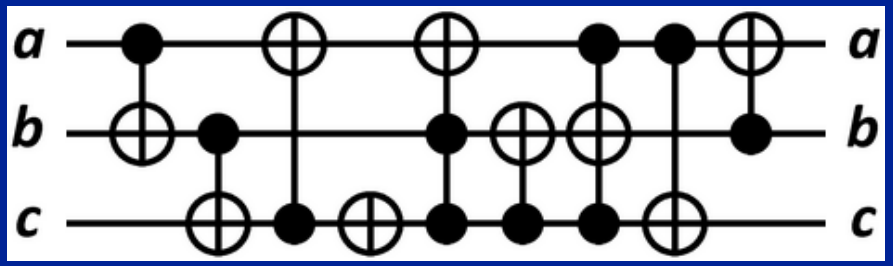
Tradeoff between GC and QC (1)

Two circuits implementing the same function

GC = 8, QC = 24



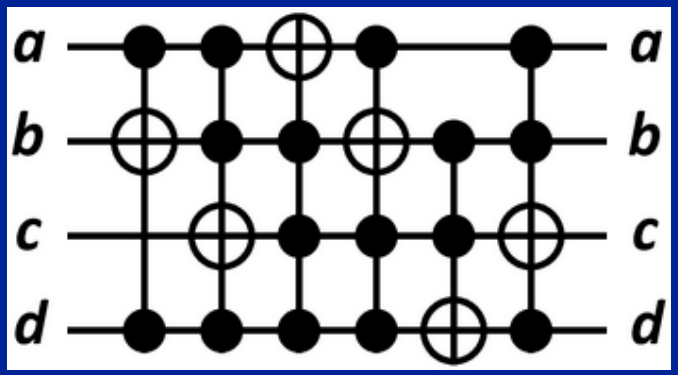
GC = 9, QC = 13



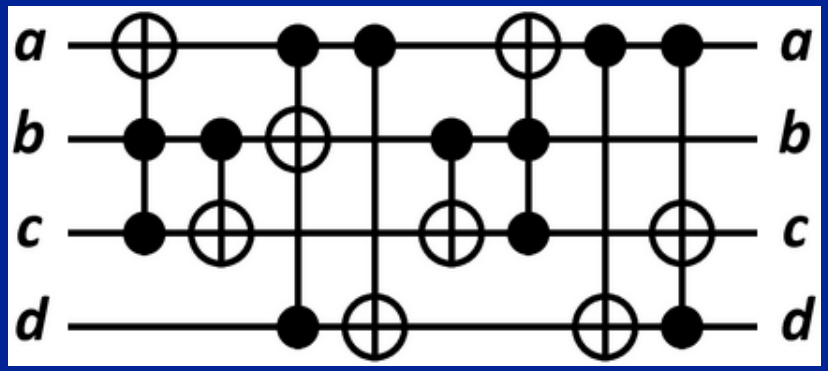
Tradeoff between GC and QC (2)

Benchmark *mini_alu* [RevLib Benchmarks Page]

RevLib: GC = 6, QC = 62



GC=8, QC = 24



Tradeoff between GC and QC (3)

- *nth_prime4_inc* [Maslov's Benchmarks Page]
- Best known: QC = 51 (GC=15) [Saeedi et al. 2010]
- Our result: QC = 26 (GC=14)

GC	QC	#circuits	time [s]
11	53-55	12	10
12	32-46	2288	591
13	31-93	187945	7282
14	**26**-114	11056332	292578

Library up to 8 gates, 16GB of memory, Power5+ 1,65GHz

Assumptions

- Binary reversible circuits
- NCT library of gates (NOT, CNOT and Toffoli gates)
- Quantum cost (QC) optimization vs. gate count (GC) and number of lines
- Selected basic algorithms will be considered
  - cycle-based algorithms
  - transformation-based algorithms
  - ESOP-based algorithms
  - BDD-based algorithms

Permutation

x	$x_1x_2x_3$	$y_1y_2y_3$	y
0	000	000	0
1	001	101	5
2	010	010	2
3	011	110	6
4	100	100	4
5	101	001	1
6	110	011	3
7	111	111	7

- **f(1)=5, f(3)=6, f(5)=1, f(6)=3**
- **(1,5) (3,6)**

Properties of permutations (1)

- cycle of length *k* (k-cycle) $(a_1, a_2, \ldots, a_k)$

  iff $f(a_1) = a_2$, $f(a_2) = a_3$, … , $f(a_k) = a_1$
- Cycles *f* and *g* are called *disjoint* if they have no elements in common
- Every permutation can be written as a set of disjoint cycles $c_0, c_1, \ldots, c_m$

Properties of permutations (2)

- Transposition
  - permutes exactly two elements
  - does not change any other elements
- Product of permutations:

  they are multiplied by first applying one, then the other, e.g. $(1,2)\circ(2,3) = (1,3,2)$
- Decomposition of a permutation: representing it as a product of two or more permutations
- Every permutation can be written as a product of disjoint cycles $c_0, c_1, \ldots, c_m$

Properties of permutations (3)

- Every permutation can be written as a product of transpositions

  Let us consider all possible decompositions of a permutation into transpositions

- Theorem: The parity of the number of transpositions is constant

- Definitions:

  - Even permutations are those of even parity

  - Odd permutations are those of odd parity

Properties of permutations (4)

- **A selection of known decompositions:**

$(x_1, x_2, \ldots, x_k) = (x_1, x_2)\,(x_{k-1}, x_k)\,(x_1, x_3, x_4, \ldots, x_{k-1})$

(a, b) (b, c) = [(a, b) (d, e)] [(d, e) (b, c)]

(a, b, c) = [(a, b) (d, e)] [(d, e) (a, c)]

(a, b, c) (d, e, f) = [(a, b) (d, e)] [(a, c) (d, f)]

An important paper

- K. Iwama, Y. Kambayashi, S. Yamashita

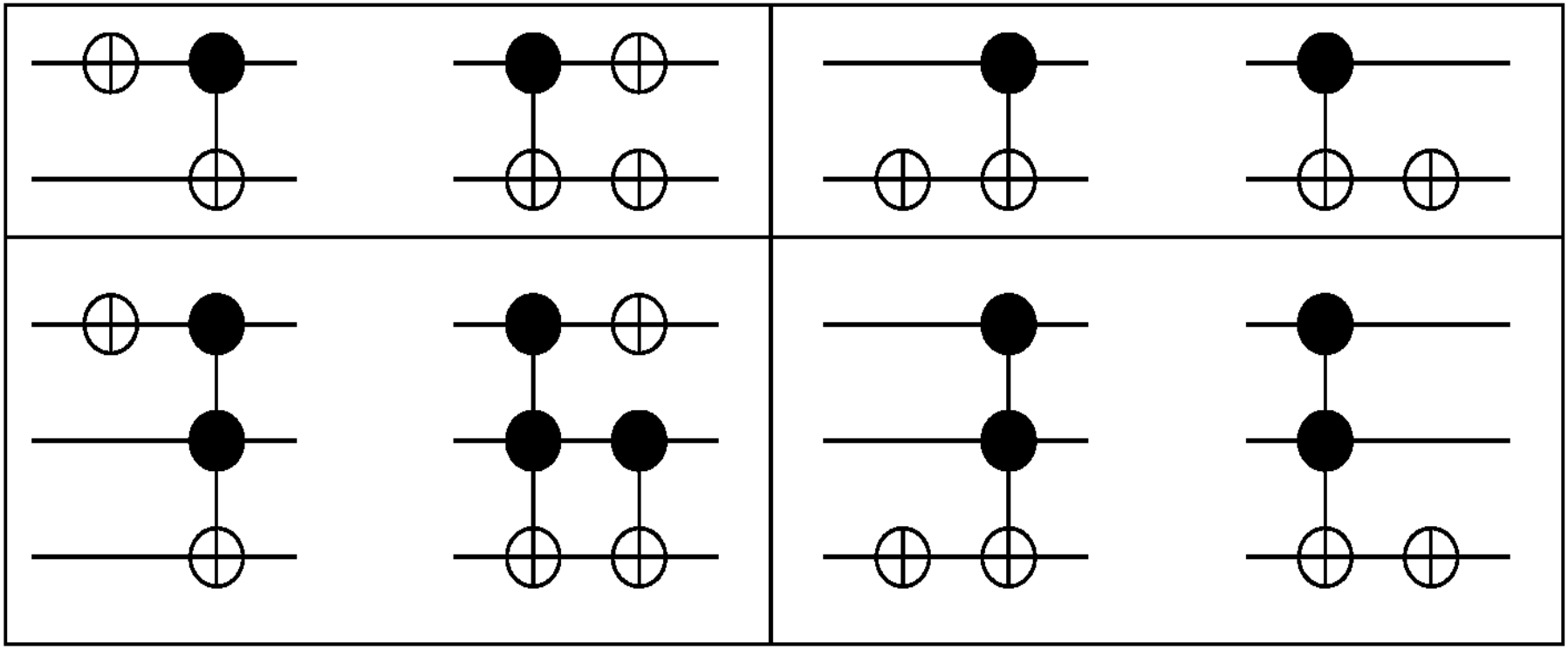
  Transformation Rules for Designing CNOT-based Quantum Circuits

