УДК 519.714:004.3

СИНТЕЗ КВАНТОВЫХ СХЕМ НА ОСНОВЕ НЕ ПОЛНОСТЬЮ ОПРЕДЕЛЕННЫХ ФУНКЦИЙ И *if*-ДИАГРАММ РЕШЕНИЙ

А. А. ПРИХОЖИЙ[1)]

[1)]*Белорусский национальный технический университет, пр. Независимости, 65, 220013, г. Минск, Беларусь*

Рассматривается задача синтеза и оптимизации логических обратимых и квантовых схем по функциональным описаниям, представленным диаграммами решений. Задача относится к ключевым проблемам, решаемым в целях создания квантовых компьютеров и технологий квантовых вычислений. Предлагается новый метод последовательной трансформации исходной функциональной спецификации в квантовую схему, предусматривающий следующие состояния проекта: сокращенную упорядоченную диаграмму двоичных решений, *if*-диаграмму решений, функциональную *if*-диаграмму решений, обратимую схему, квантовую схему. Новизна метода состоит в расширении разложений Шеннона и Давио булевой функции по отдельной переменной до разложений этой же функции по другой булевой функции с получением продуктов разложения, представленных не полностью определенными функциями. Неопределенность в продуктах разложения расширяет возможности по минимизации графового представления заданной функции. Вместо двух исходящих ветвей вершины двоичной диаграммы генерируются три исходящие ветви вершины *if*-диаграммы, что увеличивает уровень параллелизма в обратимых и квантовых схемах. Для каждого шага трансформации предложены свои правила отображения, сокращающие число линий и вентилей и глубину схемы. Сравнение новых результатов с результатами, полученными известным методом отображения вершин двоичных диаграмм решений на каскады обратимых и квантовых вентилей, показало, что использование предложенного метода позволит существенно улучшить параметры синтезируемых квантовых схем.

Ключевые слова: обратимое вычисление; квантовая логическая схема; синтез; не полностью определенная функция; разложение функции; диаграмма решений; размер схемы; глубина схемы; минимизация.

SYNTHESIS OF QUANTUM CIRCUITS BASED ON INCOMPLETELY SPECIFIED FUNCTIONS AND *if*-DECISION DIAGRAMS

A. A. PRIHOZHY[a]

[a]*Belarusian National Technical University, 65 Niezaliežnasci Avenue, Minsk 220013, Belarus*

The problem of synthesis and optimisation of logical reversible and quantum circuits from functional descriptions represented as decision diagrams is considered. It is one of the key problems being solved with the aim of creating quantum computing technology and quantum computers. A new method of stepwise transformation of the initial functional

Образец цитирования:
Прихожий АА. Синтез квантовых схем на основе не полностью определенных функций и *if*-диаграмм решений. *Журнал Белорусского государственного университета. Математика. Информатика.* 2021;3:84–97 (на англ.).
https://doi.org/10.33581/2520-6508-2021-3-84-97

For citation:
Prihozhy AA. Synthesis of quantum circuits based on incompletely specified functions and *if*-decision diagrams. *Journal of the Belarusian State University. Mathematics and Informatics.* 2021;3:84–97.
https://doi.org/10.33581/2520-6508-2021-3-84-97

Автор:
Анатолий Алексеевич Прихожий – доктор технических наук, профессор; профессор кафедры программного обеспечения информационных систем и технологий факультета информационных технологий и робототехники.

Author:
Anatoly A. Prihozhy, doctor of science (engineering), full professor; professor at the department of information system and technology software, faculty of information technologies and robotics.
prihozhy@yahoo.com

specification to a quantum circuit is proposed, which provides for the following project states: reduced ordered binary decision diagram, *if*-decision diagram, functional *if*-decision diagram, reversible circuit and quantum circuit. The novelty of the method consists in extending the Shannon and Davio expansions of a Boolean function on a single variable to the expansions of the same Boolean function on another function with obtaining decomposition products that are represented by incompletely defined Boolean functions. Uncertainty in the decomposition products gives remarkable opportunities for minimising the graph representation of the specified function. Instead of two outgoing branches of the binary diagram vertex, three outgoing branches of the *if*-diagram vertex are generated, which increase the level of parallelism in reversible and quantum circuits. For each transformation step, appropriate mapping rules are proposed that reduce the number of lines, gates and the depth of the reversible and quantum circuit. The comparison of new results with the results given by the known method of mapping the vertices of binary decision diagram into cascades of reversible and quantum gates shows a significant improvement in the quality of quantum circuits that are synthesised by the proposed method.

Keywords: reversible computation; quantum logic circuit; synthesis; incompletely specified function; expansion of function; decision diagram; circuit size; circuit depth; minimisation.

Introduction

Nowadays, the synthesis of reversible and quantum logic circuits is an intensely investigated scientific direction [1–4]. The circuits have the same number of inputs and outputs, consist of reversible gates and implement permutation functions, therefore carry out information-lossless computations and have the dramatically reduced power consumption. The application domains of reversible computations are optical and DNA computing, nanotechnologies, cryptography and quantum computers [5]. The reversible and quantum design flow is mostly similar to the design automation flow of electronic circuits. It considers a design at abstraction levels from technology independent behavioural (high-level) descriptions to technology dependent quantum-level descriptions. Methods of synthesis of reversible and quantum logic circuits has been developed in [6–9]. Firstly, a reversible quantum compiler maps an input functional or algorithmic specification to a technology independent logic description that is composed of reversible gates from a reversible gate library. Its goal is the minimisation of the number of gates and the number of lines (qubits). Different intermediate representations and formats, including truth tables, binary decision diagrams (BDD), reduced ordered binary decision diagrams (ROBDD), functional decision diagrams (FDD), reduced ordered functional diagrams (ROFDD), Reed – Muller forms, etc., have been proposed in the literature [10–14]. At this level, exact and heuristic synthesis methods are used. The method based on Boolean satisfiability [15] gives exact solutions for only small circuit sizes. To synthesise larger functional specifications, heuristic methods as follows are proposed [6; 9; 10; 16]: transformation-based approach, evolutionary algorithms, decision diagram based methods, etc. Secondly, the obtained reversible logic circuit is mapped to a quantum circuit composed of technology dependent quantum gates from a quantum elementary gate library.

The approach proposed in [12] immediately maps a ROBDD to a reversible and then a quantum circuit. A new class of *if*-decision diagrams (IFD) and functional *if*-decision diagrams (FIFD) that is based on incompletely specified Boolean functions is proposed in [17–19]. The main contribution of this paper is as follows: 1) novel expansions of incompletely and completely specified Boolean functions, which use a minimisation operation, perform efficient transition from ROBDD to IFD, support the synthesis of three-branch-node diagrams with higher parallelism; 2) new step-wise method of transforming ROBDD to IFD and further transforming IFD to FIFD of reduced size and (or) depth; 3) efficient rules of mapping FIFD nodes to cascades of reversible gates and mapping FIFD to a reversible circuit; 4) new techniques for synthesis, minimisation and parallelisation of quantum circuits from the reversible circuits.

Reversible logic circuits

A reversible circuit [1; 3] realises a reversible logic function by means of lines and reversible gates. The function is bijective since it has equal number of inputs and outputs, maps each input to a unique output, and is capable of reconstructing the input from the output. The circuit has no fan-out and feedback connections. The lines are labeled by Boolean variables of a set X. The reversible gate has the form of $G(T, C)$, where $T \subset X$ is a set of target lines, and $C \subset X$ is a set of control lines ($C \cap T = \varnothing$). The gate operation is applied to the target line if the control lines meet a true condition. The target line is represented by $\oplus$, and its control line is represented by $\bullet$. Nowadays, some reversible gate libraries are available. The NCT library [1] includes (fig. 1, *a*) the NOT gate, the CNOT gate with one control line, and the Toffoli gate with two control lines. The GT library [6] includes generalised multiple-control Toffoli gates. The Toffoli gate maps three inputs to three outputs: $G(L, C_0, C_1) = (L \oplus (C_0 \wedge C_1), C_0, C_1)$, where $\oplus$ is exclusive-or; $\wedge$ is Boolean conjunction;

L is input at target line; C_0, C_1 are conditions at control lines. The CNOT gate maps two inputs to two outputs: $G(L,\, C_0) = (L \oplus C_0,\, C_0)$. The NOT gate realises Boolean inversion: $G(L) = (\neg L)$. Therefore, the reversible circuit describes a behaviour with a superposition of three Boolean operations: $\neg$, $\wedge$ and $\oplus$.

