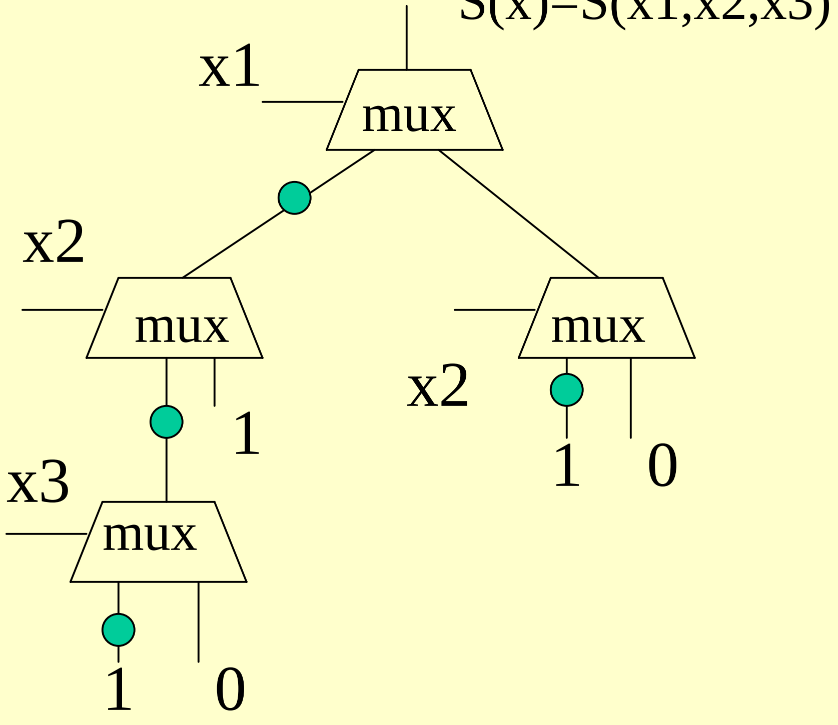
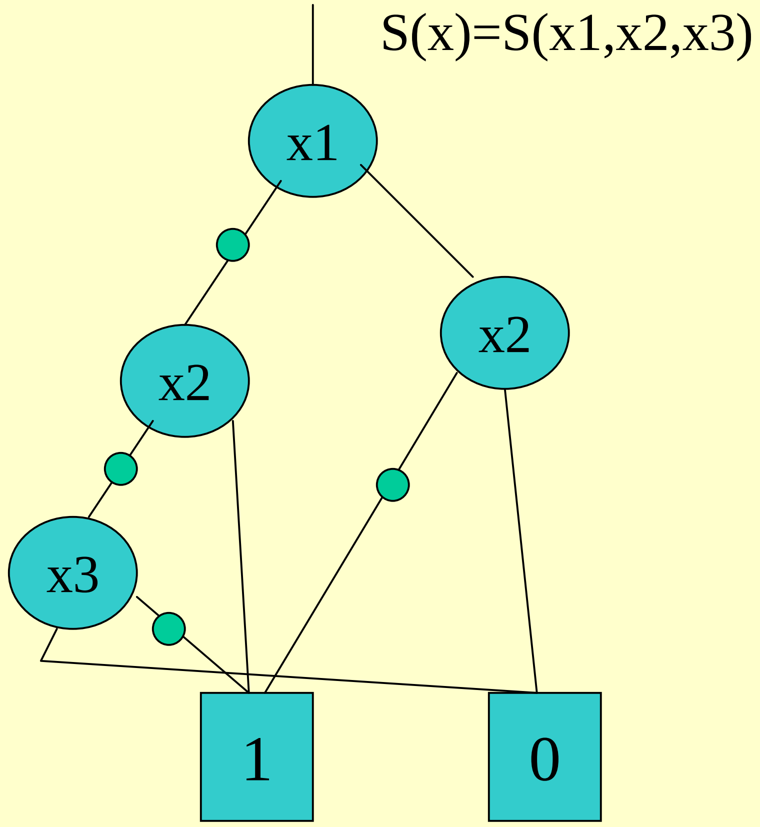
Lattice diagrams and regular layouts

As we remember, BDDs are easily mapped to circuits

Let us illustrate BDDs with the help of the examples.

Note that BDD nodes are in one to one correspondece with the gate of the MUX circuit.