  Design Automation Conference 2002

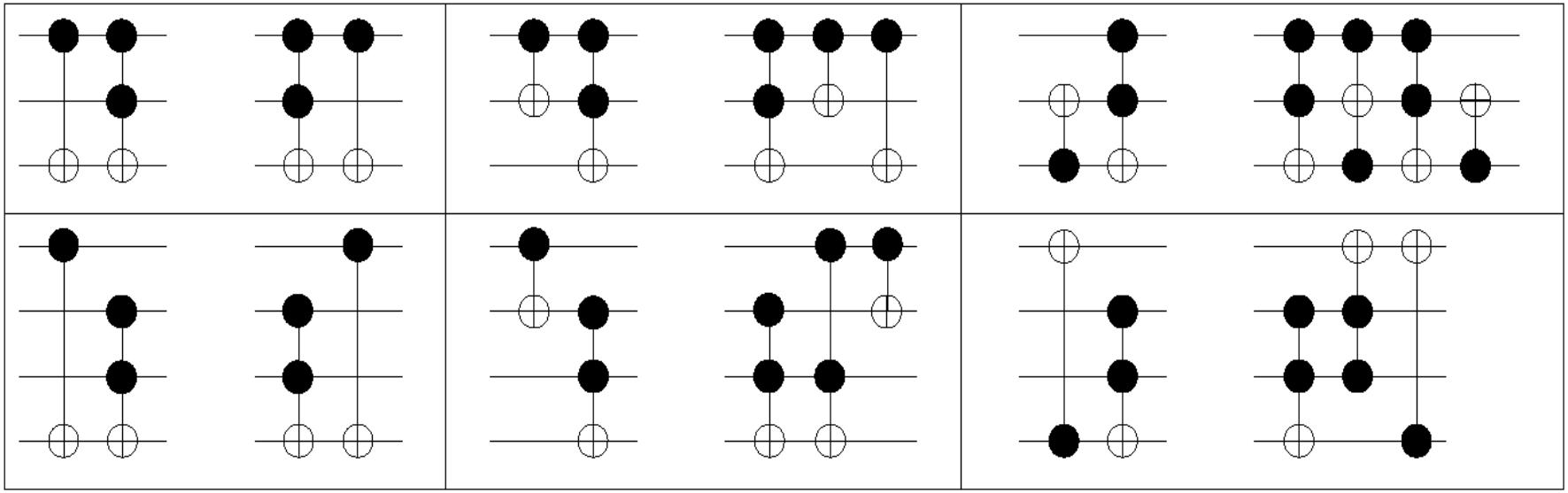
  Main ideas:

- Moving rules for gates
- Canonical form of a reversible circuit (never used in synthesis)

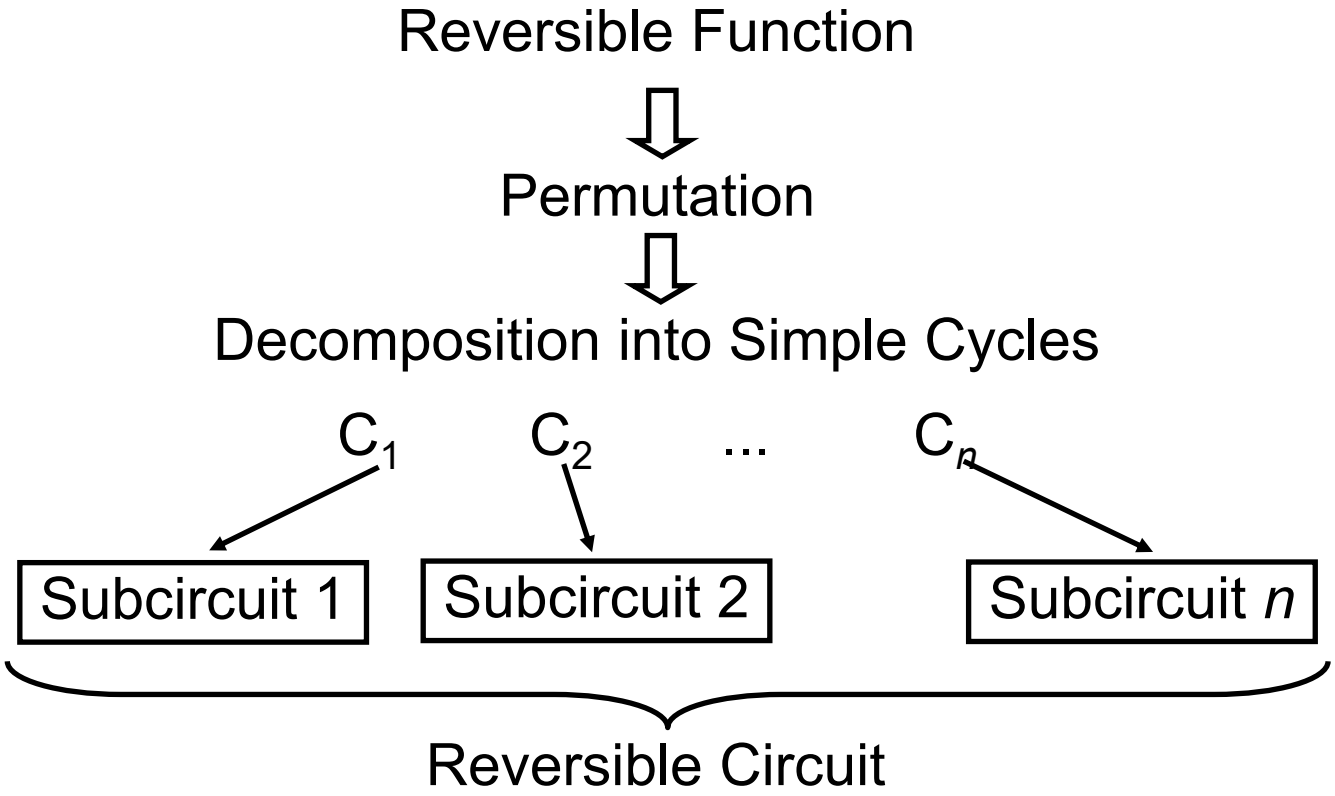
Moving rules (1)



Moving rules (2)



Cycle-based approaches (1)



Subcircuit selection is based on one cycle only

Cycle-based approaches (2)

- V. Shende, A. Prasad, I. Markov, J. Hayes

  ICCAD 2002, IEEE Trans. on CAD 2003

  Main idea:

- Permutation → product of transpositions
- Product of transpositions → product of pairs of disjoint transpositions
- Arbitrary pair of disjoint transpositions → subcircuit of the final circuit

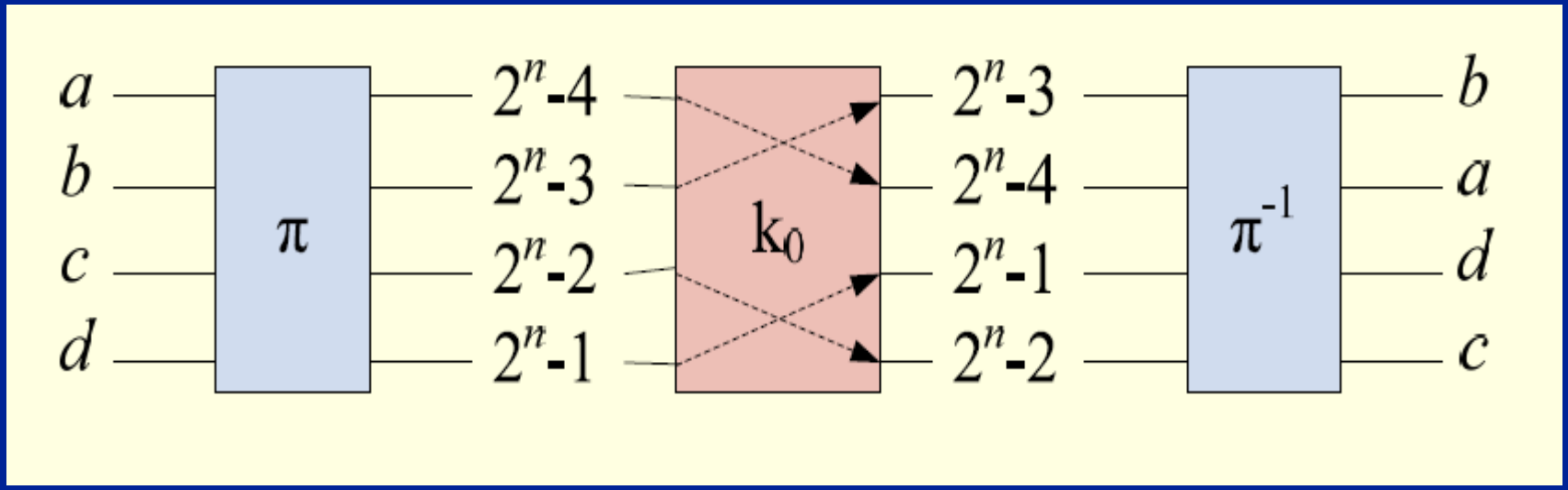
Cycle-based approaches (3)