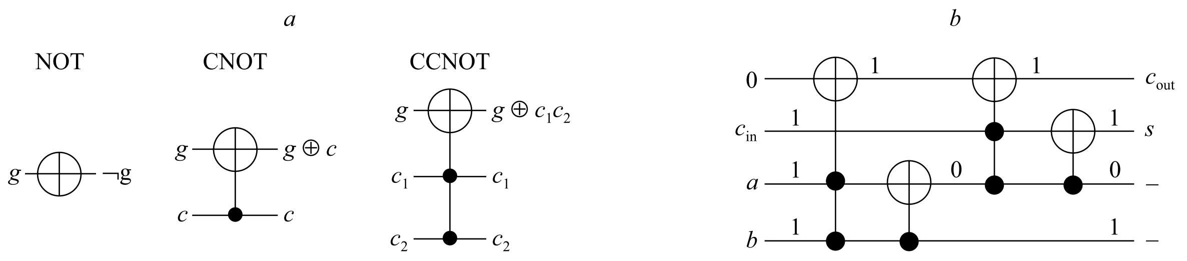


Fig. 1. Reversible gates of NCT library:
a – reversible gates NOT, CNOT and CCNOT (Toffoli);
b – realisation of 1-bit full adder by cascade of gates

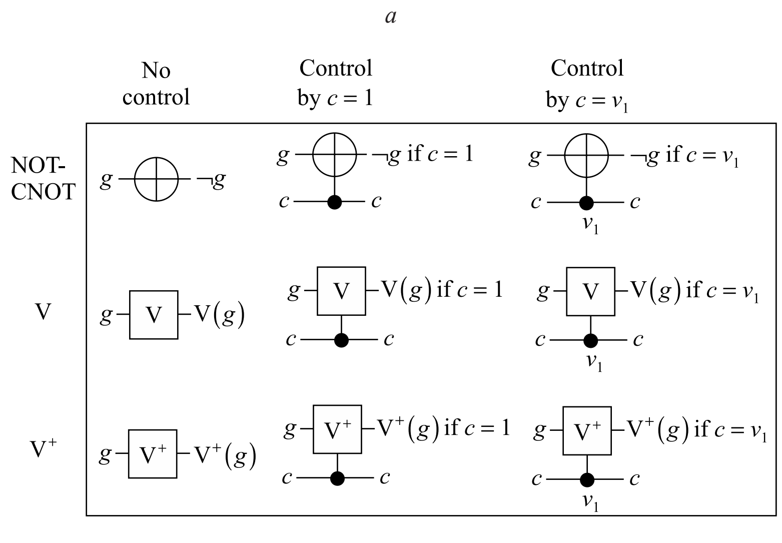
Quantum logic circuits

Quantum circuits carry out computations by changing states of qubits [1; 3]. The qubit state is $|\Psi\rangle = \alpha|0\rangle +$ $+\, \beta|1\rangle$, where α and β are complex numbers such that $|\alpha|^2 + |\beta|^2 = 1$. Quantum gates perform operations on qubits. A cascade of quantum gates applied to qubits is a quantum circuit. Quantum logic gates and circuits are reversible. Competitive quantum gate libraries are developed. The NCV library [1] is the most commonly used for synthesis of quantum circuits. Its gates require Boolean control lines. To the NOT and CNOT gates, the NCV adds V, CV, V^+ and CV^+ gates, which are known as square-root of the NOT gate:

$$\mathrm{V} = \frac{1+i}{2}\begin{pmatrix} 1 & -i \\ -i & 1 \end{pmatrix},\ \mathrm{V}^+ = \frac{1-i}{2}\begin{pmatrix} 1 & i \\ i & 1 \end{pmatrix}.$$

The NCV-$|v_1\rangle$ [3] library considers a 4-level (0, v_0, 1 and v_1) quantum system and uses qudits instead of qubits. Its gates use v_1 as control value. Figure 2, *a*, depicts the key NCV and NCV-$|v_1\rangle$ quantum gates. The NOT gate without control, and CNOT gate controlled by 1 are similar to the corresponding gates of NCT. NCV-$|v_1\rangle$ introduces a CNOT gate controlled by v_1. The gate keeps the value unchanged if $c \neq v_1$. There are V and V^+ gates without control and with control by v_1. Figure 2, *b*, describes the gates' mapping functions.

Since the quantum libraries have no gate that immediately implements the Toffoli gate, they replace the gate by a cascade of quantum gates from the NCV (fig. 3, *a*) or NCV-$|v_1\rangle$ (fig. 3, *b*) library. A n-control Toffoli gate can be replaced by $2n + 1$ quantum gates from NCV-$|v_1\rangle$. After the replacement, minimisation techniques can reduce the quantum circuit size. Figure 3, *c*, depicts a minimised quantum circuit of 1-bit full adder.



b

g	NOT(g)	V(g)	V$^+$(g)
0	1	v_0	v_1
v_0	v_1	1	0
1	0	v_1	v_0
v_1	v_0	0	1

Fig. 2. Quantum gates of NCV and NCV-$|v_1\rangle$ libraries:
a – quantum gates NOT, CNOT, V and V$^+$; *b* – operation of quantum gates in 4-level quantum system

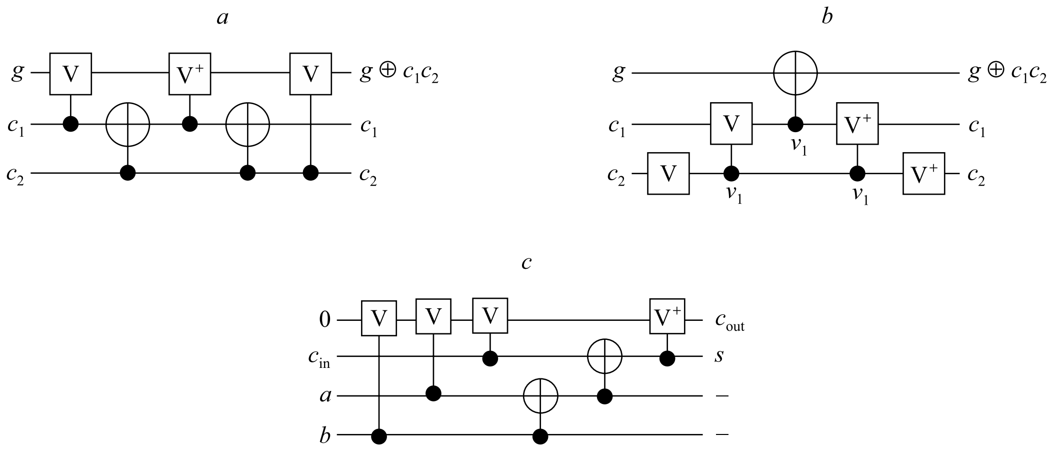


Fig. 3. Realisation of Toffoli two-control gate by cascades of quantum gates:
a – cascade of NCV library gates; *b* – cascade of NCV-$|v_1\rangle$ library gates;
c – quantum circuit for 1-bit full adder

Approaches for modelling, synthesis and optimisation of reversible and quantum circuits

Various models and methods are used for synthesis and optimisation of reversible and quantum logic circuits.

Reed – Muller expansions (Zhegalkin polynomials). The forms are based on two-level AND/EXOR and directly mapped into reversible circuits using the multi-control Toffoli gates.

Optimisation based on exclusive-sum-of-products (ESOP). Described in the literature [11] approaches for minimising the exclusive-or of Boolean cubes are applicable to reversible and quantum circuits.

Quantum operator form. It is a quantum extension [14] to the Reed – Muller form, which represents quantum circuits based on the CNOT, CV and CV$^+$ quantum gates, and permits minimisation of circuits by using properties of quantum gates in addition to Toffoli gates.

Binary decision diagram. It is a graph representation of a Boolean function [12; 13] based on the Shannon expansion (1) of Boolean function $f(x)$, $x = (x_1, \ldots, x_n)$ on variable x_i:

$$f = \neg x_i \wedge f_{x_i=0} \vee x_i \wedge f_{x_i=1}, \tag{1}$$

where $\vee$ is Boolean disjunction. Residual functions $f_{x_i=0}$ and $f_{x_i=1}$ are called negative and positive cofactors. Syntactically, BDD is a rooted acyclic directed graph consisting of terminal nodes labeled by 0 and 1 and non-terminal nodes (fig. 4, *a*) labeled by Boolean variables $x_i \in X$ and having outgoing edges *low* and *high*. Term $bdd(x_i, n, p)$ denotes a non-terminal node. In a reduced ordered ROBDD, the variable order is the same along all paths from root to leaves. Rules *S* and *I* perform the reduction: rule *S* deletes non-terminal node $v = bdd(x_i, n, n)$ and redirects the incoming edges from *v* to *n*; rule *I* removes one of two identical nodes and redirects its incoming edges to the remaining node. Figure 4, *b*, depicts an example ROBDD.