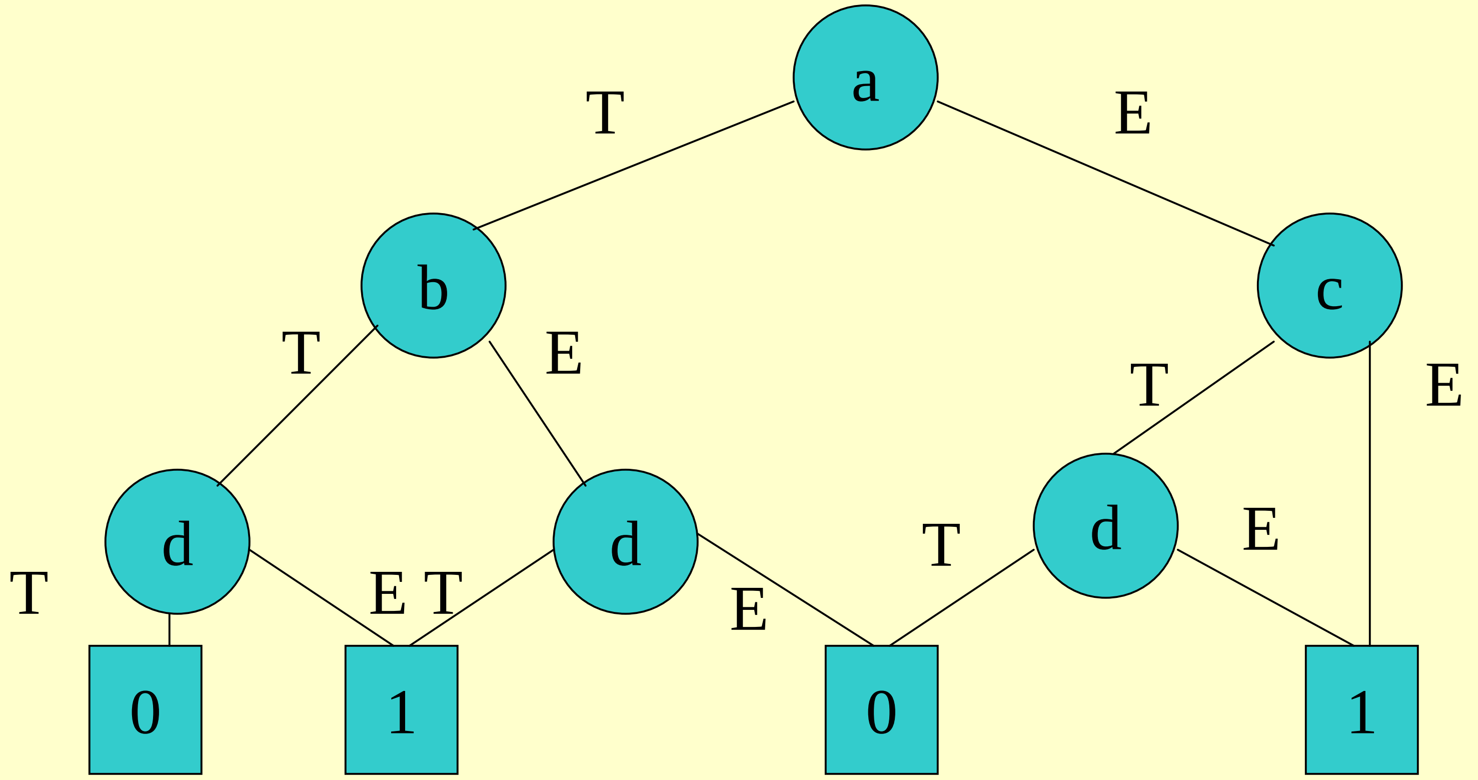




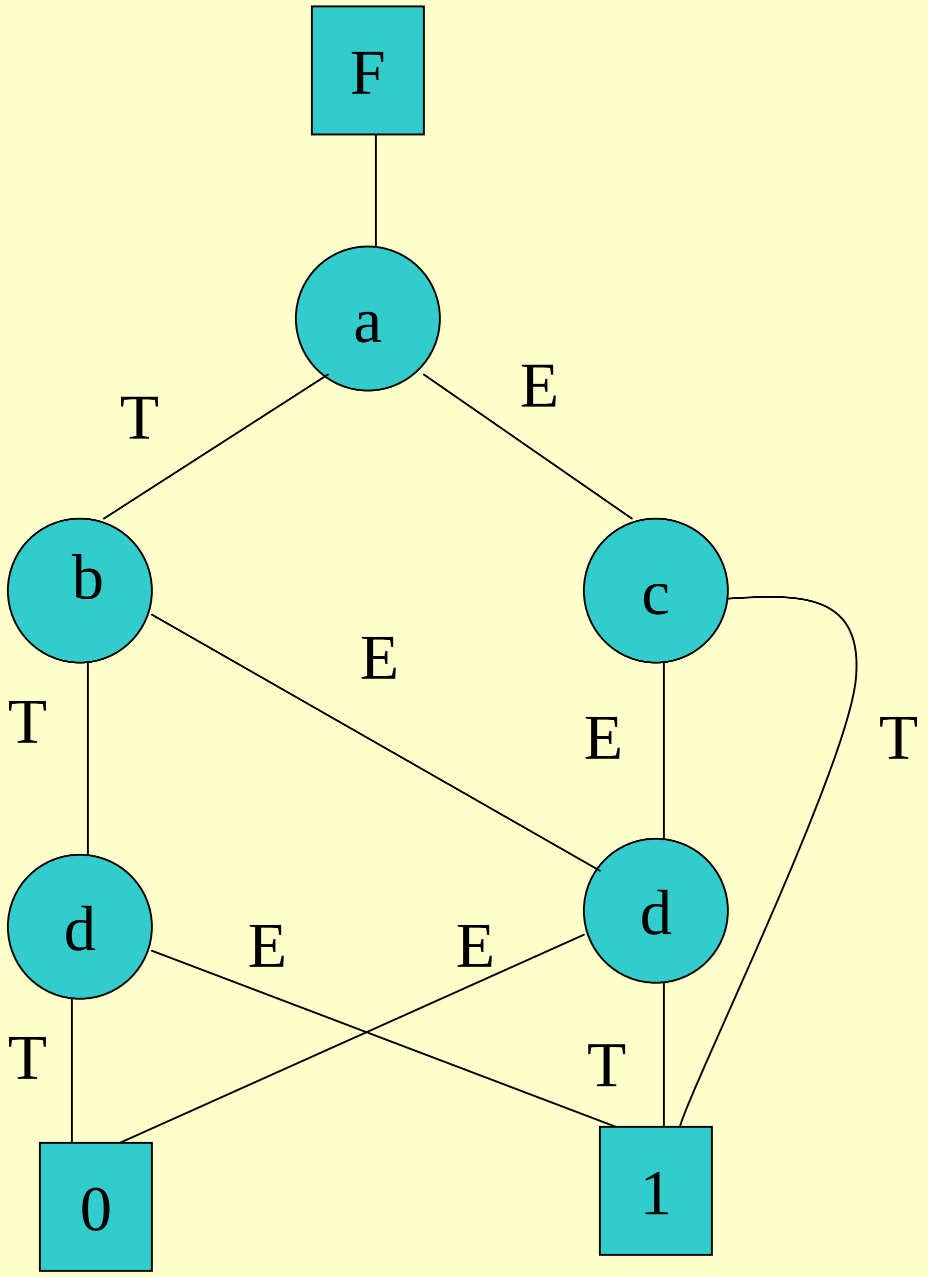
EXAMPLE

Represent following function as a BDD and ROBDD and represents multiplexor circuit for the function.

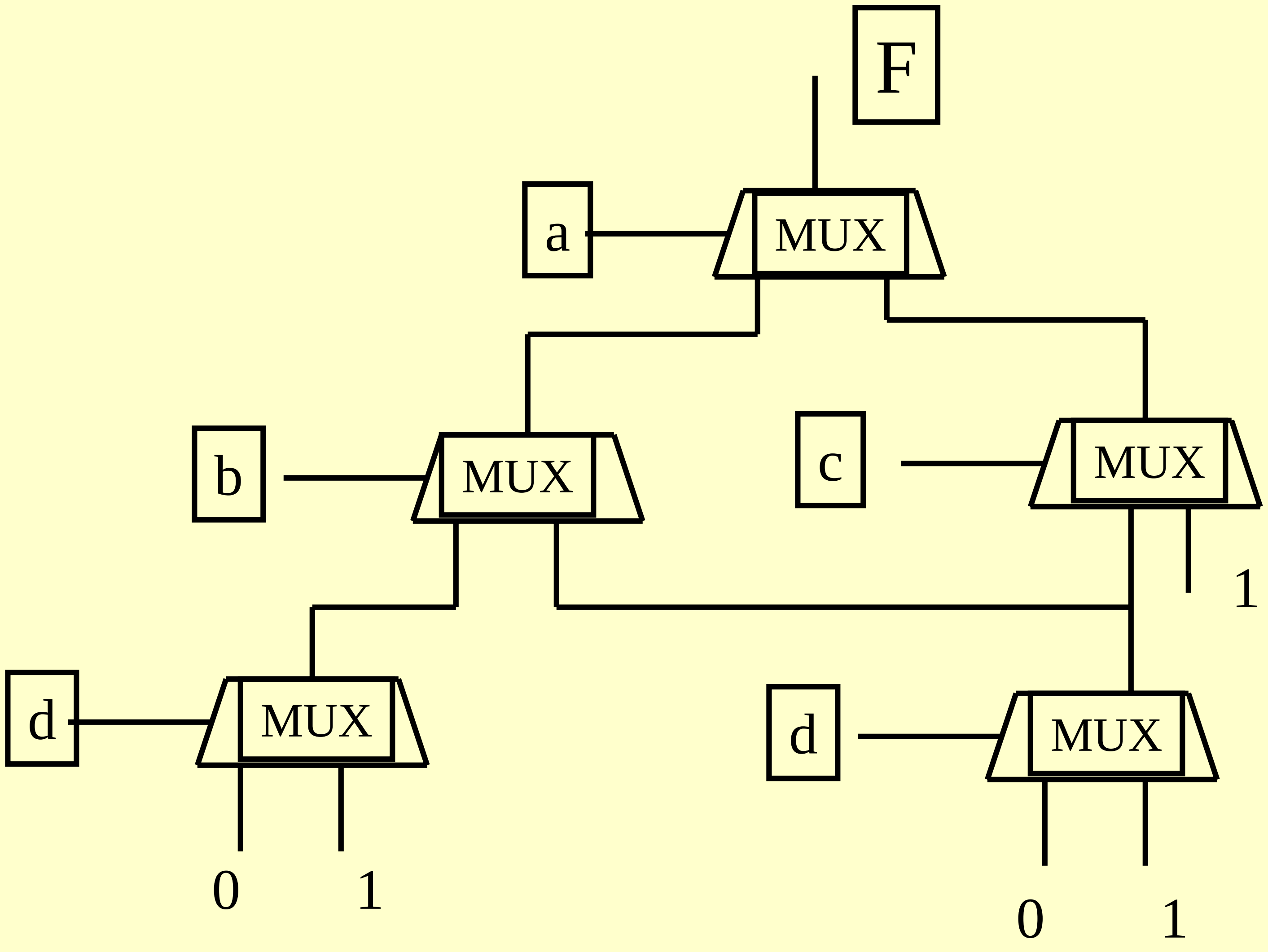
F=abd'+ab'd+a'c+a'c'd



ROBDD



MULTIPLEXOR CIRCUIT



Are there other diagrams like this?

- We want to minimize the area
- Maximize the speed
- Improve the testability
- Allow for easy synthesis and direct link to layout

LATTICE DIAGRAMS

•These diagrams are created for both symmetric (no variable repetition) and non-symmetric functions.

•Non-symmetric functions require in general variable repetition in levels.