Z. Sasanian, M. Saeedi, M. Sedighi, M. Saheb Zamani, - ASPDAC 2009

The pair of disjoint transposition

(a, b) (c, d)

can be realized by the following circuit:



Cycle-based approaches (4)

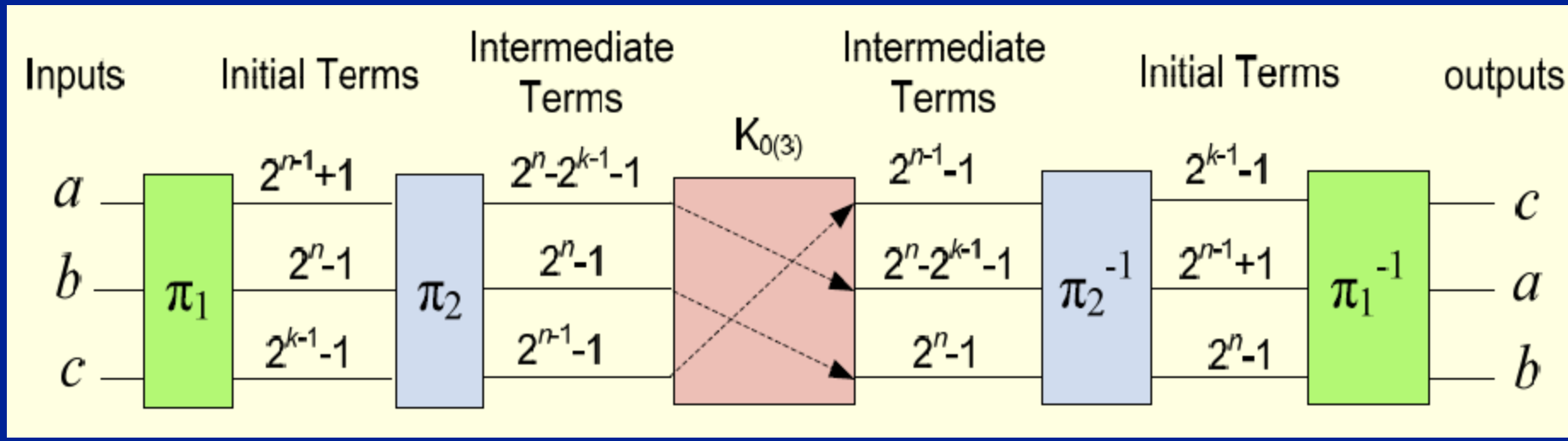
Z. Sasanian, M. Saeedi, M. Sedighi,

M. Saheb Zamani, M. Arabzadeh - ASPDAC 2009,

IWLS 2009, Microelectronics J. 2010,

JETC 2010, IWLS 2011

3-cycle (a, b, c) can be realized by the circuit:



Improvement in gate count

	Shende et al.	Sasanian et al.
(2-cycle) (2-cycle)	18n – 44	18n – 44
(3-cycle)	36n – 88	16n – 34
(3-cycle) (3-cycle)	36n – 88	22n – 34

Decomposition of cycles

LIBRARY

(2-cycle) (2-cycle),

(3-cycle)

(3-cycle) (3-cycle)

(4-cycle)

(4-cycle) (4-cycle)

(5-cycle)

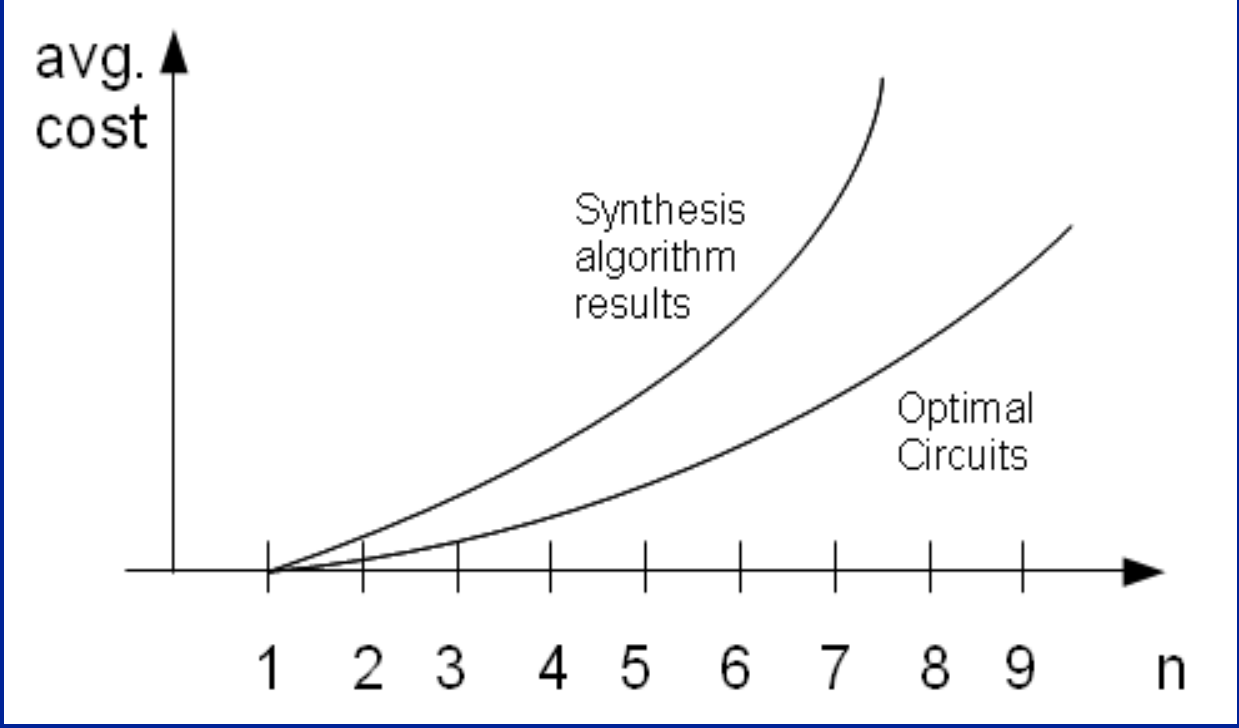
(5-cycle) (5-cycle)

The result of decomposition is a set of 5-cycles and probably a cycle of length less than 5

The synthesis of a cycle pair is more efficient than the synthesis of two single cycles so cycles pairs are searched for during the decomposition process

Observation

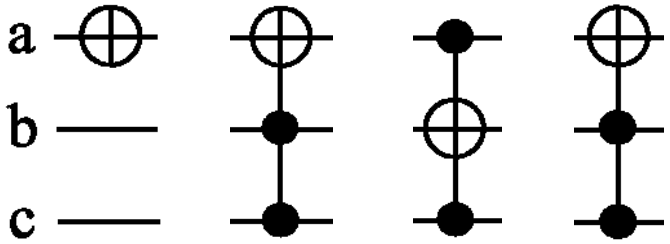
- The difference between average cost of an optimal circuit and average cost of actual designs generated by all known algorithms increases when the number of variables grows



Transformation-based approaches

Miller, Maslov, Dueck – DAC 2003

in	out	S1	S2	S3	S3
000	001	000	000	000	000
001	000	001	001	001	001
010	011	010	010	010	010
011	010	011	011	011	011
100	101	100	100	100	100
101	111	110	111	101	101
110	100	101	101	111	110
111	110	111	110	110	111



