ISSN 1814-4225 (print)
Radioelectronic and Computer Systems, 2024, no. 4(112) ISSN 2663-2012 (online)

UDC 004.8.27. **doi: 10.32620/reks.2024.4.10**

Taras KYRYLIUK, Mykhailo PALAHUTA, Vitaly DEIBUK

Yu. Fedkovych Chernivtsi National University, Chernivtsi, Ukraine

USING ARTIFICIAL INTELLIGENCE METHODS FOR THE OPTIMAL SYNTHESIS OF REVERSIBLE NETWORKS

Considering the relentless progress in the miniaturization of electronic devices and the need to reduce energy consumption, technical challenges in the synthesis of circuit design solutions have become evident. According to Moore's Law, the reduction of transistor sizes to the atomic scale faces physical limits, which complicate further development. Additionally, reducing transistor sizes causes current leakage, leading to increased thermal noise, which can disrupt the proper functioning of digital devices. A promising solution to these problems is the application of reversible logic in circuit design. Reversible logic allows for a reduction in energy and information losses because logical reversible operations are performed without loss. The research synthesized optimal reversible circuits based on reversible gates using evolutionary algorithms and compare them with existing analogues. The focus of this study is on logical circuits built using reversible gates, which can significantly reduce energy losses, which is critical for modern and future electronic devices. The synthesis of reversible circuits is closely related to quantum computing, where quantum gates also possess a reversible nature. This enables the use of synthesis methods to create quantum reversible logical computing devices, which in turn promotes the development of quantum technologies. The study focuses on the application of evolutionary artificial intelligence algorithms, specifically genetic algorithms and ant colony optimization algorithms, for the optimal synthesis of reversible circuits. As a result, a detailed description of the key concepts of the improved algorithms, simulation results, and comparison of the two methods is provided. The efficiency of the reversible device synthesis was evaluated using the proposed implementation of the genetic algorithm and the ant colony optimization algorithm. The obtained results were compared to existing analogs and verified using the Qiskit framework in the IBM quantum computing laboratory. The conclusions describe the developed algorithms, which demonstrate high efficiency in solving circuit topology optimization problems. A genetic algorithm was developed, featuring multi-component mutation and a matrix approach to chromosome encoding combined with Tabu search to avoid local optima. The ant colony optimization algorithms were improved, including several changes to the proposed data representation model, structure, and operational principles of the synthesis algorithm, enabling effective synthesis of devices on the NCT basis along with Fredkin gates. An improved structure for storing and using pheromones was developed to enable multi-criteria navigation in the solution space.

Keywords: *quantum computing; reversible circuits; synthesis; artificial intelligence; ant colony optimization; genetic algorithm; simulation; modelling.*

1. Introduction

The ever-increasing demands for miniaturizing electronic devices and reducing their energy consumption have recently become a pressing issue. According to Moore's Law [1], this trend, which has reached atomic size, has an inevitable limit. Furthermore, further reduction in the characteristic dimensions of transistors leads to leakage problems, and lowering the threshold voltage energy of digital devices results in increased thermal noise, which hinders their proper operation [2].

Motivation. One way to address these issues is to use reversible logic in circuit design [3]. This approach in the design of computational devices leads to reduced energy loss and, consequently, to reduced information loss. As has been demonstrated [4], to prevent information loss at the logical level, a circuit must consist of reversible gates. This approach has also proven useful in technologies such as low-power CMOS design [5], adiabatic circuits [6], encryption circuits [7], optical computing [8], bioinformatics, and quantum computing [9]. Reversible computing refers to the paradigm of building digital devices based on the bijective mapping of an n-input function to n-outputs. This allows for reversible computations, where each input signal combination results in a unique output combination and vice versa. Each output pattern allows the input to be restored. Such transformations do not lead to information loss at the logical level, unlike classical irreversible logic.

The construction of reversible logic gates, and consequent reversible devices based on them, is the main

task of reversible logic synthesis. Since quantum logic gates are reversible, which is a fundamental characteristic of quantum computing, the synthesis of reversible logic circuits is closely related to quantum logic synthesis, and methods for synthesizing reversible functions can be used for the implementation of quantum devices. At present, tools for quantum computing, such as Microsoft's Q#, Azure Functions, and IBM's Qiskit, are becoming available for scientific and commercial purposes [10].