Types of diagrams

S BDD $\longrightarrow$ Shannon Lattice

pD Functional Decision Diagram $\longrightarrow$ Positive Davio Lattice

nD Negative Functional Decision Diagram $\longrightarrow$ Negative Davio Lattice

S, pD, nD Kronecker Functional Decision Diagram $\longrightarrow$ Kronecker Davio Lattice

Shannon Lattice

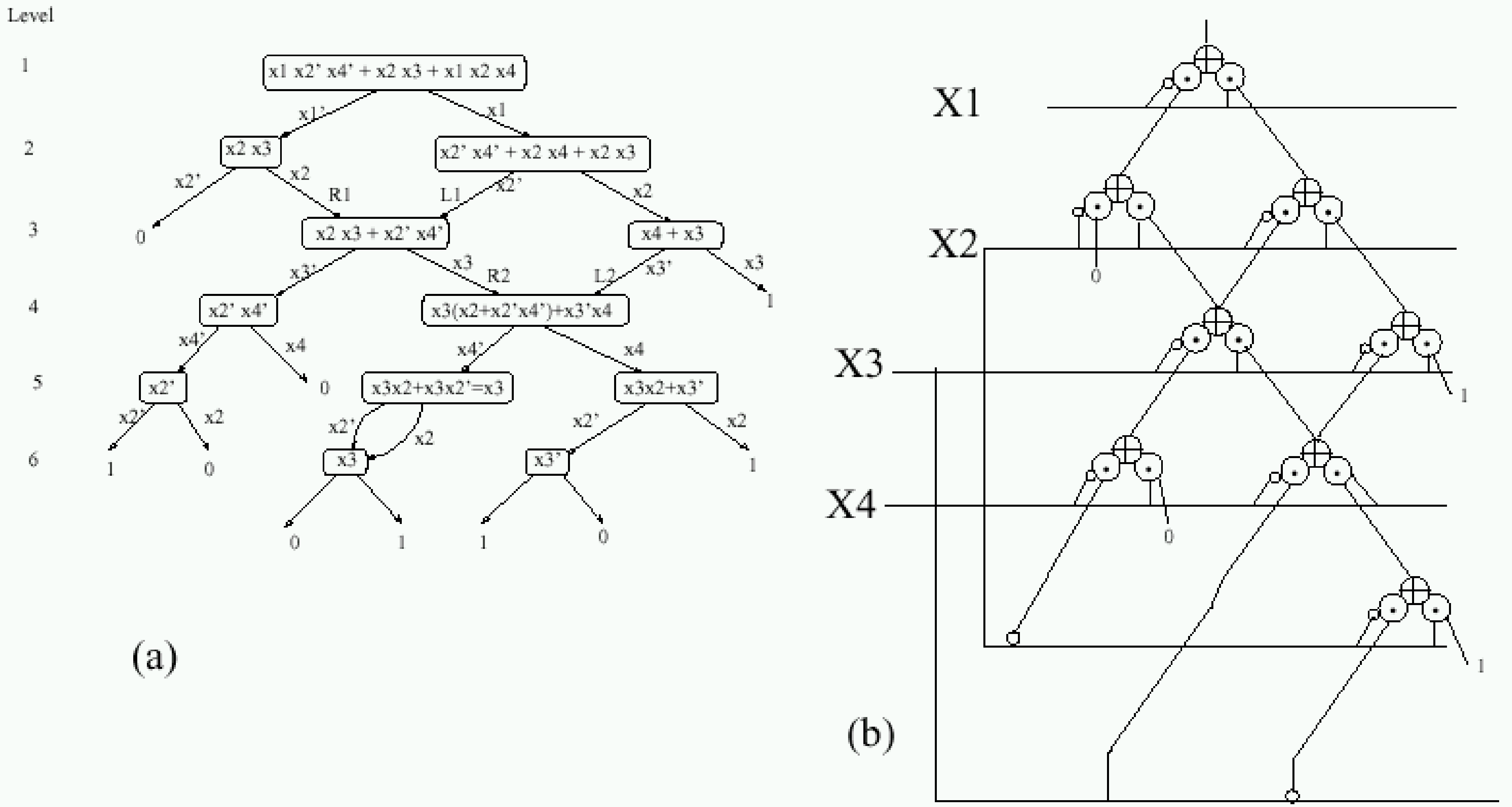
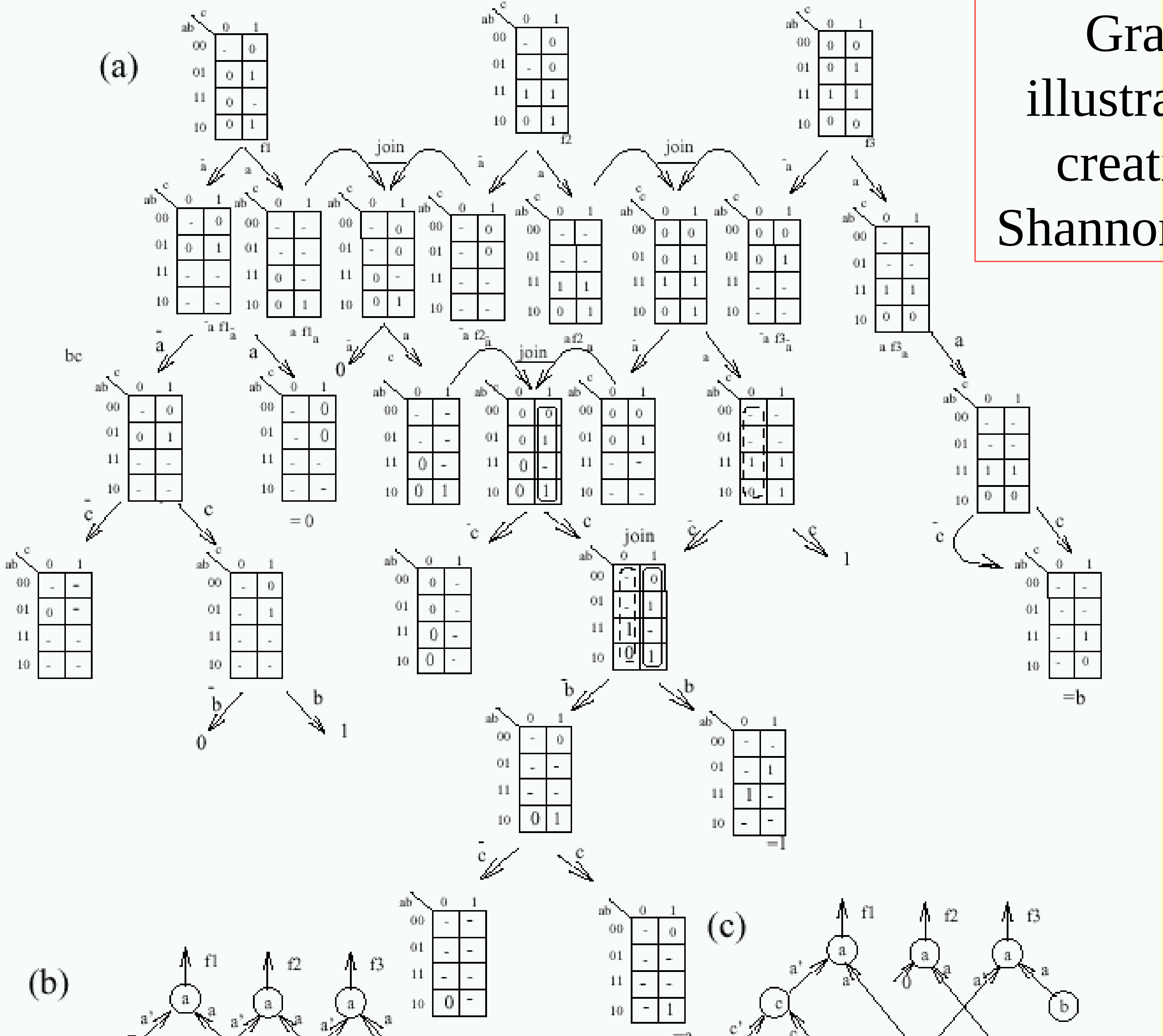


Figure 2: *Method for creation of a Single-Output Shannon Lattice for a completely specified function represented by ON cubes, (b) the circuit corresponding to a) before the propagation of constants.*

Graphic illustration of creation of Shannon Lattice



Creating Shannon Lattices by cofactoring and joining (Cont)