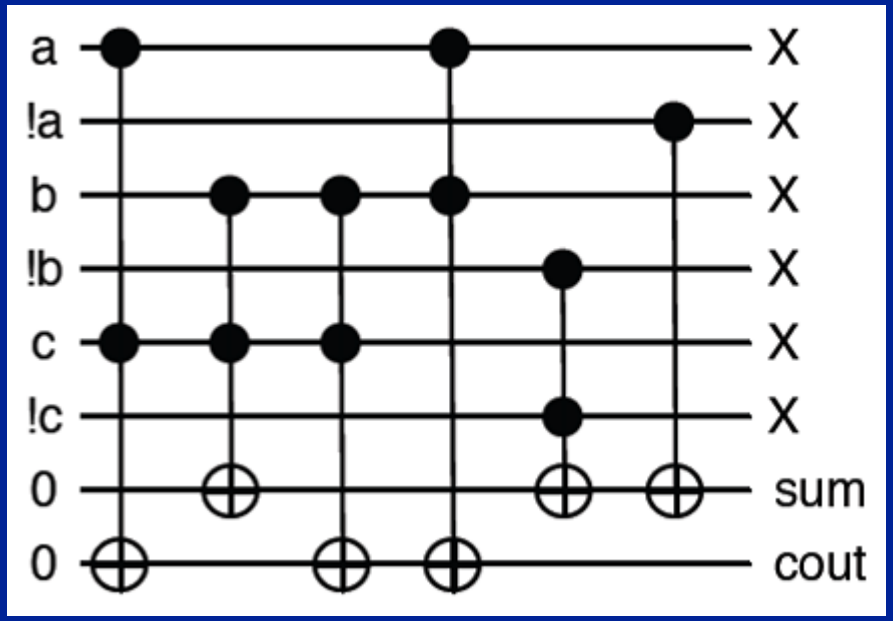
Gate selection is based on one row only

ESOP-based approaches

Fazel, Thornton, Rice – PACRIM 2007

$$cout = ac \oplus bc \oplus ab$$

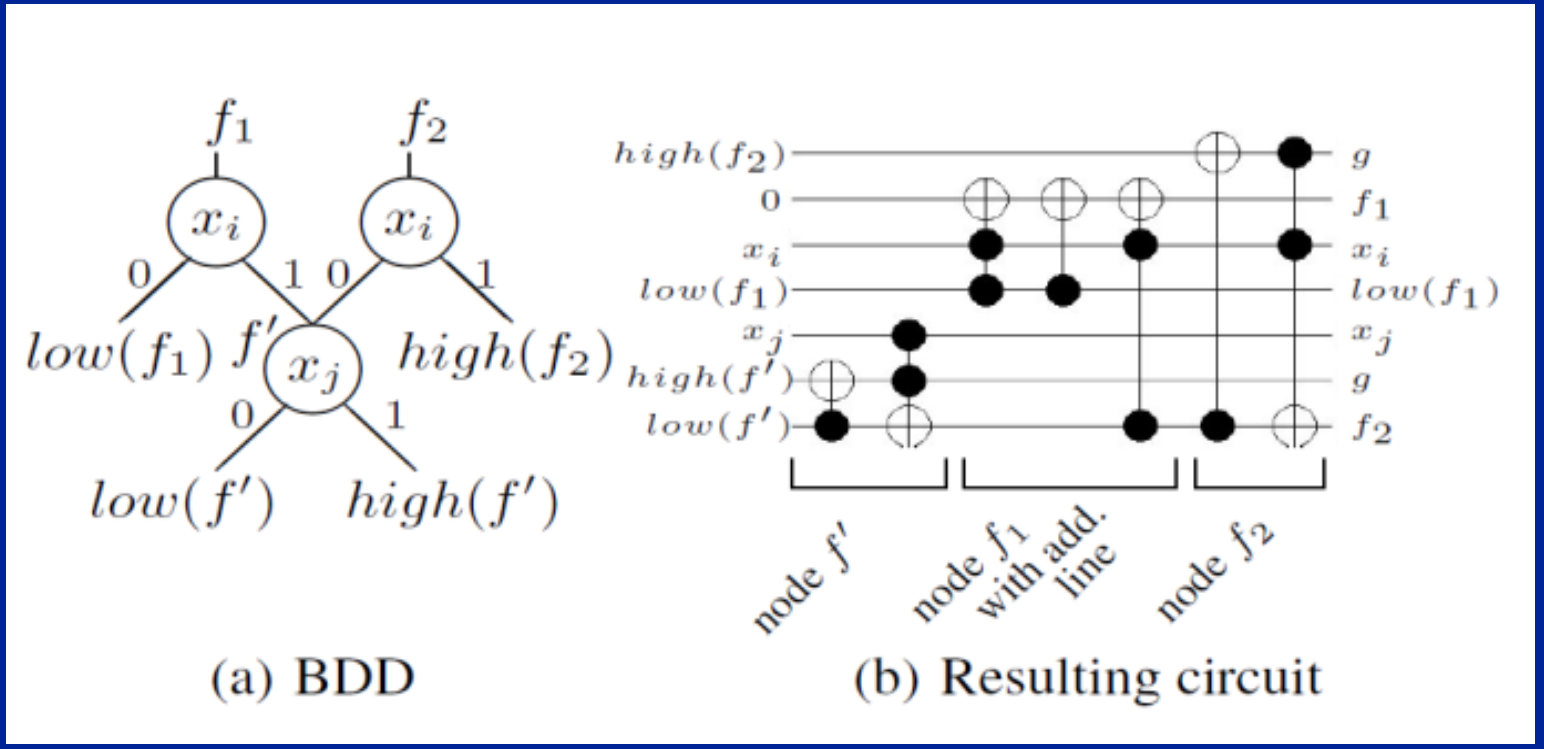
$$sum = bc \oplus b'c' \oplus a'$$



Gate selection is based on one product only

BDD-based approaches

R. Wille, R. Drechsler – DAC 2009



Subcircuit selection is based on one node only

Numerical results (using RevKit)

Format of results: #L GC QC

	hwb5	hwb6
Our result	5 38 80	6 48 98
TB (RevKit)	5 54 570	6 144 3364
MP	5 48 504	6 145 3013
CSP-2	5 48 492	6 137 2808
BDD (RevKit)	27 87 267	46 161 513
Neg-Davio (RevKit)	29 94 294	44 173 541
Pos-Davio (RevKit)	27 98 262	50 169 457

Advantages and drawbacks

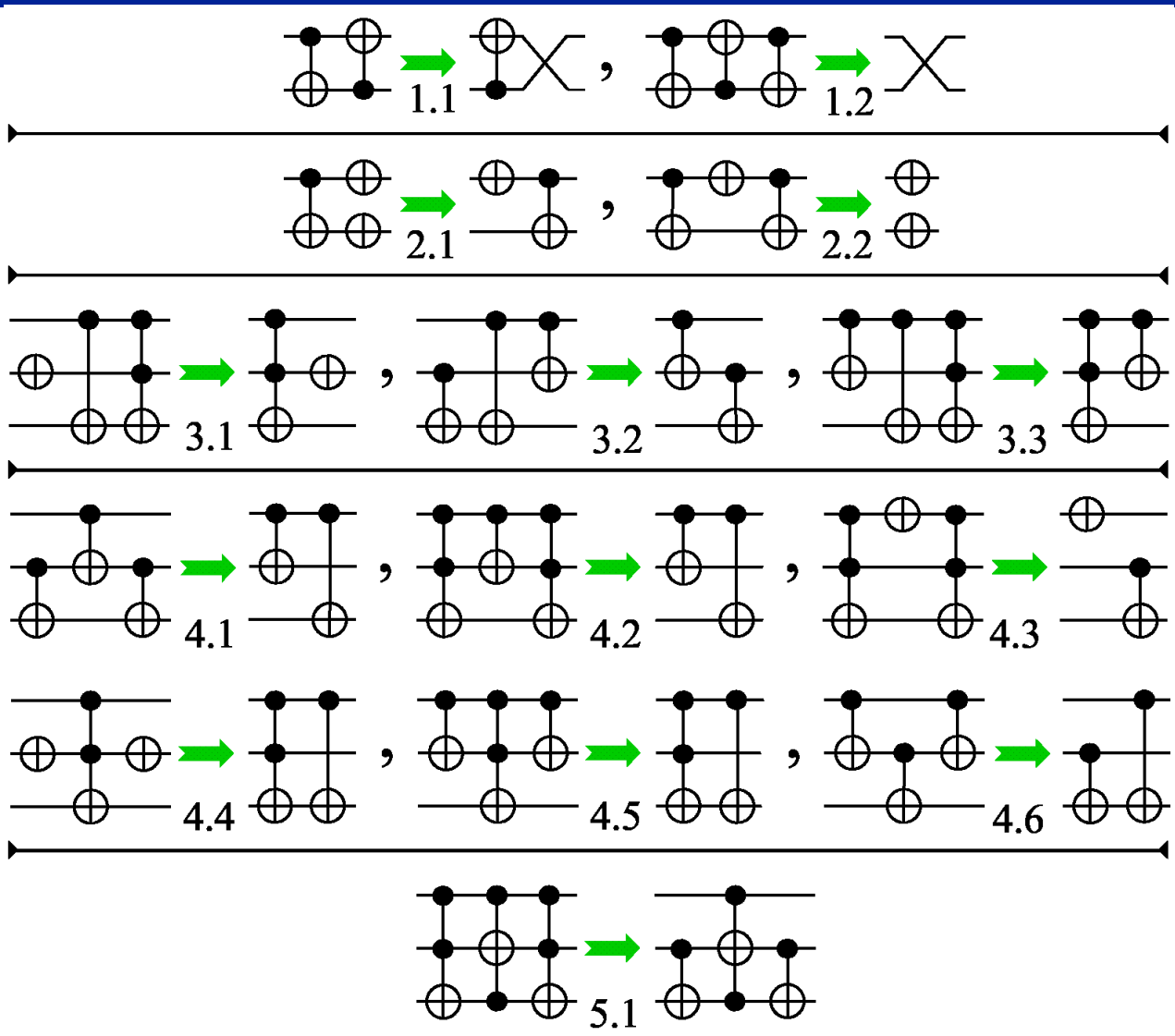
- Advantages
  - Convergence
  - Scalability
  - Fast synthesis
- Drawbacks
  - Algorithms are based on local decisions
  - Many redundant gates
  - Large (maximal-size) gates (excessive QC) in some approaches
  - Excessive number of additional lines