State of the art. Evolutionary algorithms, unlike analytical methods, allow us to approach the problem of optimal synthesis of reversible networks without restrictions on the number of input variables and demonstrate good results in generating circuit solutions. In particular, genetic and ant algorithms, which utilize the principles of natural selection and ant colony behavior, enable the optimization of the topology of desired circuits in terms of hardware complexity, quantum cost, delay time, and other parameters.

One of the goals of this study is to develop and apply an improved genetic algorithm(GA) for the synthesis of optimal reversible networks [11,12]. We proposed a GA with multi-component mutation, a matrix approach to chromosome coding, and Tabu search [13] to escape local optima when synthesizing an optimal reversible circuit; the improvement of the circuit after its evaluation in the form of a pin exchange table (lines on which the gates are placed).

Another evolutionary algorithm used in this work for the optimal synthesis of reversible circuits is the ant colony algorithm (ACO), which has proven its promise in solving combinatorial problems, such as the traveling-salesman problem [14], protein-ligand docking [15], scheduling [16], etc. ACO combines a probabilistic decision-making model as a tool for overcoming stagnation problems with a distributed memory model, which is an effective way to find the attainable optimum. Thus, ACO can be used to build efficient and effective solutions, which has led to a more detailed analysis of its use for the synthesis of reversible devices. ACO was first applied for synthesizing reversible logic circuits in [17]. In this paper, we propose an improved approach in which an extended basis using Fredkin gates that preserve parity is used to enhance its performance.

The proposed ACO variation can synthesize circuits on any universal basis without any modifications. At the same time, the synthesis speed was significantly higher than that of the other basic sets.

This study aims to compare the abovementioned artificial intelligence methods for the reversible circuit synthesis task.

This article is organized as follows. Section 2 provides the basic concepts and terms used to describe the problems of reversible circuit synthesis are described in Section 2. Section 3 describes the improved GA and ACO algorithms. Section 4 presents the simulation results and a comparison of the two methods. The conclusions present the research results.

2. Objective and Approach. Reversible gates and circuits

In conventional (classical) computing, most logic gates (AND, OR, XOR, etc.) are irreversible. Thus, once the output values are determined, it is impossible to restore the initial state of the system, which results in information loss. When a bit of information is erased, energy is released in the form of heat, calculated as E = kTln2 Joules, where k is Boltzmann's constant and T is the absolute temperature of the environment. This principle was proposed by Landauer [18] and has been confirmed experimentally [19]. In ultra-small logic gates, heat dissipation is critical because it leads to errors and noise during computations. In reversible computing, and thus in circuits based on reversible gates, each input signal is uniquely mapped to the corresponding output signal, and vice versa, meaning that such mapping is bijective. This allows for computations with minimal energy loss, thus reducing errors.

Using the described approaches, reversible gates have been created—fundamental blocks of reversible and quantum computing—enabling reversible operations on quantum bits (qubits). There are several types of reversible gates, each with its own unique properties and applications. The simplest examples are the NOT gate (Fig. 1,a) for inversion, the CNOT gate (Fig. 1,b) for a controlled inversion gate, and the Toffoli gate, also known as CCNOT (Fig. 1,c) for a double controlled inversion gate, which form the universal NCT basis (Fig. 1). Another reversible logic gate is the Fredkin gate or CSWAP (Fig. 1,e), which performs a controlled swap operation if the control bit equals 1. Unlike the previous gates, the Fredkin gate preserves parity: the number of inputs equals the number of outputs. It is also universal, allowing the synthesis of complete devices based on it [20]. The control bits can control the operation of the respective gates using either a 1 or 0 signal (Fig. 1,d), significantly expanding the functional capabilities of the respective circuits; such a gate basis is called a mixed-polarity basis.

Based on the principles of gate operation, the concepts of Generalized Toffoli and Fredkin (Fig. 1,f) gates (GT, GF) with n controlled lines were introduced [21]. Reversible circuits are evaluated based on: the number of input lines, quantum cost, delay time (circuit depth), number of garbage outputs, and auxiliary inputs.