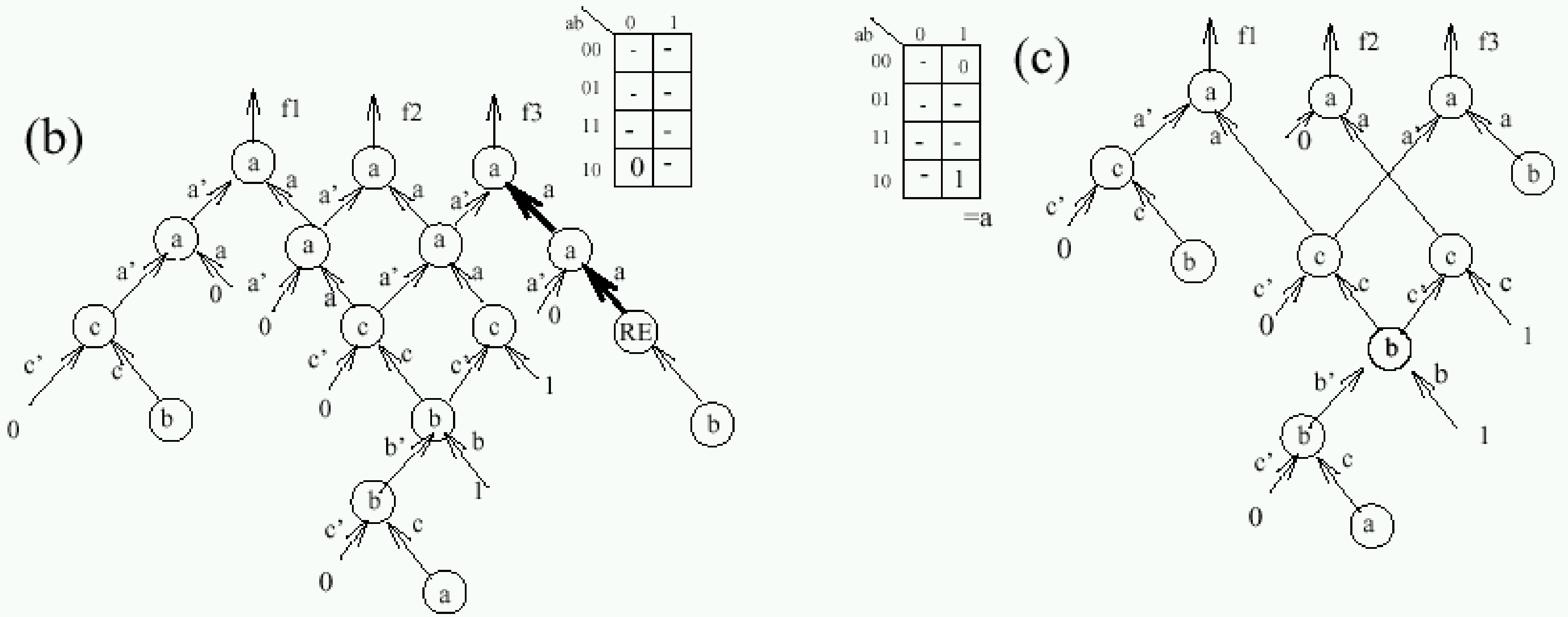


Figure 3: *The method to create the Multi-Output Ordered Shannon Lattice Diagram for an incompletely specified function of three outputs, (a) the method to create the expansions and joining cofactors, (b) the Multi-Output Ordered Shannon Lattice Diagram derived using method from (a), (c) the Folded Shannon Lattice Diagram obtained after logic/layout simplification of the Ordered Shannon Lattice Diagram from b)*.

Kroneckers Lattices and their joining rules

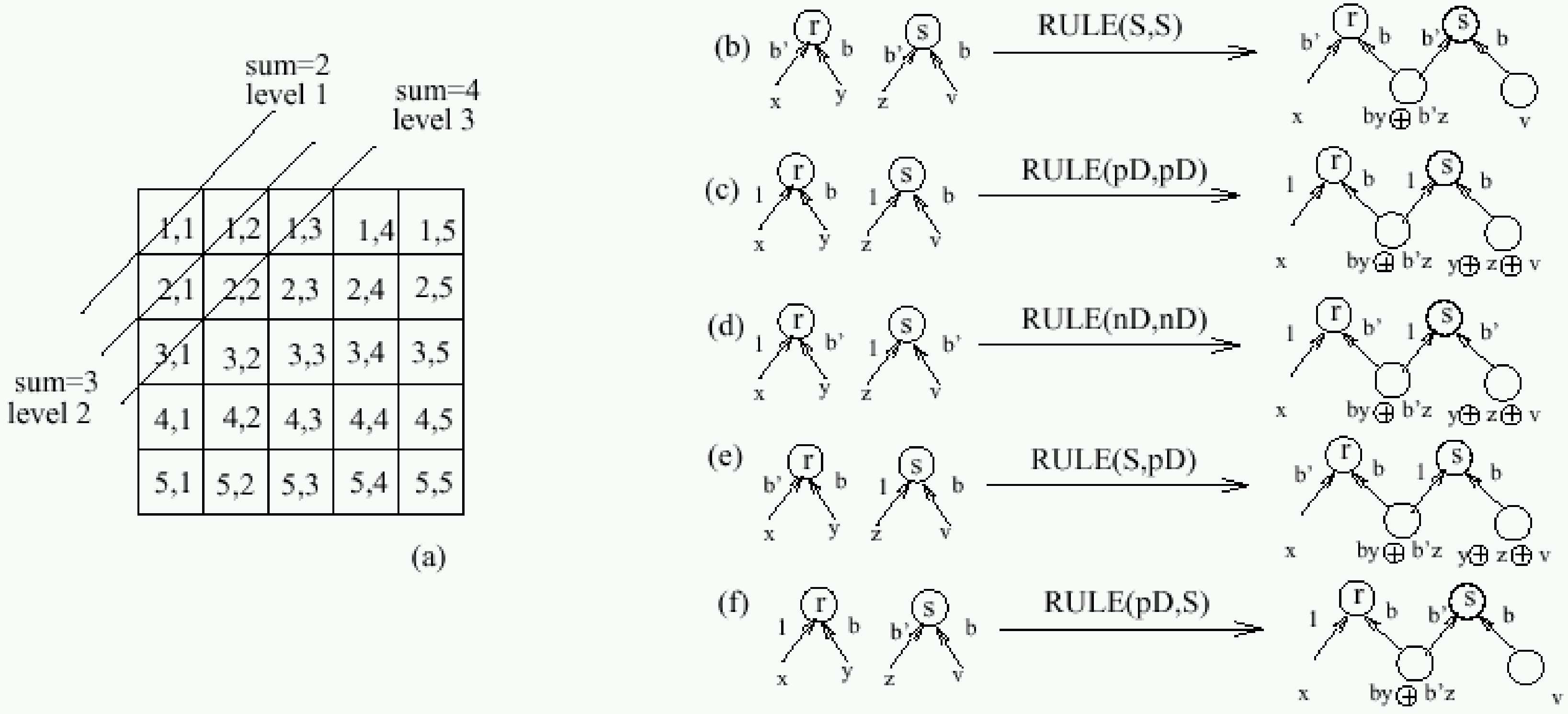
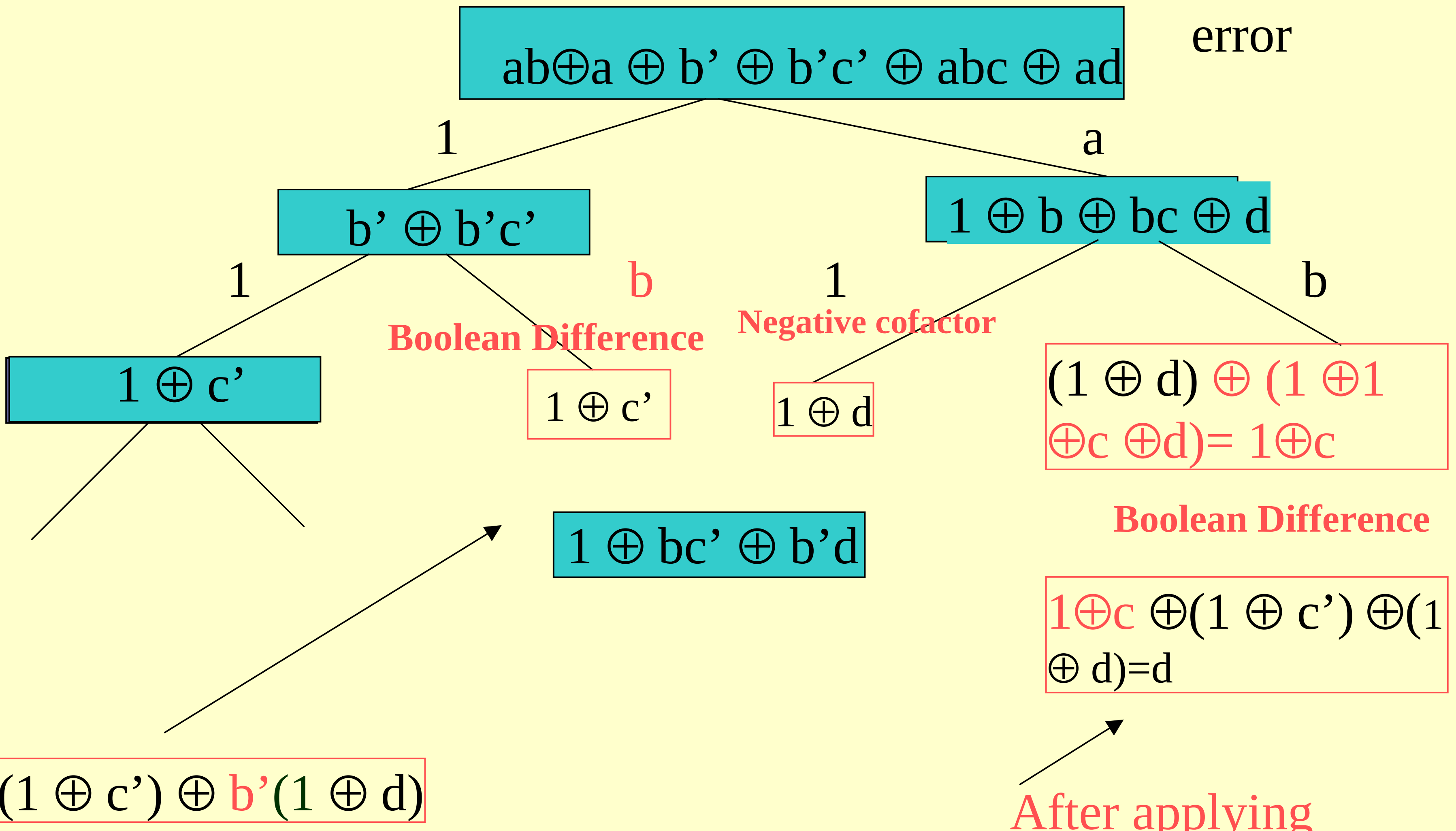