Result - time-consuming post-synthesis reduction, e.g. reduction of the number of lines using RevKit leads to spectacular growth in GC and QC:

	hwb4			mini-alu			hwb6		
BDD	15	39	123	15	36	112	46	161	513
BDD - reduction of lines	10	389	10138	9	681	31408	31	3305	125593

Templates

D. Maslov, D. M. Miller, G. Dueck 2003-2007



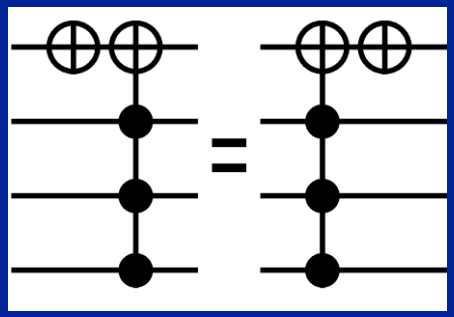
Reduction of maximal–size gates

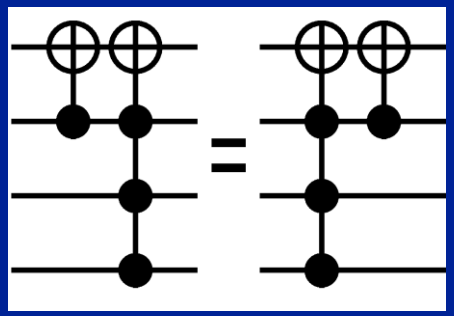
- **Shende, Prasad, Markov, Hayes – TCAD 2003**

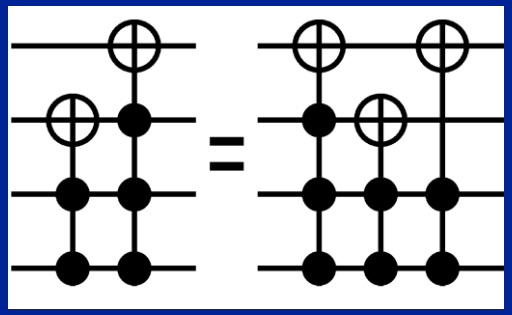
  From the theoretical results proved in this paper it follows that

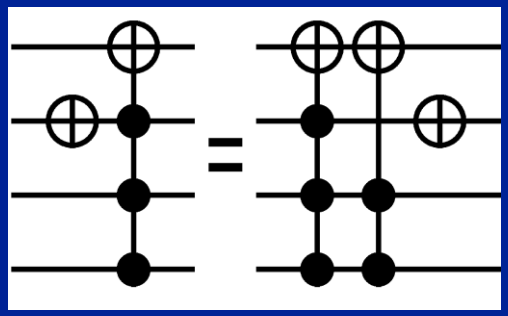
- **the number of maximal–size gates in any reversible circuit can be reduced to:**
- **(a) zero for even functions (i.e. with even number of transpositions)**
- **(b) one for odd functions (i.e. with odd number of transpositions)**