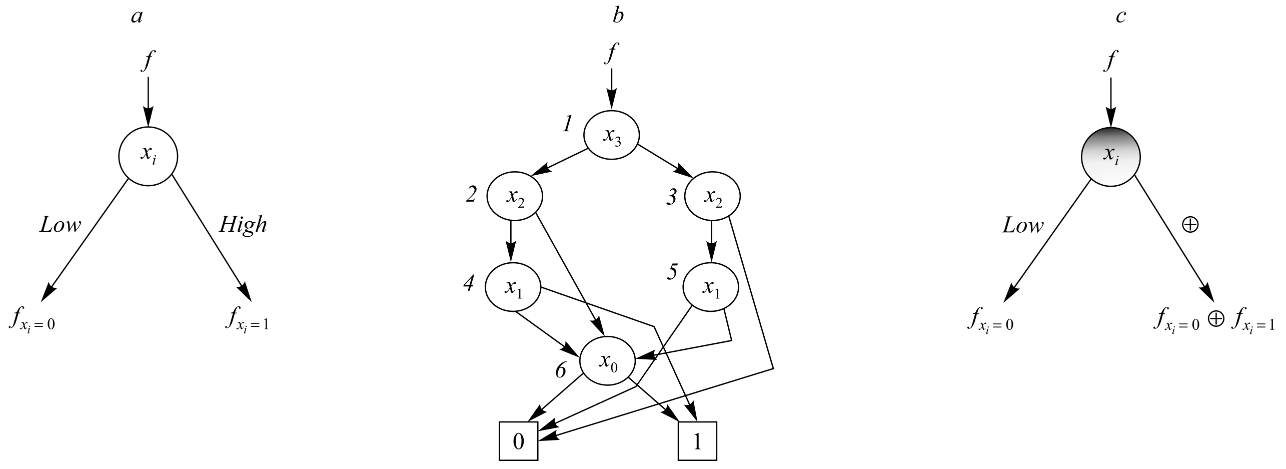


Fig. 4. Binary and functional decision diagrams: *a* – non-terminal node of BDD;
b – example ROBDD (*1–6* denote the diagram node numbers); *c* – non-terminal node of FDD

Functional decision diagrams. The positive and negative Davio expansions (2) and (3) of Boolean function $f(x)$ was a basic idea for creating the FDD:

$$f = f_{x_i=0} \oplus x_i \wedge \left(f_{x_i=0} \oplus f_{x_i=1}\right), \tag{2}$$

$$f = f_{x_i=1} \oplus \neg x_i \wedge \left(f_{x_i=0} \oplus f_{x_i=1}\right). \tag{3}$$

Constructively FDD is similar to BDD. The non-terminal node (fig. 4, *c*) is labeled by $x_i \in X$, and has two outgoing edges *low* and ⊕, which point two sub-diagrams representing cofactors $n = f_{x_i=0}$ and $b = f_{x_i=0} \oplus f_{x_i=1}$ respectively for (2). Term $fdd(x_i, n, b)$ denotes the FDD. A ROFDD orders the variables and has exactly two terminal nodes labeled by 0 and 1. Two rules *D* and *I* reduce the diagram: rule *D* deletes non-terminal node $v = fdd(x_i, n, 0)$ and redirects the incoming edges from *v* to *n*; rule *I* is the same as for ROBDD. Figure 5, *a*, depicts an example ROFDD derived from the example ROBDD. The ROFDD's nodes match well reversible gates.

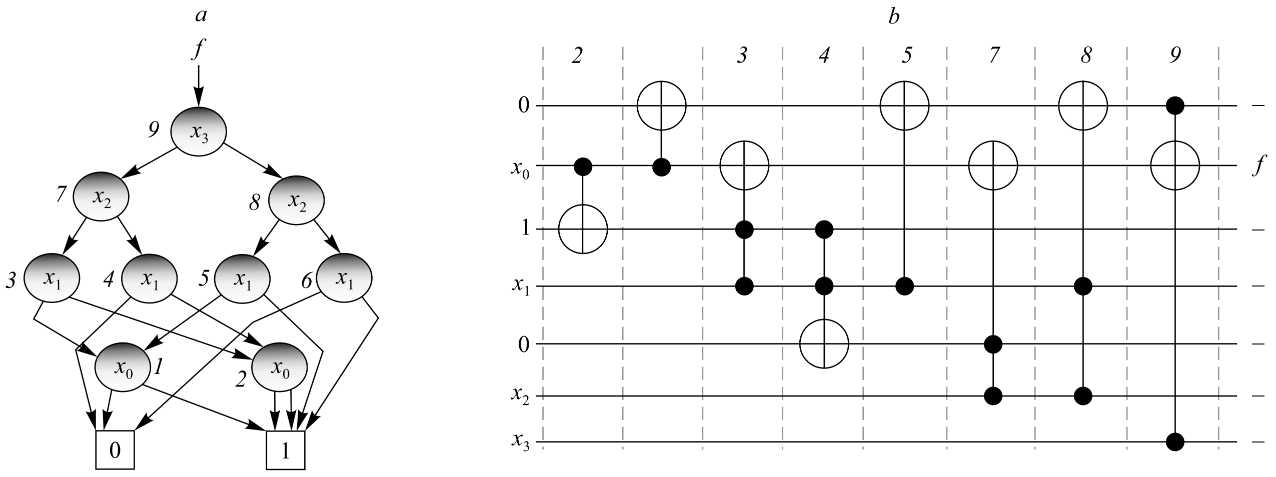


Fig. 5. Mapping BDD to reversible circuit:
a – example ROFDD that is functionally equivalent
to example ROBDD (*1–9* denote the diagram node numbers);
b – realisation of ROFDD by reversible circuit (diagram node numbers are above columns)

Works [11; 12] propose a technique of synthesising a reversible circuit from ROBDD, which substitutes all diagram nodes with cascades of reversible gates. The size of reversible circuit directly depends on the ROBDD size. Since the ROBDD size can grow exponentially over the number of input variables, the reversible circuit size can grow exponentially too. An alternative technique first transforms ROBDD to ROFDD, and then maps the ROFDD to a reversible circuit. Figure 5, *b*, depicts a reversible circuit synthesised from the example ROFDD.

Quantum gates controlled by various quantum levels. The gates controlled by value v_1 that are introduced in the NCV-$|v_1\rangle$ library reduce the overall size of cascades that realise the multiple-control Toffoli gates [3; 6].

Template matching. It is a common approach to reversible and quantum circuit simplification [6]. Each template that is a cascade of gates in a circuit is replaced by a functionally equivalent smaller cascade.

Using additional ancillaries. Sometimes extra ancillary lines leads to reducing the quantum circuit size and depth [16].

Parallelisation of gates. In a reversible or quantum circuit, two adjacent gates $G(T1, C1)$ and $G(T2, C2)$ can be parallelised [16] and decrease the circuit depth if $(T1 \cup C1) \cap (T2 \cup C2) = \varnothing$.

Expansions of incompletely specified functions

Let $B = \{0, 1\}$ be the set of Boolean values. Set $M = \{0, 1, dc\}$ extends *B* with a don't care value *dc*. A partial variable y_i assumes values from *M*. A partial function $\Phi(y)$ (also known as Kleene function) of variable $y = (y_1, \dots, y_n)$ is a mapping Φ: $M^n \to M$. From functions $\Phi(y)$ and $\Psi(y)$, a superposition $\Phi(y_1, \dots, y_{i-1}, \Psi(y), y_{i+1}, \dots, y_n)$ can be created describing a new partial function. Key unary and binary partial operations are analogues of Boolean unary and binary operations. A partial negation $\sim y_i$ (analogue of Boolean negation $\neg$) produces 1, 0 and *dc* if y_i is 0, 1 and *dc* respectively. Table 1 defines key partial binary operations: conjunction (&), disjunction (+), implication ($\Rightarrow$) and exclusive disjunction ($\otimes$). These are analogues of the Boolean operations $\wedge$, $\vee$, $\rightarrow$ and $\oplus$. For partial operations ~, & and +, the following laws hold: associativity, commutativity, distributivity, identity, annihilator, idempotence, double negation, De Morgan's laws and others.

Table 1

Binary partial logic operations

Variable y_1	0	0	0	1	1	1	*dc*	*dc*	*dc*	Notation
Variable y_2	0	1	*dc*	0	1	*dc*	0	1	*dc*	
Conjunction	0	0	0	0	1	*dc*	0	*dc*	*dc*	$y_1 \mathbin{\&} y_2$
Disjunction	0	1	*dc*	1	1	1	*dc*	1	*dc*	$y_1 + y_2$
Implication	1	1	1	0	1	*dc*	*dc*	1	*dc*	$y_1 \Rightarrow y_2$
Exclusive disjunction	0	1	*dc*	1	0	*dc*	*dc*	*dc*	*dc*	$y_1 \otimes y_2$

An incompletely specified function $\varphi(x)$ of vector Boolean variable $x = (x_1, \ldots, x_n)$ is a mapping φ: $B^n \rightarrow M$. Three sets represent $\varphi(x)$: on-set ON^{φ}; off-set OFF^{φ} and don't care set DC^{φ}. Three Boolean characteristic functions describe the sets: $\varphi^{\mathrm{on}}(x)$, $\varphi^{\mathrm{off}}(x)$ and $\varphi^{\mathrm{dc}}(x)$. Function $\varepsilon(x)$ is an extension of $\varphi(x)$ if $\mathrm{ON}^{\varepsilon} \supseteq \mathrm{ON}^{\varphi}$ and $\mathrm{OFF}^{\varepsilon} \supseteq \mathrm{OFF}^{\varphi}$.

Definition 1. A pair $(f(x)|d(x))$ of Boolean functions is called a value/domain representation of an incompletely specified function $\varphi(x)$ if $f^{\mathrm{on}}(x) \subseteq \varphi^{\mathrm{on}}(x) \cup \varphi^{\mathrm{dc}}(x)$ denotes a value Boolean function and $d(x) = \neg\varphi^{\mathrm{dc}}(x)$ denotes a certainty domain Boolean function (or simply domain function). The pair specifies that $\varphi(x) = 1$ in the area described by Boolean characteristic function $\varphi^{\mathrm{on}}(x) = f(x) \wedge d(x)$, $\varphi(x) = 0$ in the area described by Boolean characteristic function $\varphi^{\mathrm{off}}(x) = \neg f(x) \wedge d(x)$, and $\varphi(x) = dc$ in the area described by Boolean characteristic function $\varphi^{\mathrm{dc}}(x) = \neg d(x)$.