Figure 1: *(a) Array to explain Lattice concepts. (b) - (f) Joining Rules to create Kronecker Lattice Diagrams and related diagrams. Left side - before joining non-isomorphic nodes, right side - after joining nodes and possibly, propagating correction to the right predecessor of node s. Corrections are propagated only in rules (c), (d), and (e).*

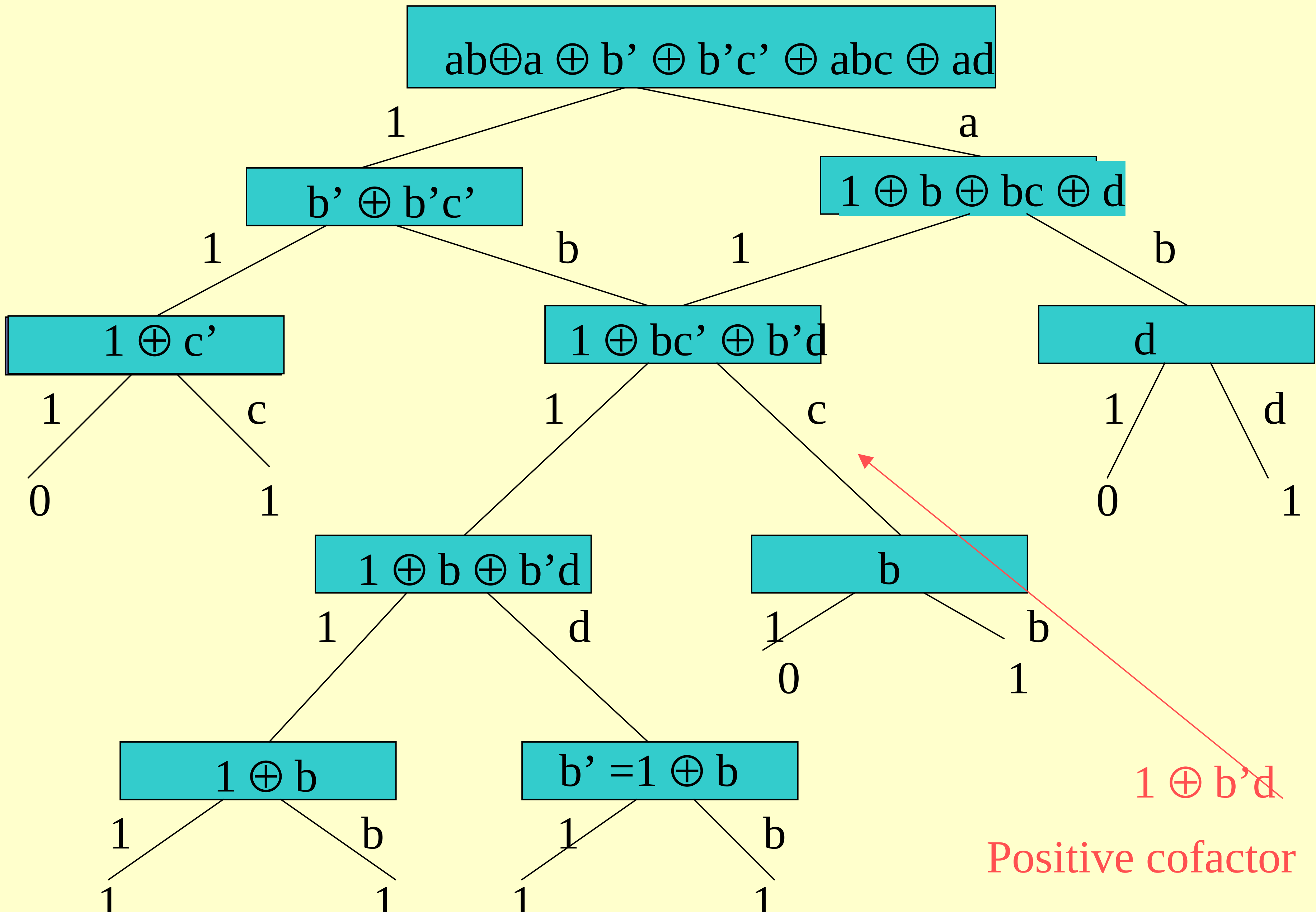
EXAMPLE OF Positive Davio Lattice



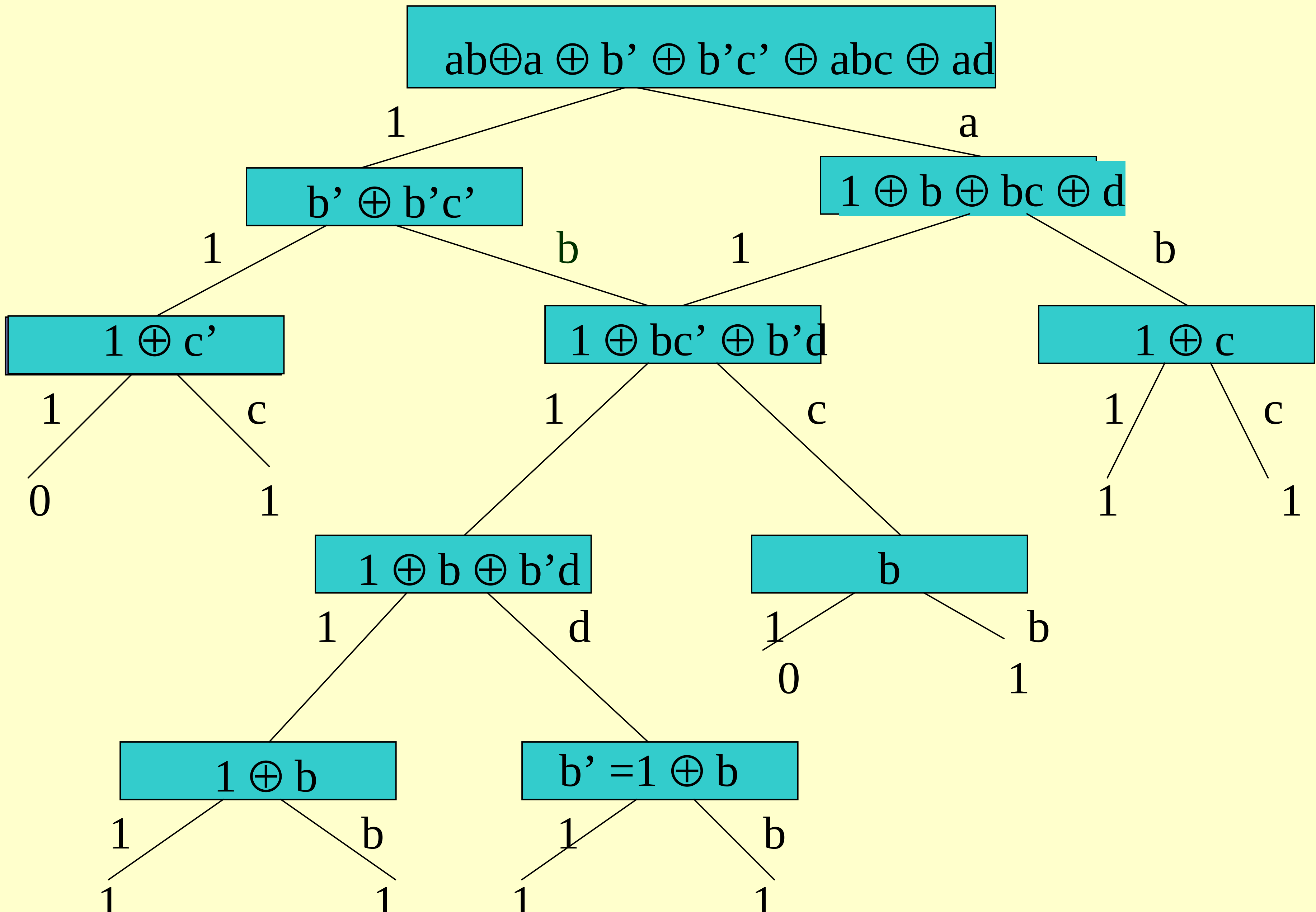
After applying correction rule

Apply rule (pD,pD)

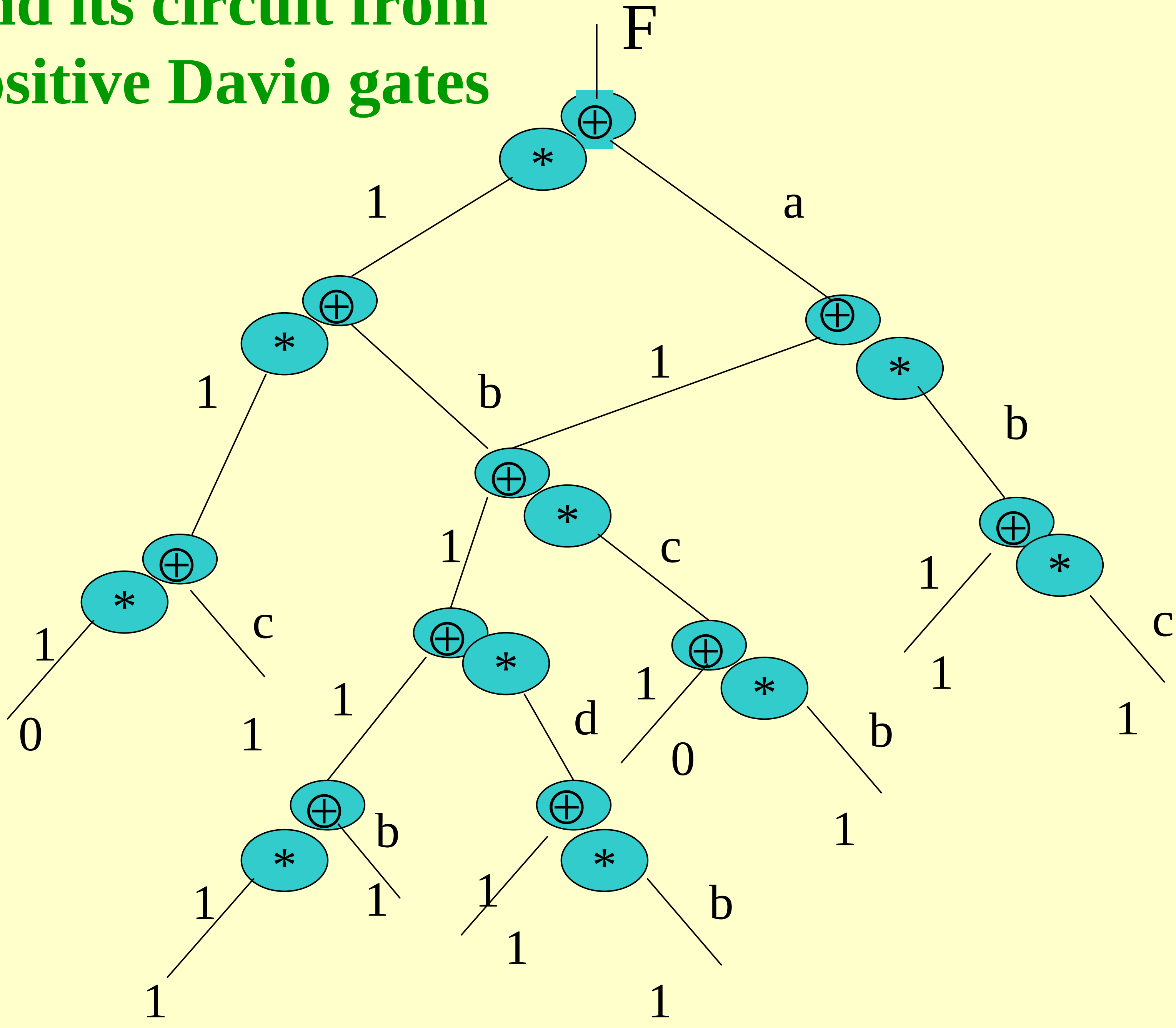
EXAMPLE OF Positive Davio Lattice(cont)