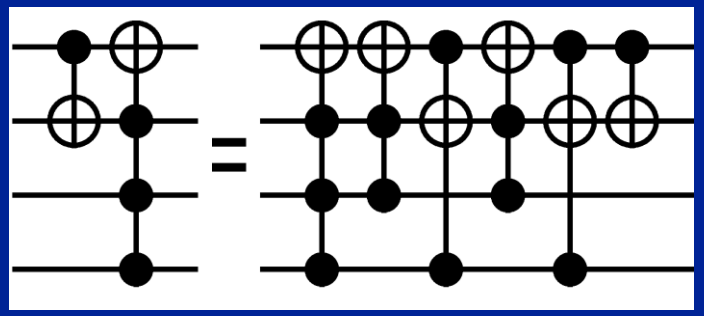
Templates for moving T4 gates

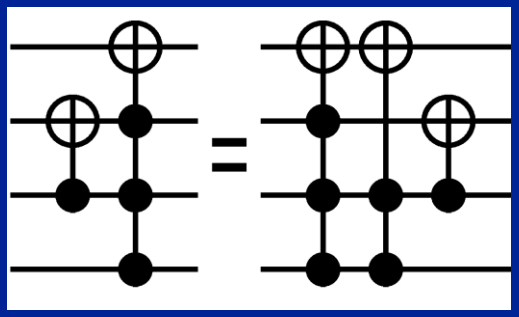


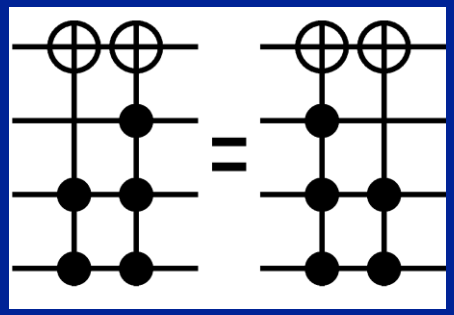


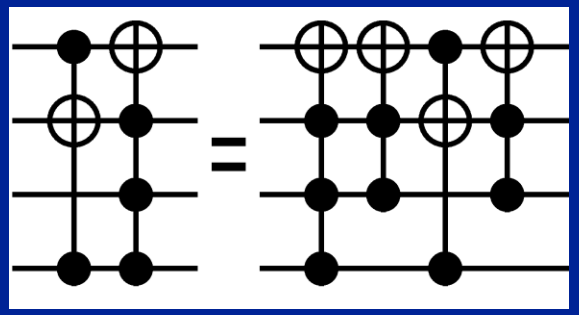




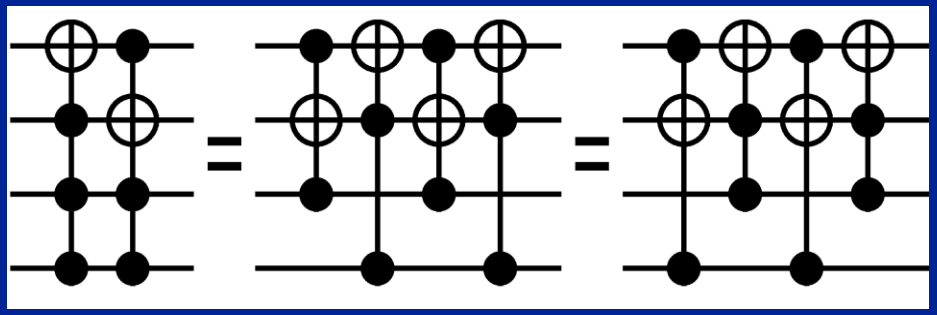


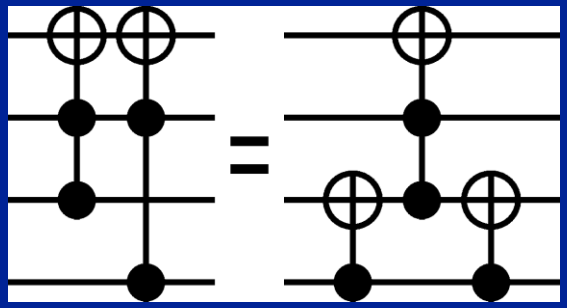


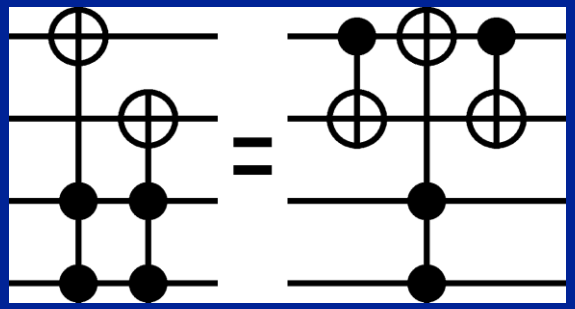




Templates for reducing QC for pairs of Toffoli gates

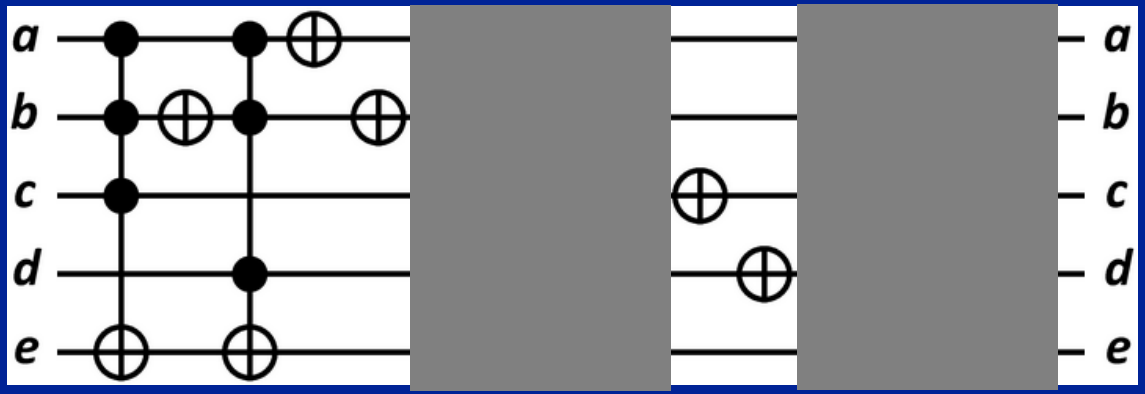


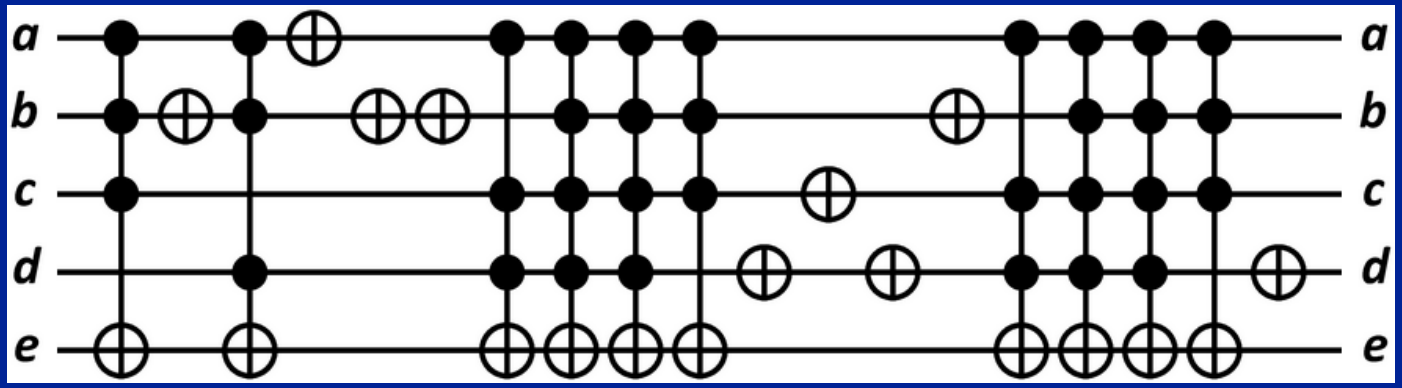




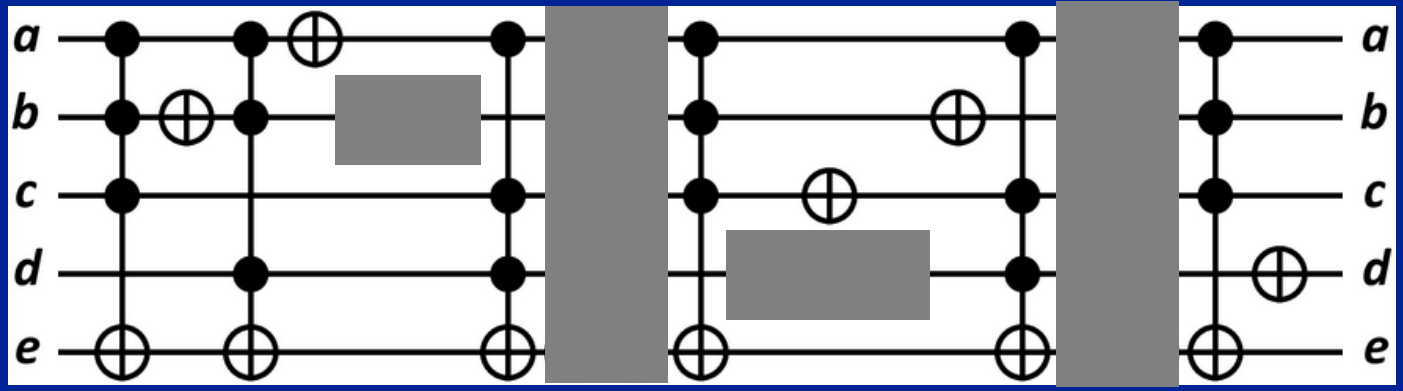
Application of templates: example (1/9)

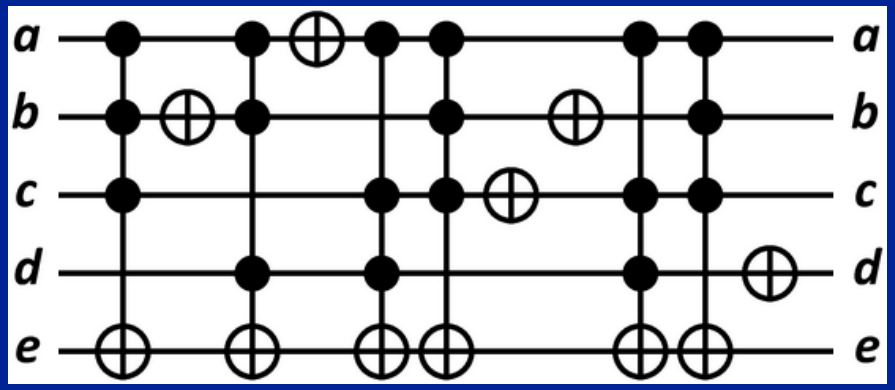
Hamza, Dueck – Workshop on Rev. Comp. 2010



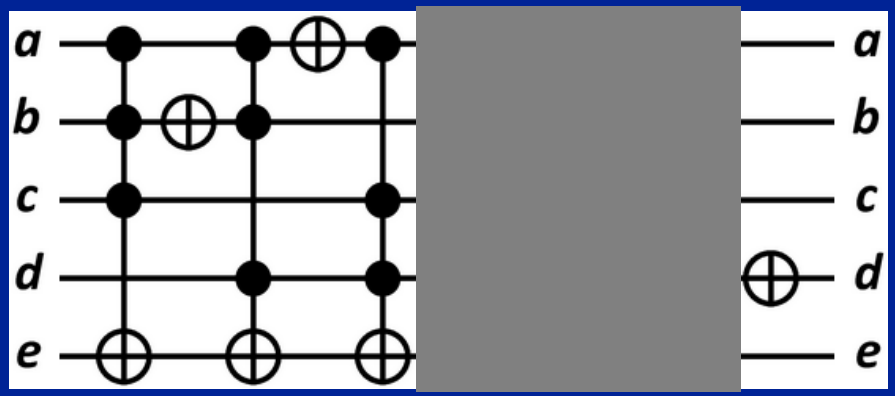


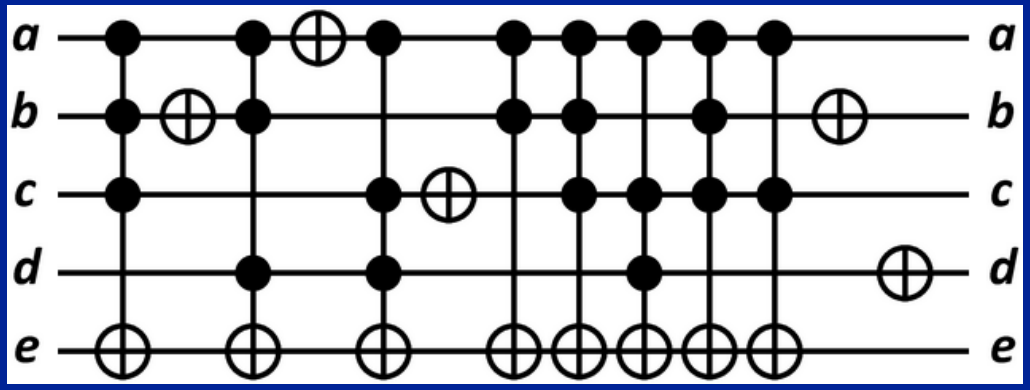
Application of templates: example (2/9)



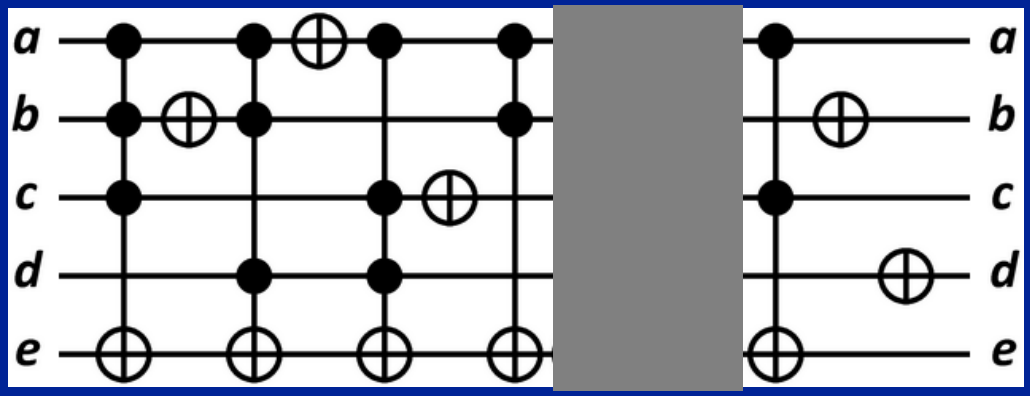


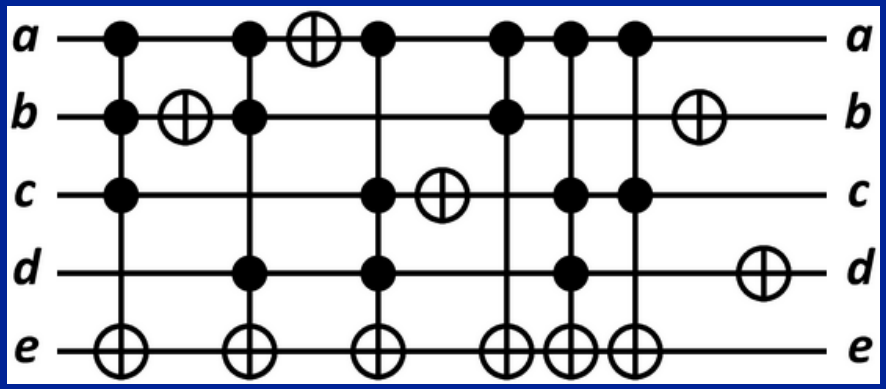
Application of templates: example (3/9)