Function $\varphi(x)$ specifies constant 0, constant 1 and constant *dc* by terms $(0|1)$, $(1|1)$ and $(v|0)$ respectively where *v* is an arbitrary Boolean function. Since the partial logic operations can be applied to pairs $(f(x)|d(x))$, it becomes possible to construct expressions for describing various incompletely specified functions. It also appears to be possible to reduce such expressions to a pair of Boolean functions that are described by Boolean expressions.

Theorem 1. *Following equalities hold for any incompletely specified functions* $(v_1|d_1)$ *and* $(v_2|d_2)$.

$$\sim(v_1|d_1) = (\neg v_1|d_1), \tag{4}$$

$$(v_1|d_1) \mathbin{\&} (v_2|d_2) = (v_1 \wedge v_2 | d_1 \wedge d_2 \vee \neg v_1 \wedge d_1 \vee \neg v_2 \wedge d_2), \tag{5}$$

$$(v_1|d_1) + (v_2|d_2) = (v_1 \vee v_2 | d_1 \wedge d_2 \vee v_1 \wedge d_1 \vee v_2 \wedge d_2), \tag{6}$$

$$(v_1|d_1) \Rightarrow (v_2|d_2) = (v_1 \rightarrow v_2 | d_1 \wedge d_2 \vee \neg v_1 \wedge d_1 \vee v_2 \wedge d_2), \tag{7}$$

$$(v_1|d_1) \otimes (v_2|d_2) = (v_1 \oplus v_2 | d_1 \wedge d_2). \tag{8}$$

P r o o f. The equalities are proved in work [18, p. 65–71].

An advantage of equalities (4)–(8) is that the value function of each right part is a Boolean operation that corresponds to the partial operation of the left part. The domain function of the right part is a Boolean function of four essential variables v_1, d_1, v_2, d_2. Equalities (4)–(8) allow the development of expansions of incompletely specified functions on partial logic operations. Let construct expansions on operations &, + and ~.

Theorem 2. *The following equality holds for any incompletely specified function* $(f|d)$ *and any Boolean function c.*

$$(f|d) = (c|1) \mathbin{\&} (f|d \wedge c) + \sim(c|1) \mathbin{\&} (f|d \wedge \neg c). \tag{9}$$

P r o o f. Using (4)–(6), the right part of (9) is equivalently transformed to its left part:

$$(c|1) \mathbin{\&} (f|d \wedge c) + \sim(c|1) \mathbin{\&} (f|d \wedge \neg c) =$$
$$= (c \wedge f | d \wedge c \vee \neg c \wedge 1 \vee \neg f \wedge d \wedge c) + (\neg c \wedge f | d \wedge \neg c \vee c \wedge 1 \vee \neg f \wedge d \wedge \neg c) =$$

$$=(c\wedge f\,|\,d\vee\neg c)+(\neg c\wedge f\,|\,d\vee c)=$$
$$=(c\wedge f\vee\neg c\wedge f\,|\,(d\vee\neg c)\wedge(d\vee c)\vee c\wedge f\wedge(d\vee\neg c)\vee\neg c\wedge f\wedge(d\vee c))=$$
$$=(f\,|\,d\vee f\wedge d\wedge c\vee f\wedge d\wedge\neg c)=(f\,|\,d).$$

The theorem is proved.

Expansion (9) generalises the well-known Shannon expansion for incompletely specified functions. It replaces a Boolean variable with an arbitrary Boolean function, and replaces cofactors on one Boolean variable with cofactors that are incompletely specified functions.

Corollary 1. *If function d is constant* 1 *and* $(f\,|\,1)$ *is a completely specified function, (9) is reduced to (10):*

$$(f\,|\,1)=(c\,|\,1)\,\&\,(f\,|\,c)+\sim(c\,|\,1)\,\&\,(f\,|\,\neg c).\qquad(10)$$

Expansion (10) of the completely specified function introduces terms that describe incompletely specified functions with the same Boolean value function f and domain functions c and $\neg c$ reducing the area of certainty. It is a strength of the expansion. Note that (8) does not support the development of similar expansions for exclusive disjunction.

The following theorem and its corollaries allow to obtain novel expansions of an incompletely specified function whose domain function is represented by a complex Boolean expression. The expression is constructed of such operations as Boolean negation, conjunction, disjunction, implication and exclusive disjunction. The key expansion decomposes an incompletely specified function with the if-then-else Boolean function in the domain part.

Theorem 3. *The following equality holds for any Boolean functions f, e, g and h:*

$$(f\,|\,e\wedge g\vee\neg e\wedge h)=(e\,|\,1)\,\&\,(f\,|\,e\wedge g)+\sim(e\,|\,1)\,\&\,(f\,|\,\neg e\wedge h).\qquad(11)$$

P r o o f. The right part of (11) is equivalently transformed to its left part using (4)–(6):

$$(e\,|\,1)\,\&\,(f\,|\,e\wedge g)+\sim(e\,|\,1)\,\&\,(f\,|\,\neg e\wedge h)=$$
$$=(e\wedge f\,|\,e\wedge g\vee\neg e\wedge 1\vee e\wedge\neg f\wedge g)+(\neg e\wedge f\,|\,\neg e\wedge h\vee e\wedge 1\vee\neg e\wedge\neg f\wedge h)=$$
$$=(e\wedge f\,|\,\neg e\vee g)+(\neg e\wedge f\,|\,e\vee h)=$$
$$=(e\wedge f\vee\neg e\wedge f\,|\,(\neg e\vee g)\wedge(e\vee h)\vee e\wedge f\wedge(\neg e\vee g)\vee\neg e\wedge f\wedge(e\vee h))=$$
$$=(f\,|\,g\wedge h\vee e\wedge g\vee\neg e\wedge h\vee e\wedge f\wedge g\vee\neg e\wedge f\wedge h)=(f\,|\,e\wedge g\vee\neg e\wedge h).$$

The theorem is proved.

Corollary 2. *If function g is Boolean constant* 1 *and the domain function is Boolean disjunction* $e\vee h$ *then (11) is reduced to (12):*

$$(f\,|\,e\vee h)=(e\,|\,1)\,\&\,(f\,|\,e)+\sim(e\,|\,1)\,\&\,(f\,|\,\neg e\wedge h).\qquad(12)$$

Corollary 3. *If function h is Boolean constant* 1 *and the domain function is Boolean implication* $e\rightarrow g=$ $=\neg e\vee g$ *then (11) is reduced to (13):*

$$(f\,|\,e\rightarrow g)=(e\,|\,1)\,\&\,(f\,|\,e\wedge g)+\sim(e\,|\,1)\,\&\,(f\,|\,\neg e).\qquad(13)$$

Corollary 4. *If function* $g=\neg h$ *and the domain function is Boolean exclusive disjunction* $e\oplus h=$ $=e\wedge\neg h\vee\neg e\wedge h$ *then (11) is reduced to (14):*

$$(f\,|\,e\oplus h)=(e\,|\,1)\,\&\,(f\,|\,e\wedge\neg h)+\sim(e\,|\,1)\,\&\,(f\,|\,\neg e\wedge h).\qquad(14)$$

Corollary 5. *Incompletely specified function* $(f\,|\,c)$, $c=e\wedge g\vee\neg e\wedge h$ *that is described by (11) can be considered as a positive cofactor of completely specified function* $(f\,|\,1)$ *in (10). Since* $\neg c=e\wedge\neg g\vee\neg e\wedge\neg h$, *a negative cofactor* $(f\,|\,\neg c)$ *is inferred from (11) by substituting* $\neg g$ *instead of g and substituting* $\neg h$ *instead of h. As a result, equality (10) can be transformed to (15):*

$$(f\,|\,1)=(c\,|\,1)\,\&\,\big[(e\,|\,1)\,\&\,(f\,|\,e\wedge g)+\sim(e\,|\,1)\,\&\,(f\,|\,\neg e\wedge h)\big]+$$
$$+\sim(c\,|\,1)\,\&\,\big[(e\,|\,1)\,\&\,(f\,|\,e\wedge\neg g)+\sim(e\,|\,1)\,\&\,(f\,|\,\neg e\wedge\neg h)\big].\qquad(15)$$

Equalities (9)–(15) have a wonderful property: all domain functions in right part terms are conjunctions of positive and negative literals. The property simplifies the construction of cofactors of completely and incompletely specified functions.

Expansions based on minimisation of incompletely specified functions

In an incompletely specified function $\varphi = \left(f \,|\, c\right)$, Boolean function *f* may be replaced with any extension *v* from slice (16) without changing φ:

$$\left(f \wedge c\right)^{\text{on}} \subseteq v^{\text{on}} \subseteq \left(f \vee c\right)^{\text{on}}. \tag{16}$$

Since the functions of slice (16) may obtain different features, in particular they can produce quantum circuits of smaller time delay and (or) occupied area, the author of works [17; 18] introduced an operation $v = \min\left(f \,|\, c\right)$ to select a best function of the slice. In (9)–(15), all incompletely specified functions represented by a pair $\left(f \,|\, c\right)$ of Boolean functions may be replaced with the Boolean function $\min\left(f \,|\, c\right)$, and $\left(f \,|\, 1\right)$ may be replaced with *f* in case all partial logical operations are replaced by corresponding Boolean operations. It should be noted, in a single expression, several *min* operations can be considered as mutually dependent. It can improve the result of incompletely specified function minimisation.