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

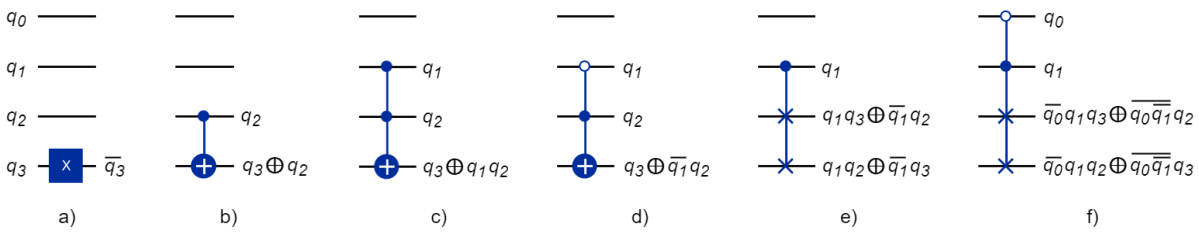


Fig. 1. The reversible logic gates with logical functions:
a – NOT; b – CNOT; c – CCNOT(Toffoli); d – C'CNOT; e – CSWAP(Fredkin); f – C'CSWAP

The quantum cost is often used as a comparative characteristic of reversible gates because quantum computing is one of the primary applications of reversible gates. The total quantum cost of a circuit is the sum of the quantum costs of each gate in the circuit [22].

The circuit depth is defined as the maximum number of quantum gates (operations) that must be executed sequentially to solve a specific computational procedure. The more parallel operations are performed, the better the circuit can be optimized, resulting in a smaller depth.

To implement reversible circuits, the garbage outputs and auxiliary inputs are added. Auxiliary inputs help achieve a better-optimized circuit with fewer gates and less depth and, in some cases, help to preserve parity. Garbage outputs are outputs whose values are not read but influence the activation of the respective gates, thereby allowing the synthesis of a circuit at a lower cost. However, adding constant inputs and garbage outputs complicates physical implementation and introduces the problem of managing information from garbage outputs.

Recently, evolutionary search algorithms have gained widespread application in the task of synthesizing quantum and reversible networks due to their simplicity of execution, ability for global search, and lack of restrictions on the number of input signals.

3. Methodology and implementation. Improved evolutionary algorithms

Evolutionary metaheuristic methods offer unique approaches to solving complex combinatorial problems by modelling natural phenomena based on populations [13]. Specifically, the Ant Colony Optimization (ACO) algorithm and the Genetic Algorithm (GA) are prominent methods. Let us consider these approaches in the context of the aforementioned problem and analyse their characteristics and propose improvements.

3.1. Genetic synthesis algorithm

The concept of chromosomes is used to represent quantum circuits, where each gene represents a vector of gates associated with a sequence of gates. Fig. 2 shows a class diagram of the chromosomes.

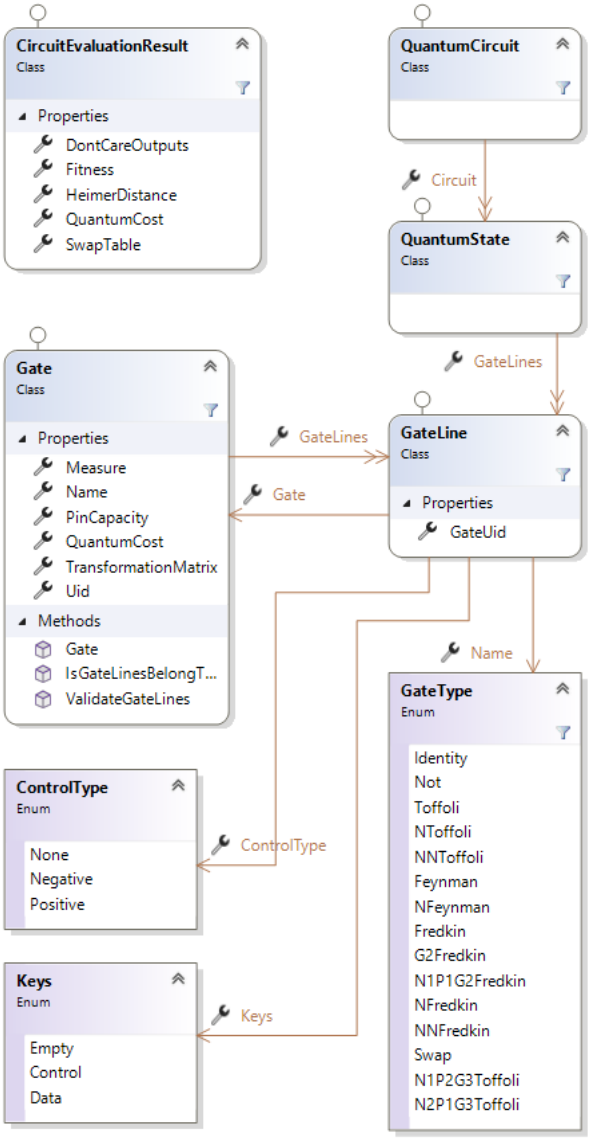


Fig. 2. DNA Class diagram

Encoding: The encoding of circuits is implemented using objects, providing a more flexible and transparent mechanism for the modifications and evaluation of quantum circuits during synthesis. A chromosome consists of two objects: a quantum reversible circuit (QuantumCircuit) and the evaluation results of the

synthesized circuit (CircuitEvaluationResult).