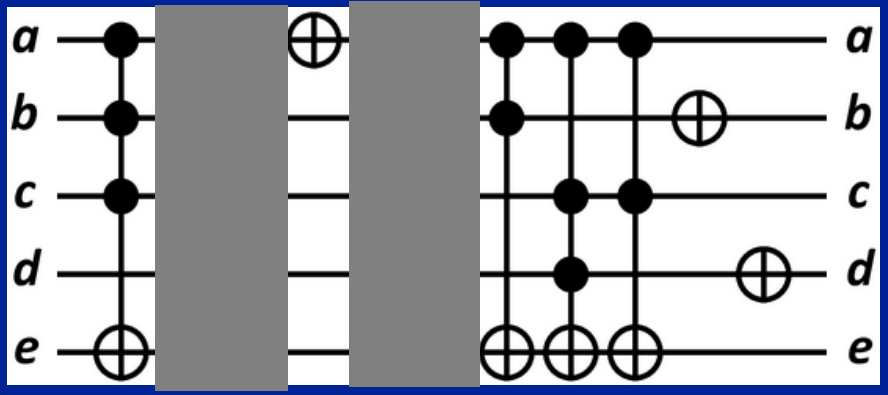


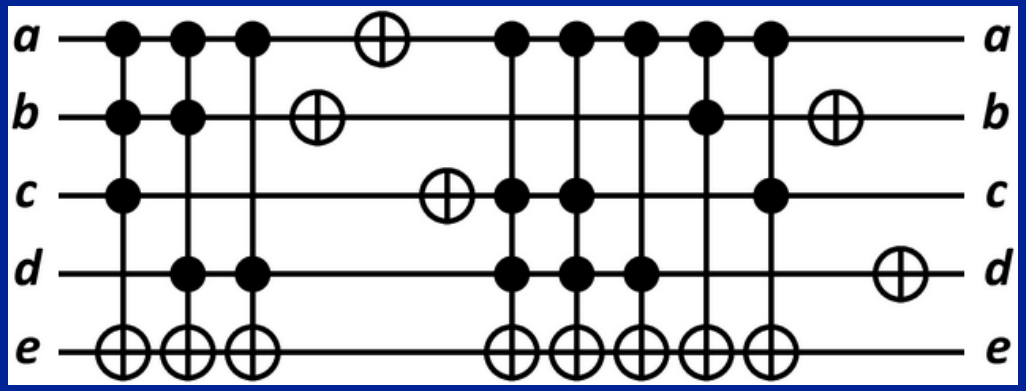
Application of templates: example (4/9)



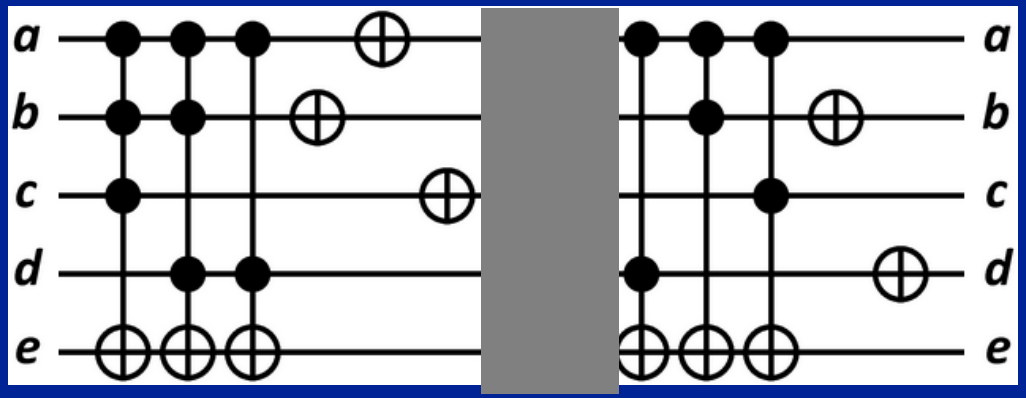


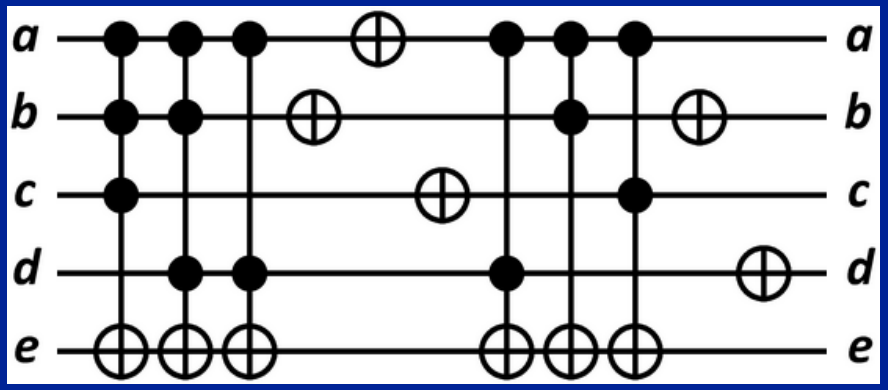
Application of templates: example (5/9)



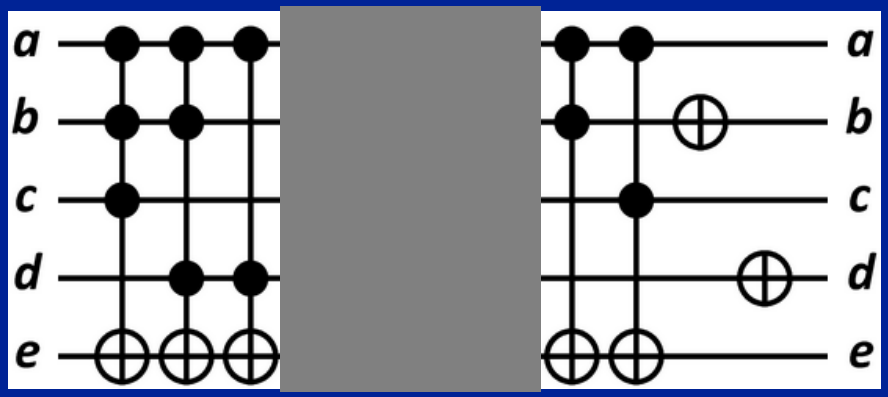


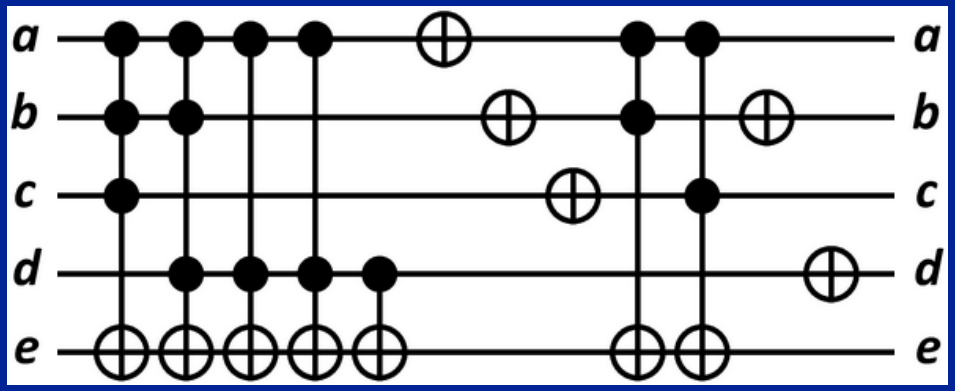
Application of templates: example (6/9)