Thus, expansion (10) is transformed to the following expansion:

$$f = c \wedge \min\left(f \,|\, c\right) \vee \neg c \wedge \min\left(f \,|\, \neg c\right). \tag{17}$$

Expansion (17) generalises the Shannon expansion (2) by replacing a Boolean variable x_i with a Boolean function *c*, and replacing the cofactors on x_i with the cofactors on *c* and *min*. Expansion (15) with $c = e \wedge g \vee \neg e \wedge h$ from corollary 5 can be rewritten as follows:

$$\begin{aligned} f = c \wedge \left[e \wedge \min\left(f \,|\, e \wedge g\right) \vee \neg e \wedge \min\left(f \,|\, \neg e \wedge h\right)\right] \vee \\ \vee \neg c \wedge \left[e \wedge \min\left(f \,|\, e \wedge \neg g\right) \vee \neg e \wedge \min\left(f \,|\, \neg e \wedge \neg h\right)\right]. \end{aligned} \tag{18}$$

Expansion (18) has no analogue in the literature. Its special cases are expansion (19) for $c = e \wedge g$:

$$f = c \wedge \left[\min\left(f \,|\, e \wedge g\right)\right] \vee \neg c \wedge \left[e \wedge \min\left(f \,|\, e \wedge \neg g\right) \vee \neg e \wedge \min\left(f \,|\, \neg e\right)\right] \tag{19}$$

and expansion (20) for $c = e \oplus h$:

$$\begin{aligned} f = c \wedge \left[e \wedge \min\left(f \,|\, e \wedge \neg h\right) \vee \neg e \wedge \min\left(f \,|\, \neg e \wedge h\right)\right] \vee \\ \vee \neg c \wedge \left[e \wedge \min\left(f \,|\, e \wedge h\right) \vee \neg e \wedge \min\left(f \,|\, \neg e \wedge \neg h\right)\right]. \end{aligned} \tag{20}$$

Equalities (17)–(20) not only select a particular value function from slice (16) for each *min* operation, they rather prove to be held for any value function from slice (16) that corresponds to every incompletely specified term. The positive and negative Davio expansions are also generalised by means of using incompletely specified functions and the *min* operation:

$$f = \min\left(f \,|\, \neg c\right) \oplus c \wedge \left(\min\left(f \,|\, \neg c\right) \oplus \min\left(f \,|\, c\right)\right), \tag{21}$$

$$f = \min\left(f \,|\, c\right) \oplus \neg c \wedge \left(\min\left(f \,|\, \neg c\right) \oplus \min\left(f \,|\, c\right)\right). \tag{22}$$

In expansions (17)–(22), the value function of all incompletely specified terms is *f*. Contrary, the domain functions are different as well as the level of uncertainty they describe.

Most important realisations of the *min* operation are those reducing the number of essential variables in cofactors. For this reason, we consider the replacement of functions *e*, *g* and *h* with Boolean variables: $e = x_i$, $g = x_j$ and $h = x_k$. Moreover, we introduce new cofactors: $f_{x_i = 0,\, x_j = 0}$, $f_{x_i = 0,\, x_k = 1}$, $f_{x_i = 1,\, x_k = 0}$, $f_{x_i = 1,\, x_j = 1}$, $f_{x_k = x_i}$ and $f_{x_k = \neg x_i}$ of function *f*. The first four cofactors are constructed on Boolean conjunction and are given by residual functions of two variables less. Cofactors $f_{x_k = x_i}$ and $f_{x_k = \neg x_i}$ being introduced by the authors of works [18; 20] replace variable x_k with variable x_i and its negation $\neg x_i$ respectively, and reduce by one the number of essential variables in function *f*. If $c = x_i \wedge x_j \vee \neg x_i \wedge x_k$, expansion (18) is transformed to expansion (23):

$$f = c \wedge \left(x_i \wedge f_{x_i = 1,\, x_j = 1} \vee \neg x_i \wedge f_{x_i = 0,\, x_k = 1}\right) \vee \neg c \wedge \left(x_i \wedge f_{x_i = 1,\, x_j = 0} \vee \neg x_i \wedge f_{x_i = 0,\, x_k = 0}\right). \tag{23}$$

If $c = x_i \wedge x_j$, expansion (19) is transformed to expansion (24):

$$f = c \wedge f_{x_i = 1,\, x_j = 1} \vee \neg c \wedge \left(x_i \wedge f_{x_i = 1,\, x_j = 0} \vee \neg x_i \wedge f_{x_i = 0}\right). \tag{24}$$

If $c = x_i \oplus x_k$, expansion (20) is transformed to expansion (25):

$$f = c \wedge f_{x_k = \neg x_i} \vee \neg c \wedge f_{x_k = x_i}. \tag{25}$$

Expansions (17)–(25) are capable of efficiently solving the problems of reversible and quantum circuit modelling, synthesis and optimisation.

If-decision diagrams

The author of works [17–19] proposed the concept of IFD and FIFD that are derived from expansions of incompletely specified functions. IFD is a rooted directed acyclic labeled graph consisting of non-terminal and terminal nodes. Semantically a non-terminal node of the graph is interpreted as a representation of Boolean function using (17). Figure 6, *a*, depicts a non-terminal node of IFD, which is not labeled and has three outgoing edges *if*, *high* and *low*. Edge *if* points a sub-graph representing function *c*. Edge *high* points a sub-graph representing function $g = \min(f \mid c)$. Edge *low* points a sub-graph representing function $h = \min(f \mid \neg c)$. A term $ifd(c, g, h)$ denotes the node. A labeled terminal node represents a Boolean constant 0, constant 1, variable x_i or its negation $\neg x_i$. IFD can be with or without complement edges. Many reduction rules can be applied to IFD. The *S* and *I* rules of reducing ROBDD are also applicable to IFD. The rules as follows being inapplicable to BDD are new for reducing IFD: $ifd(c, c, h) = ifd(c, 1, h)$, $ifd(c, \neg c, h) = ifd(c, 0, h)$, $ifd(c, g, c) = ifd(c, g, 0)$, $ifd(c, g, \neg c) = ifd(c, g, 1)$ and others. A biconditional binary decision diagram that is proposed in works [18; 20] is a special case of IFD at $c = x_i \oplus x_{i+1}$ and is efficiently applicable to the synthesis of reversible and quantum circuits.

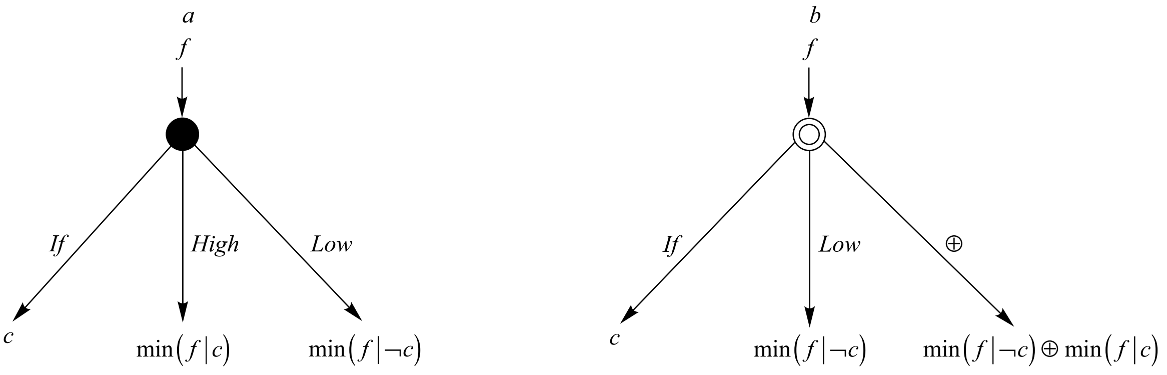


Fig. 6. Non-terminal node of IFD (*a*) and positive FIFD (*b*)

Figure 6, *b*, depicts a non-terminal node of positive FIFD constructed using expansion (21). The node has three outgoing edges *if*, *low* and ⊕, which point sub-graphs representing functions *c*, $\min(f \mid \neg c)$ and $\min(f \mid \neg c) \oplus \min(f \mid c)$ respectively. Slightly different negative FIFD is constructed using expansion (22). The non-terminal nodes of IFD and FIFD have different views. IFD and FIFD are a promising generalisation of BDD and FDD. All BDDs and FDDs including ROBDDs and ROFDDs can be directly modelled by IFDs and FIFDs. At the same time, tremendous number of IFD and FIFD configurations exist that are not modelled by BDD and FDD. The configurations may allow producing quantum circuit solutions, which overcome those provided by BDD and FDD regarding time and cost parameters. IFD and FIFD have three outgoing edges instead of two in BDD and FBDD, therefore their capabilities for parallelisation are much larger [19].