Final Positive Davio Lattice



And its circuit from Positive Davio gates



Example of Positive Davio Lattice

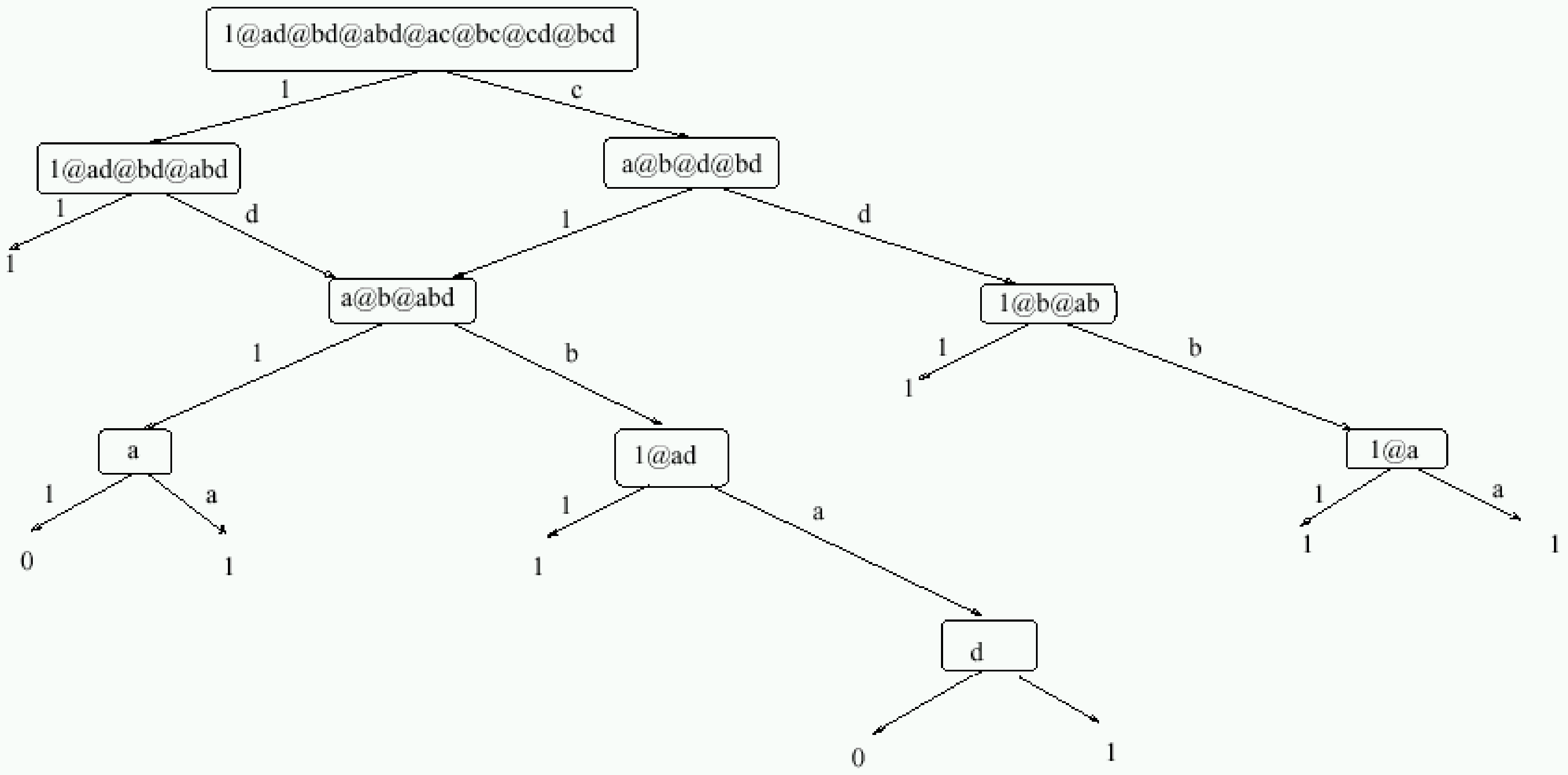
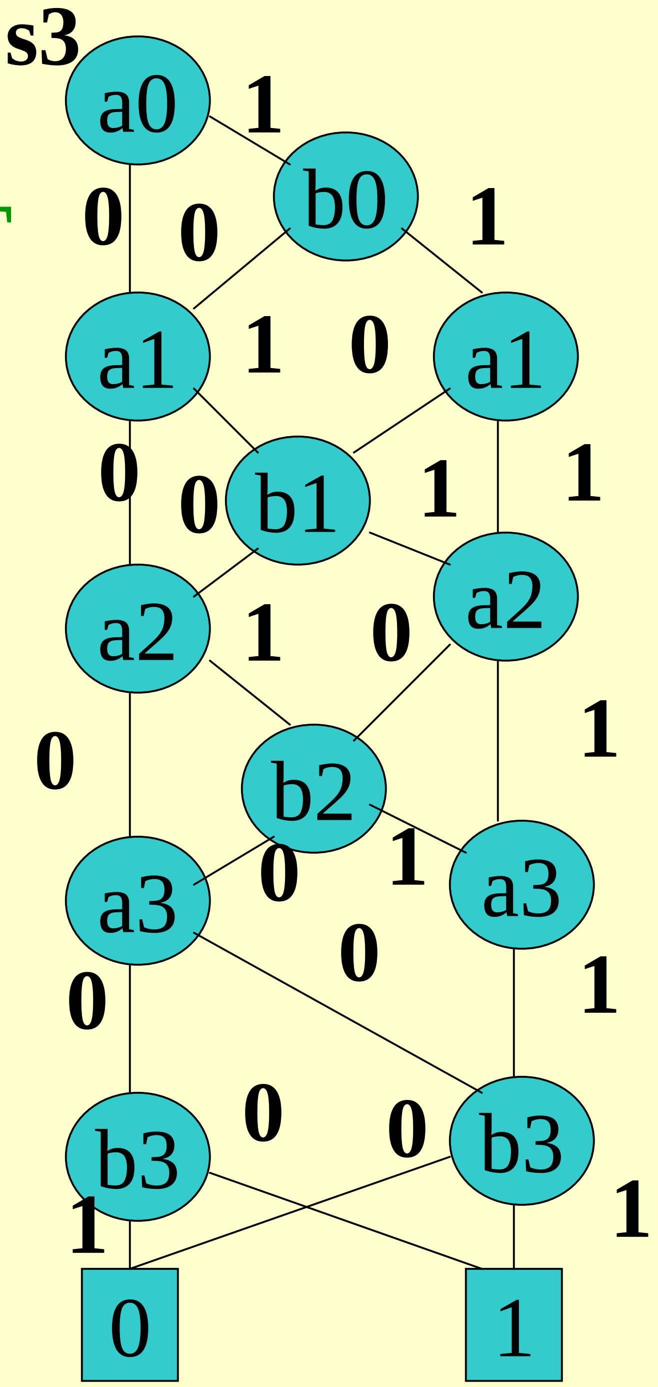


Figure 4: *The method to create Functional Lattice Diagram. Positive Davio expansions are used and (pD,pD) joinings are applied to the function represented in RM form.*

BDD FOR 4 BIT ADDER CIRCUIT

$$\begin{array}{r} a3\ a2\ a1\ a0 \\ +\ b3\ b2\ b1\ b0 \\ \hline s4\ s3\ s2\ s1\ s0 \end{array}$$



Think how you can use this diagram?

Free lattice diagrams

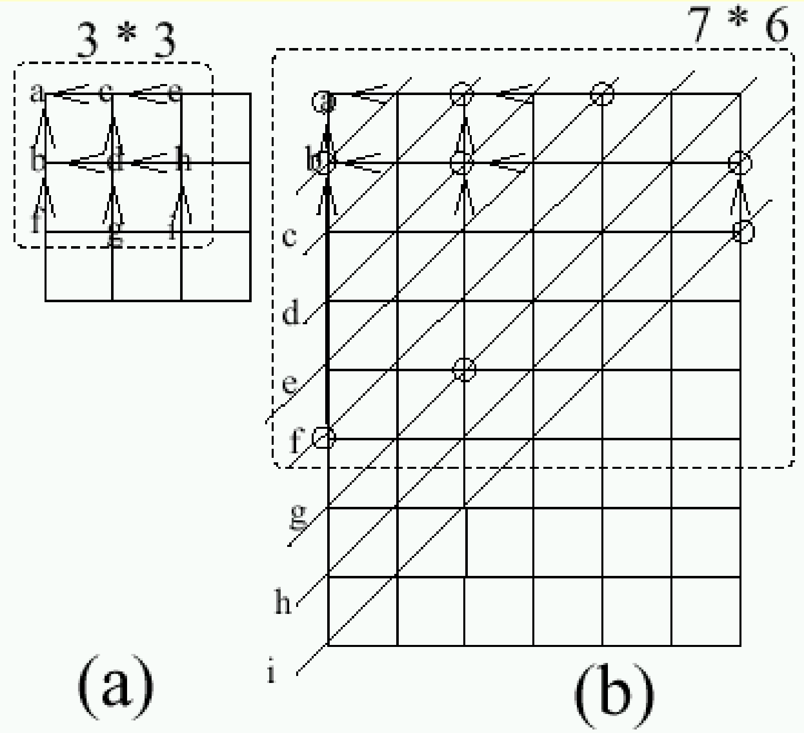
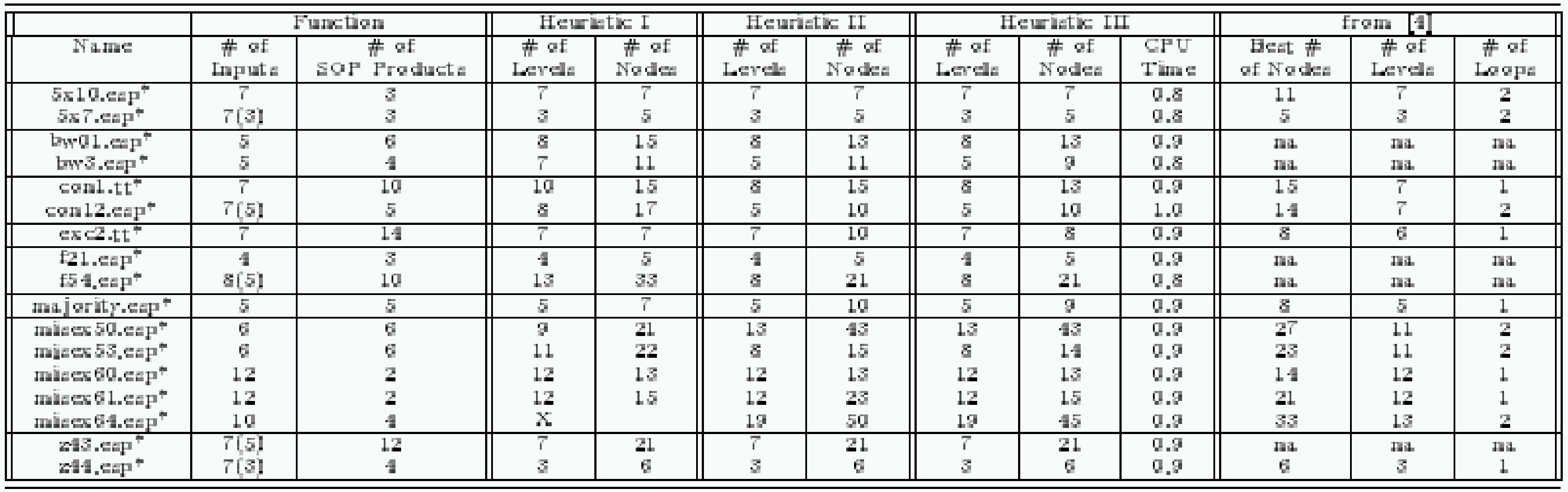