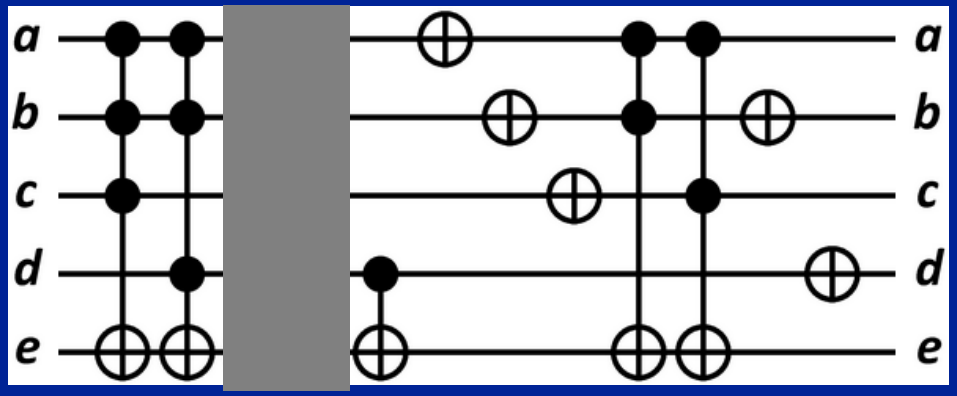


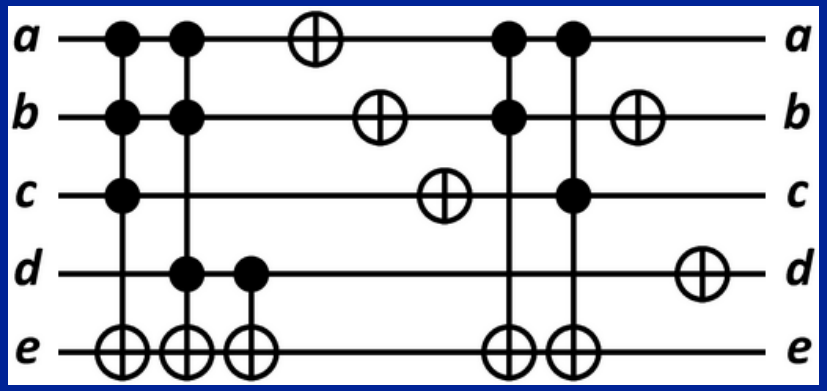
Application of templates: example (7/9)



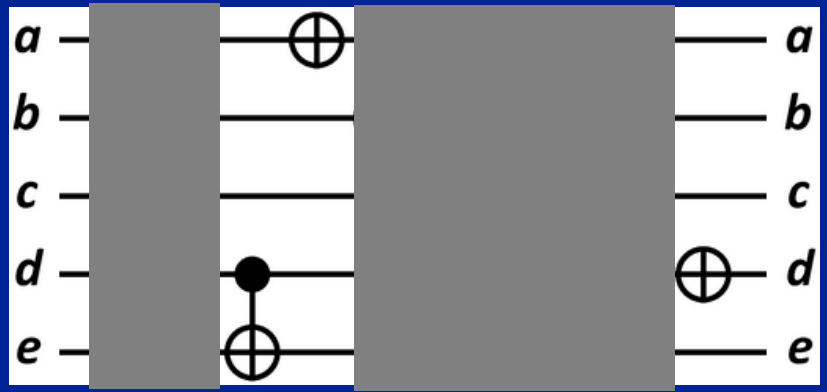


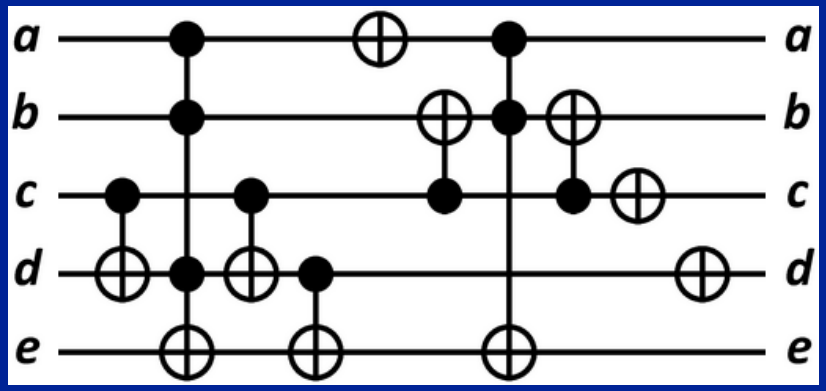
Application of templates: example (8/9)





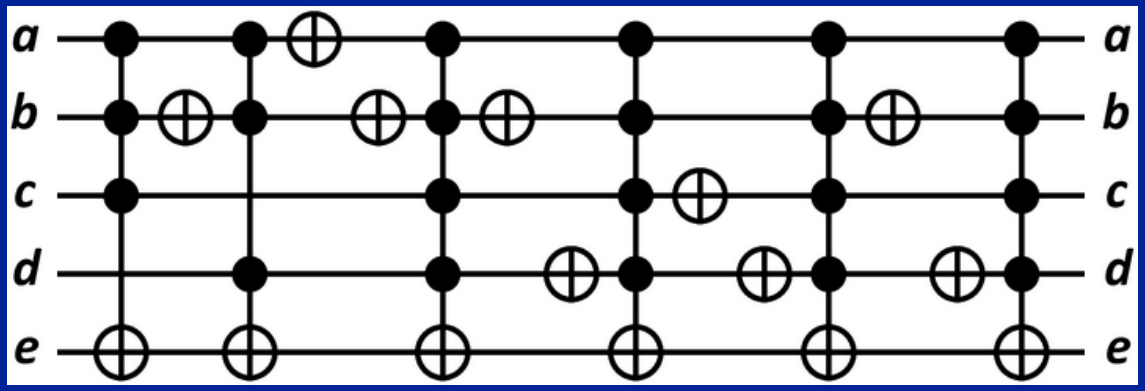
Application of templates: example (9/9)



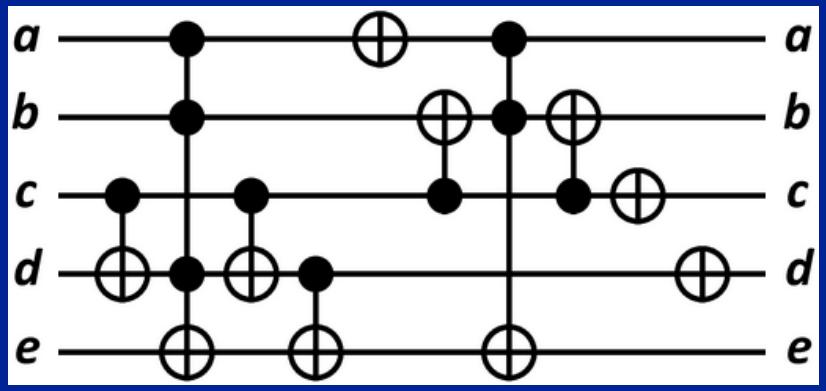


Application of templates: final result

GC=15 QC= 151

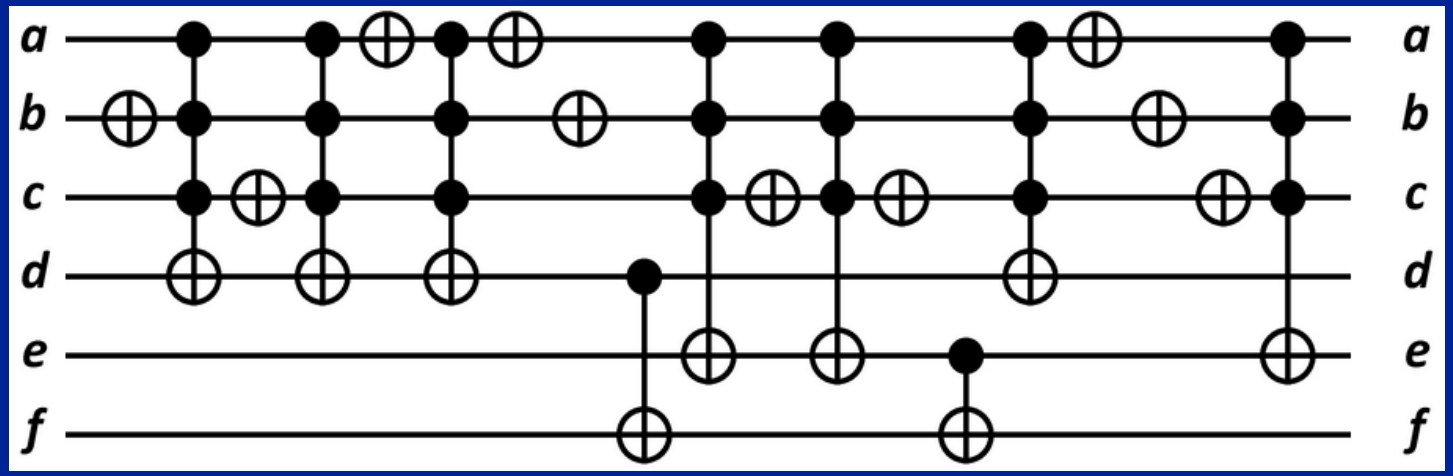


GC=10 QC= 26



Application of templates: 2nd example

GC=19 QC= 103



GC=15 QC= 55