Synthesis of *if*-decision diagrams from binary decision diagrams

Starting from BDD or ROBDD, we ask the question, what could be a method of generating a functionally equivalent IFD, which has better parameters? To synthesise an IFD of a smaller size and (or) depth, we have developed a method, which intensively exploits the expansions of logic functions proposed in previous sections. Key points of the method are as follows:

1) the transition from ROBDD to IFD which reduces the diagram size and (or) depth is accomplished by the step-wise replacement of nodes with two outgoing edges by nodes with three outgoing edges; the transition is carried out by applying the proposed expansions of incompletely and completely specified functions, minimisation operation and reduction rules;

2) the construction of a new non-terminal *ifd*-node is based on the selection of two or three variables x_i, x_j and x_k, and a Boolean function c from the set $\{x_i \wedge x_j \vee \neg x_i \wedge x_k, x_i \wedge x_j, x_i \oplus x_k, \ldots\}$ including numerous modifications of the set's functions obtained by replacing positive literals with negative literals in various combinations;

3) the selection of a preferable expansion and one or more sub-diagrams is carried out for the chosen c function. The selection depends on the expansion properties and sub-diagram features associated with their ability of further diagram reduction;

4) the introduction of new *ifd*-nodes and applying the selected expansion to selected sub-diagrams give a new intermediate IFD;

5) the application of reduction rules to the current IFD gives a next-step IFD of decreased size and depth;

6) the diagram step-wise transformation is over if no expansion and sub-diagram have been found improving the IFD parameters.

Let demonstrate how the technique works on the example ROBDD depicted in fig. 4, *b*, which is immediately transformed to a non-reduced initial IFD depicted in fig. 7, *a*.

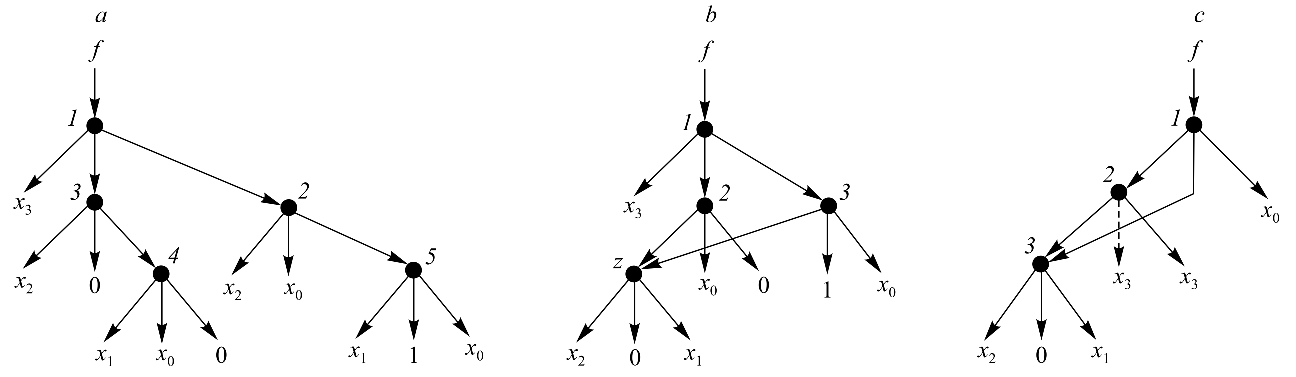


Fig. 7. Transforming example ROBDD to IFD:
a – initial IFD (*1–5* denote the diagram node numbers);
b – intermediate IFD (*1–3* denote the diagram node numbers; *z* is a function associated with the node);
c – minimised IFD (*1–3* denote the diagram node numbers; dash line denotes complement edge)

First observe that the sub-diagram consisting of non-terminal nodes *2* and *4* (node numbering in fig. 4, *b*) can be represented by a term $t_2 = ifd\big(x_2, ifd(x_1, t_6, t_6), ifd(x_1, 1, t_6)\big)$, where t_2 and t_6 are sub-diagrams on nodes *2* and *6* respectively. Let construct function $c = \neg x_2 \wedge x_1$ that selects constant 1 in the term and re-writes (24) to (26):

$$f = c \wedge f_{x_2=0,\, x_1=1} \vee \neg c \wedge \left(\neg x_2 \wedge f_{x_2=0,\, x_1=0} \vee x_2 \wedge f_{x_2=1}\right). \tag{26}$$

Expansion (26) allows grouping identical nodes in such a way that the IFD is reduced significantly. Its application to t_2 gives $ifd\big(ifd(x_2, 0, x_1), 1, ifd(x_2, ifd(x_1, t_6, t_6), t_6)\big)$, which can be reduced to $\mathrm{IFD}_2 = ifd\big(ifd(x_2, 0, x_1), 1, t_6\big)$ using the S reduction rule twice. Similarly, the sub-diagram consisting of non-terminal nodes *3* and *5* is represented as $t_3 = ifd\big(x_2, 0, ifd(x_1, t_6, 0)\big)$. Applying expansion (26) to t_3 gives $ifd\big(ifd(x_2, 0, x_1), t_6, ifd(x_2, ifd(x_1, 0, 0), 0)\big)$, which is reduced by the S reduction rule to $\mathrm{IFD}_3 = ifd\big(ifd(x_2, 0, x_1), t_6, 0\big)$.

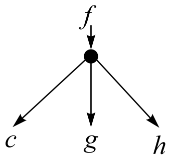
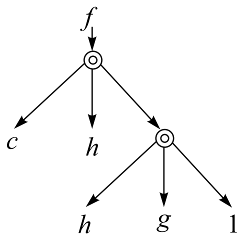
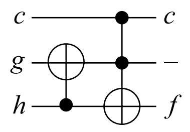
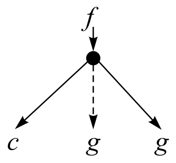
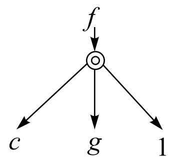
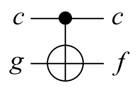
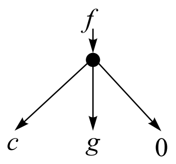
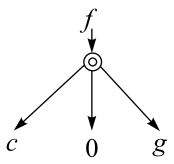
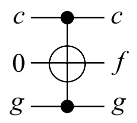
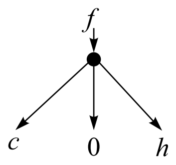
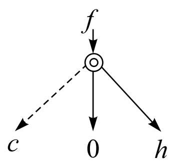
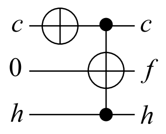
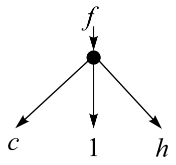
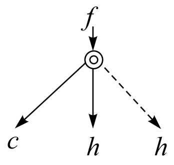
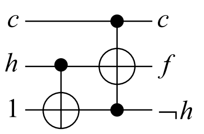
Merging the initial IFD root with IFD_2 and IFD_3 yields an intermediate IFD depicted in fig. 7, *b*. Its size is 4 non-terminal nodes, one less against the IFD shown in fig. 7, *a*. Since the outgoing *if*-edges of nodes *1*, *2* and *3* point two variables x_3 and z, and the *high*-edge of node *2* and the *low*-edge of node *3* point the same variable x_0, it is reasonable to apply expansion (25) to the IFD using $c = z \oplus x_3$. Term $ifd\big(\neg z, ifd(z, x_0, 0), ifd(z, 1, x_0)\big)$ describes the cofactor $f_{x_3=\neg z}$. It can be reduced to $ifd(\neg z, 0, 1) = z$. Term $ifd\big(z, ifd(z, x_0, 0), ifd(z, 1, x_0)\big)$ describes the cofactor $f_{x_3=z}$. It can be reduced to $ifd(z, x_0, x_0) = x_0$. Merging the diagrams that represent function $c = z \oplus x_3$, cofactor $f_{x_3=\neg z}$ and cofactor $f_{x_3=z}$ gives the final optimised IFD depicted in fig. 7, *c*. The diagram size is 3 non-terminal nodes, 2 nodes less than the size of the initial IFD shown in fig. 7, *a*.

Mapping *if*-decision diagrams to reversible circuits

The technique of mapping IFD to a reversible circuit works in two steps: 1) mapping the IFD to a FIFD; 2) mapping FIFD to a reversible circuit. Observing the IFDs depicted in fig. 7, we recognise five types of node. Table 2 presents rules of mapping each node of IFD to corresponding one or two nodes of FIFD.