Figure 5: *Area comparison of folded and ordered Lattices for the same function: (a) new approach of Folded Lattice Diagram with every input available at every node (more complex routing), (b) PSBDD and Ordered Lattice realization with the same variables in diagonal buses.*

Experimental designs

- How many times variables must be repeated for practical circuits?

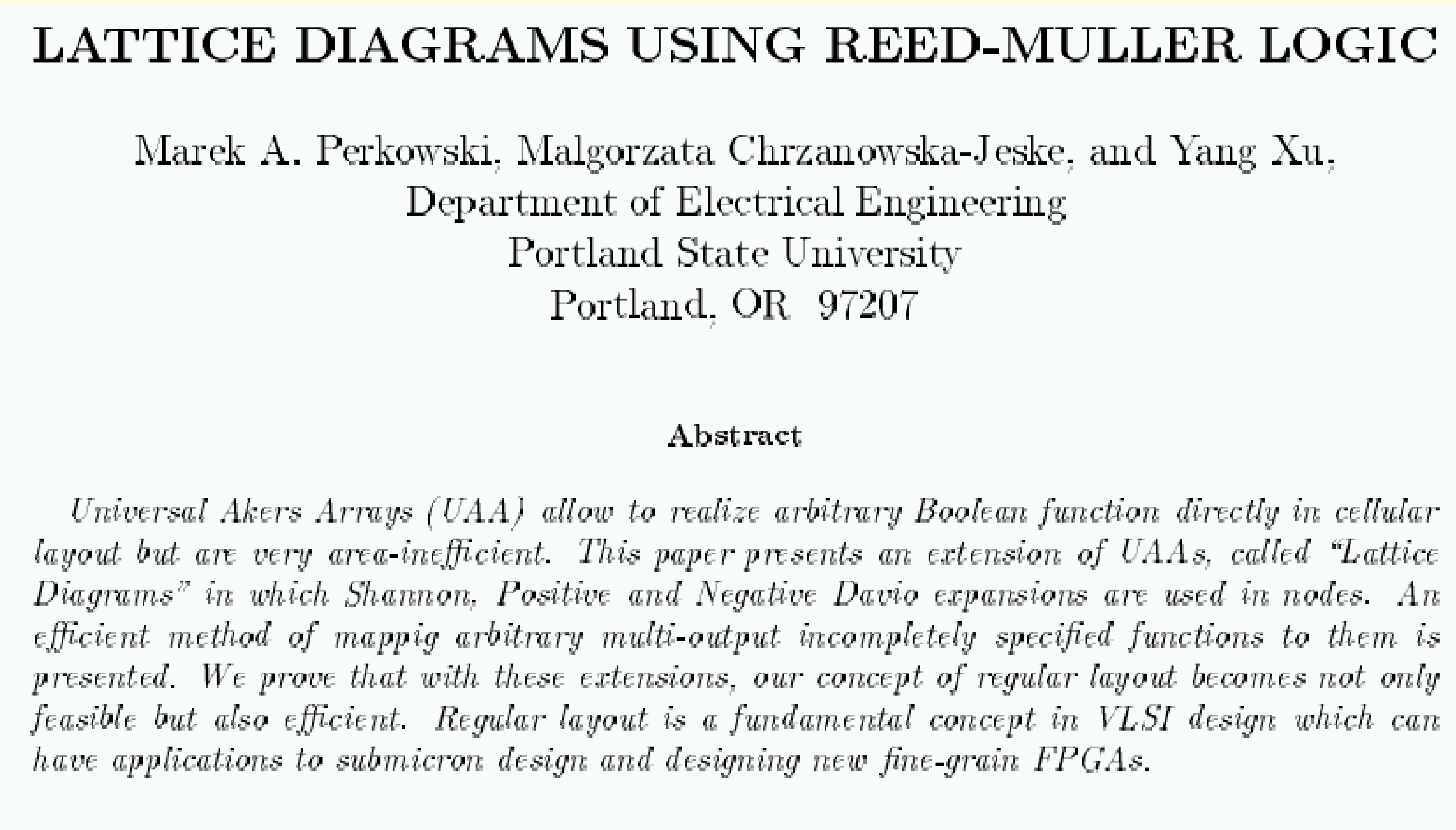
2-3 times repetition is enough



	Function		Heuristic I		Heuristic II		Heuristic III			from [4]		
Name	# of Inputs	# of SOP Products	# of Levels	# of Nodes	# of Levels	# of Nodes	# of Levels	# of Nodes	CPU Time	Best # of Nodes	# of Levels	# of Loops
5x10.esp*	7	3	7	7	7	7	7	7	0.8	11	7	2
5x7.esp*	7(3)	3	3	5	3	5	3	5	0.8	5	3	2
bw01.esp*	5	6	8	15	8	13	8	13	0.9	na	na	na
bw3.esp*	5	4	7	11	5	11	5	9	0.8	na	na	na
con1.tt*	7	10	10	15	8	15	8	13	0.9	15	7	1
con12.esp*	7(5)	5	8	17	5	10	5	10	1.0	14	7	2
exc2.tt*	7	14	7	7	7	10	7	8	0.9	8	6	1
f21.esp*	4	3	4	5	4	5	4	5	0.9	na	na	na
f54.esp*	8(5)	10	13	33	8	21	8	21	0.8	na	na	na
majority.esp*	5	5	5	7	5	10	5	9	0.9	8	5	1
misex50.esp*	6	6	9	21	13	43	13	43	0.9	27	11	2
misex53.esp*	6	6	11	22	8	15	8	14	0.9	23	11	2
misex60.esp*	12	2	12	13	12	13	12	13	0.9	14	12	1
misex61.esp*	12	2	12	15	12	23	12	15	0.9	21	12	1
misex64.esp*	10	4	X		19	50	19	45	0.9	33	13	2
z43.esp*	7(5)	12	7	21	7	21	7	21	0.9	na	na	na
z44.esp*	7(3)	4	3	6	3	6	3	6	0.9	6	3	1

Table 1: *Results for the version of the program with: (1) One Polarity, (2) Look Ahead. X means the process cannot stop. Heuristics I, II and III will be described in detail in a forthcoming paper. Last three columns has the results from [4] for comparison.*

To read more….



LATTICE DIAGRAMS USING REED-MULLER LOGIC

Marek A. Perkowski, Malgorzata Chrzanowska-Jeske, and Yang Xu,
Department of Electrical Engineering
Portland State University
Portland, OR 97207

Abstract

Universal Akers Arrays (UAA) allow to realize arbitrary Boolean function directly in cellular layout but are very area-inefficient. This paper presents an extension of UAAs, called "Lattice Diagrams" in which Shannon, Positive and Negative Davio expansions are used in nodes. An efficient method of mappig arbitrary multi-output incompletely specified functions to them is presented. We prove that with these extensions, our concept of regular layout becomes not only feasible but also efficient. Regular layout is a fundamental concept in VLSI design which can have applications to submicron design and designing new fine-grain FPGAs.

Homework

- For a simple 4-bit adder, design the following diagrams, and next their corresponding circuits:
  - KFDD, Kronecker Lattice
  - Positive Davio Lattice