Table 2

Rules of mapping IFD nodes to FIFD nodes and further to reversible gates

No.	Function	Term	IFD node	Equivalent FIFD nodes	Reversible gates
1	$f = c \wedge g \vee \neg c \wedge h$	$ifd(c,\ g,\ h)$			
2	$f = c \oplus g$	$ifd(c,\ \neg g,\ g)$			
3	$f = c \wedge g$	$ifd(c,\ g,\ 0)$			
4	$f = \neg c \wedge h$	$ifd(c,\ 0,\ h)$			
5	$f = c \vee h$	$ifd(c,\ 1,\ h)$			

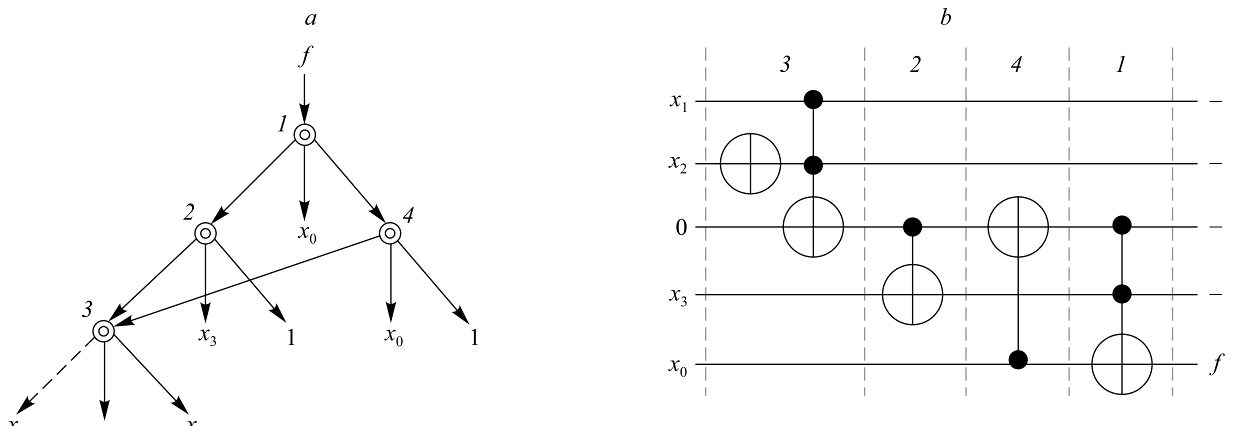


Fig. 8. Transforming optimised IFD to FIFD and then to reversible circuit:
a – FIFD functionally equivalent to optimised IFD (*1–4* denote the diagram node numbers);
b – reversible circuit realising the FIFD (diagram node numbers are above columns)

For each IFD-node type, table 2 describes the corresponding Boolean function, term, graphical view of the node, corresponding FIFD nodes, and reversible gates, which implement the nodes. The IFD nodes realise the functions as follows: $c \wedge g \vee \neg c \wedge h$, $c \oplus g$, $c \wedge g$, $\neg c \wedge h$ and $c \vee h$. Let apply the rules to the optimised IFD shown in fig. 7, *c*. Figure 8, *a*, depicts the functionally equivalent FIFD. To estimate the efficiency of the proposed transformation technique, we apply the rules to the IFD from fig. 7, *a*, and obtain the immediate FIFD (fig. 9, *a*). While the FIFD has the size of 7 non-terminal nodes and the depth of 4 nodes, the optimised FIFD has the size of 4 non-terminal nodes and the depth of 3 nodes. Mapping the FIFD nodes to reversible gates yields for optimised and immediate FIFDs the reversible circuits depicted in fig. 8, *b*, and 9, *b*, respectively.

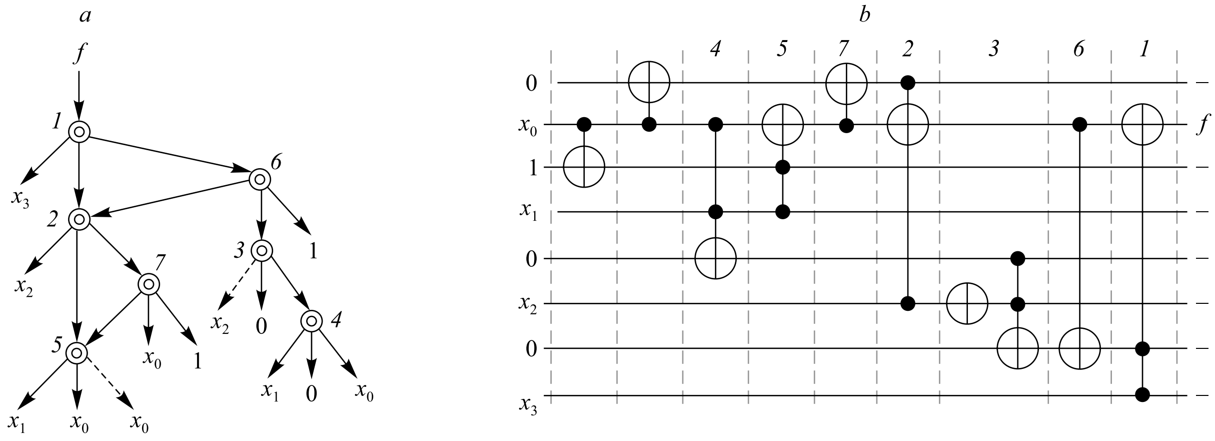


Fig. 9. Transforming immediate IFD to FIFD and then to reversible circuit:
a – FIFD that is functionally equivalent to immediate IFD (*1–7* denote the diagram node numbers);
b – reversible circuit realising the FIFD (diagram node numbers are above columns)

The comparison of three reversible circuits shows the efficiency of the proposed synthesis – transformation – optimisation technique. The circuit from fig. 5, *b*, synthesised from the ROFDD (see fig. 5, *a*) consists of 8 reversible gates, has the depth of 8 and uses 3 ancillary lines. The circuit from fig. 9, *b*, synthesised from the direct FIFD (see fig. 9, *a*) has the worsen parameters since it consists of 10 reversible gates, has the depth of 10 and uses 4 ancillary lines. The optimised circuit (see fig. 8, *b*) derived from FIFD has much better parameters since it consists of 5 reversible gates, has the depth of 5 gates and uses only 1 ancillary line.

Synthesis of quantum logic circuits from reversible circuits

The synthesis technique and parameters of quantum logic circuits essentially depend on the quantum library. Since NCV-$|v_1\rangle$ is a library that allows the generation of good quality quantum circuits [3], we have developed a technique of mapping the reversible gates of NCT library to quantum gates of NCV-$|v_1\rangle$ library, which consists of the following steps:

1) replacing all two-control Toffoli reversible gates with the cascade of quantum gates that is depicted in fig. 3, *b*; the procedure reduces the critical path of quantum circuit;

2) reducing the obtained circuit size by annihilating consequent V^+ and V gates and deleting gates for unessential line outputs;

3) parallelising the execution of quantum gates and reducing the circuit depth.

The procedure searches for such assignment of the V and V^+ gates to lines, which gives the lowest size and depth of resulting circuit. It has been applied to three reversible circuits shown in fig. 8, *b*, fig. 5, *b*, and fig. 9, *b*. Table 3 and fig. 10 show that the quantum circuit that is synthesised from the optimised IFD (see fig. 8, *b*) has the lowest size of 9 gates and the lowest depth of 8. The circuit synthesised from the ROBDD has a higher size of 16 gates and a higher depth of 11. The circuit synthesised from the direct IFD has a worst size of 21 gates and a worst depth of 17. The final and initial figures for the size and depth (see table 3) show that the size reduction of 4, 12 and 9 gates and the depth reduction of 1, 5 and 4 are obtained by the technique when applied to the optimised IFD, ROBDD and immediate IFD respectively.

Table 3

Comparison of synthesised reduced parallelised quantum circuits

Circuit	NOT gates	CNOT gates	V gates	CV gates	V^+ gates	CV^+ gates	Size (initial)	Depth (initial)	Ancillary lines
Optimised IFD	1	4	2	2	–	–	9 (13)	8 (9)	1
ROBDD	0	8	3	4	0	1	16 (28)	11 (16)	3
Immediate IFD	1	9	4	5	1	1	21 (30)	17 (21)	4

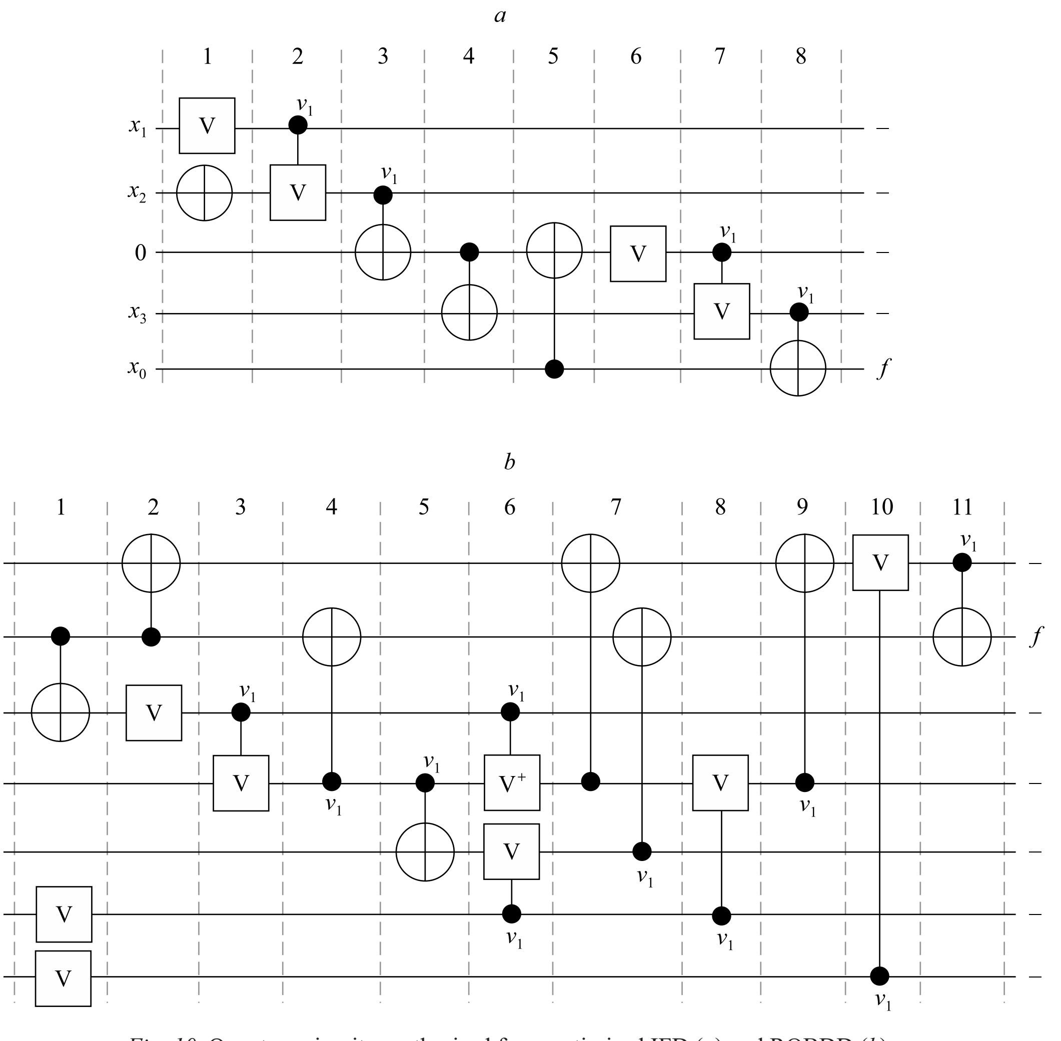


Fig. 10. Quantum circuits synthesised from optimised IFD (*a*) and ROBDD (*b*).
Numbers of columns indicate parallelisation steps

Conclusion

The problem of modelling, synthesis and optimisation of digital reversible and quantum circuits has been considered. All quantum circuits are reversible, therefore the first step in the quantum circuit synthesis process is the optimisation of reversible circuits with further mapping to quantum circuits. Although several models and methods have been developed for synthesis of reversible and quantum circuits (Reed – Muller forms or Zhegalkin polynomials, quantum operator forms, BDD, FDD and others), the problem of minimising the circuit size and depth is still open and requires further scientific research.

It is assumed in the literature that the ROBDD that is derived from the Shannon expansion with applying reduction rules is a most compact representation of Boolean functions. The promising extension of ROBDD is IFD, which is based on decomposition of incompletely specified functions and is capable of increasing the parallelisation potential of diagrams. This paper has obtained theoretical results which show that the IFD can provide graph representations that have lower size and depth against ROBDD. To synthesise IFDs with such properties, novel expansions of completely and incompletely specified Boolean functions that are based on the minimisation operation have been proposed in the paper. They allow for a particular diagram to find reconstructions that lead to reduction of the resulting diagram size and depth.

To synthesise a reversible circuit, the rules of transforming IFD to FIFD, which intensively use exclusive-or operations and extend the known functional binary decision diagrams have been proposed. New expansions of incompletely specified functions that extend the positive and negative Davio expansions of Boolean functions lie in the basis of constructing and generating FIFD. The FIFD is a source of generating a reversible circuit of smaller size by means of mapping diagram nodes to cascades of reversible gates. The technique of synthesising and optimising a quantum circuit from the generated reversible circuit has been developed in the paper. The optimisation aims at the reduction of quantum gate count after replacing Toffoli gates with cascades of quantum gates. It takes into account unessential output variables and carries out the reduction of the quantum circuit depth due to parallelisation. All stages of transforming ROBDD through IFD to a quantum circuit are illustrated by an example, which has shown the advantages of the obtained theoretical results.

References

1. Barenco A, Bennett CH, Cleve R, Di Vincenzo DP, Margolus N, Shor P, et al. Elementary gates for quantum computation. *Physical Review A.* 1995;52(5):3457–3467. DOI: 10.1103/PhysRevA.52.3457.

2. Nielsen M, Chuang IL. *Quantum computation and quantum information.* Cambridge: Cambridge University Press; 2000. 700 p.

3. Sasanian Z, Wille R, Miller DM. Realizing reversible circuits using a new class of quantum gates. In: Groeneveld P, editor. *The 49th Annual design automation conference 2012; 2012 June 3–7; San Francisco, USA*. New York: Association for Computing Machinery; 2012. p. 36–41.

4. Handique M, Sonka A. An extended approach for mapping reversible circuits to quantum circuits using NCV-$|v_1\rangle$ library. *Procedia Computer Science.* 2018;125:832–839. DOI: 10.1016/j.procs.2017.12.106.

5. Wille R, Chattopadhyay A, Drechsler R. From reversible logic to quantum circuits: logic design for an emerging technology. In: Najjar WA, Gerstlauer A, editors. *Proceedings of the 2016 International conference on embedded computer systems: architectures, modeling and simulation; 2016 July 17–21; Samos, Greece.* New York: IEEE; 2016. p. 268–274. DOI: 10.1109/SAMOS.2016.7818357.

6. Sasanian Z. Technology mapping and optimization for reversible and quantum circuits [dissertation]. Victoria: University of Victoria; 2012. 145 p.

7. Smith K. Technology-dependent quantum logic synthesis and compilation [dissertation] [Internet]. Dallas: Southern Methodist University; 2019 [cited 2021 September 9]. 133 p. Available from: https://scholar.smu.edu/engineering_electrical_etds/30.

8. Burgholzer L, Raymond R, Sengupta I, Wille R. Efficient construction of functional representations for quantum algorithms. In: Yamashita S, Yokoyama T, editors. *Reversible Computation. Proceedings of 13th International conference; virtual event; 2021 July 7–8.* Cham: Springer; 2021. p. 227–241. (Lecture Notes in Computer Science; volume 12805). DOI: 10.1007/978-3-030-79837-6_14.

9. Sasamala TN, Singh AK, Mohan A. Reversible logic circuit synthesis and optimization using adaptive genetic algorithm. *Procedia Computer Science.* 2015;70:407–413. DOI: 10.1016/j.procs.2015.10.054.

10. Bhattacharjee A, Bandyopadhyay C, Mukherjee A, Wille R, Drechsler R, Rahaman H. Efficient implementation of nearest neighbor quantum circuits using clustering with genetic algorithm. In: *2020 IEEE 50th International Symposium on Multiple-Valued Logic (ISMVL); 2020 November 9–11; Miyazaki, Japan.* Los Alamitos: IEEE; 2020. p. 40–45. DOI: 10.1109/ISMVL49045.2020.00-32.

11. Meuli G, Schmitt B, Ehlers R, Riener H, De Micheli G. Evaluating ESOP optimization methods in quantum compilation flows. In: Thomsen M, Soeken M, editors. *Reversible Computation. Proceedings of the 11th International conference; 2019 June 24–25; Lausanne, Switzerland.* Cham: Springer; 2019. p. 191–208. (Lecture Notes in Computer Science; volume 11497). DOI: 10.1007/978-3-030-21500-2_12.

12. Wille R, Drechsler R. Effect of BDD optimization on synthesis of reversible and quantum logic. *Electronic Notes in Theoretical Computer Science.* 2010;253(6):57–70. DOI: 10.1016/j.entcs.2010.02.006.

13. Zulehner A, Niemann P, Drechsler R. Accuracy and compactness in decision diagrams for quantum computation. In: *2019 Design, automation and test in Europe Conference and Exhibition (DATE); 2019 March 25–29; Florence, Italy.* [S. l.]: IEEE; 2019. p. 280–283. DOI: 10.23919/DATE.2019.8715040.

14. Lukac M, Kameyama M, Perkowski M, Kerntopf P. Minimization of quantum circuits using quantum operator forms. arXiv:1701.01999. 2017 [cited 2021 September 9]: [11 p.]. Available from: https://arxiv.org/abs/1701.01999.

15. Große D, Wille R, Dueck GW, Drechsler R. Exact multiple-control Toffoli network synthesis with SAT techniques. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems.* 2009;28(5):703–715. DOI: 10.1109/TCAD.2009.2017215.

16. Abdessaied N, Wille R, Soeken M, Drechsler R. Reducing the depth of quantum circuits using additional circuit lines. In: Dueck GW, Miller DM, editors. *Reversible Computation. Proceedings of the 5th International Conference; 2013 July 4–5; Victoria, BC, Canada.* Berlin: Springer; 2013. p. 221–233 (Lecture Notes in Computer Science; volume 7948). DOI: 10.1007/978-3-642-38986-3_18.

17. Prihozhy AA. *If*-diagrams: theory and application. In: *Proceedings of the 7th International workshop PATMOS'97; 1997 September 8–10; Louvain-la-Neuve, Belgium.* [S. l.]: UCL; 1997. p. 369–378.

18. Prihozhy AA. *Chastichno opredelennye logicheskie sistemy i algoritmy* [Incompletely specified logical systems and algorithms]. Minsk: Belarusian National Technical University; 2013. 343 p. Russian.

19. Prihozhy AA. Synthesis of parallel adders from *if*-decision diagrams. *System Analysis and Applied Information Science.* 2020; 2:61–70. DOI: 10.21122/2309-4923-2020-2-61-70.

20. Amarú L, Gaillardon P-E, De Micheli G. Biconditional BDD: a novel canonical BDD for logic synthesis targeting XOR-rich circuits. In: *2013. Design, Automation & Test in Europe Conference & Exhibition; 2013 March 18–22; Grenoble, France.* [S. l.]: IEEE; 2013. p. 1014–1017. DOI: 10.7873/DATE.2013.211.

Received 19.10.2021 / revised 21.10.2021 / accepted 29.10.2021.