The circuit object (chromosome) contains a list of genes, QuantumState (G1 – G5), which is represented by a vector, GateLines (pins where gates are placed). Each GateLine contains a reference (i.e., a memory address) to a gate (Gate), with multiple lines possibly referencing the same gate. Each gene (QuantumState) is represented by a vector of GateLines arranged in parallel, with each GateLine in the vector positioned at a specific index (pin) to maximize available space. GateLines are grouped by the unique identifier GateUid of the current state (GateState) for the corresponding gate (Gate).

A chromosome is described by a matrix in which columns correspond to genes (GateState) with parallel gates placed. The rows of the matrix represent the pins of the circuit, where the respective gate contacts are located (Fig. 3). This chromosome representation eliminates the need for decoding prior to evaluation and increases the mutation and crossover capabilities of the genetic algorithm. This approach also allows for four-point crossover, exchanging not only multiple genes but also parts of genes and individual gates.

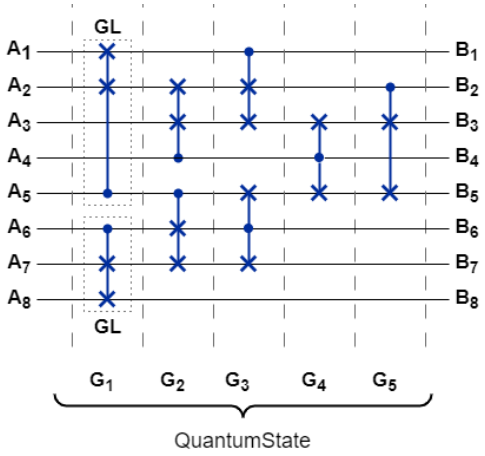


Fig. 3. Chromosome encoding

The other endpoint of the chromosome is the evaluation result of the synthesized circuit (CircuitEvaluationResult). The memory function for each individual chromosome contains information about the fitness function value, the number of errors (Hamming distance), quantum cost, the index of pins of garbage outputs, and the pin exchange table.

Creation of the Initial Population: The initial population is created by stochastically filling the chromosome with gates from the chosen basis (e.g., Fredkin, NOT, CNOT, Toffoli) based on the parameters described below.

The chromosome length N and the number of pins (lines where gates are placed) Q, along with the maximum number of generations G and the number of individuals in a generation Cmax, are set. Here, N and Q correspond to the number of columns and rows in the matrix, respectively.

Evaluation: Before calculating the fitness function, the truth table of the synthesized chromosome is computed. Each type of gate contains a calculation function for computing the gate output. The input vector of the first value of the truth table is fed into the first gene, and computations are performed for each parallel gate of the gene. The output value of the gate of the first gene is the input vector for the gates of the next gene. This operation is repeated until gene N is reached. After computing the output truth table of the obtained circuit, the fitness function is calculated as follows:

$$F = F(Hd) + F(QC), \quad (1)$$

where the hard constraint component is:

$$F(Hd) = \alpha(1 + Hd)^{-1}, \quad (2)$$

Hd – Hamming distance between the reference and obtained truth tables, α is a weighting coefficient. The soft constraint component:

$$F(Hd) = \beta(1 + Qc)^{-1}, \quad (3)$$

evaluates the optimality of the circuit relative to quantum cost, where Qc is the quantum cost of the evaluated circuit and is calculated as the sum of the quantum costs of the gates in the circuit, and β is a weighting coefficient. The optimization process aims to maximize the fitness function value for a given circuit, with $\alpha + \beta = 1$.

Genetic Operators: Selection proceeds as follows: E elites – individuals with the highest fitness function values are selected and migrate unchanged to the next generation. If the chromosome length is no less than two genes, a two-point crossover occurs with a probability Pc. Parental individuals are selected via roulette wheel selection.

The algorithm also implements multicomponent gene mutation with probability Pm. Mutation can occur in one of the following three ways, represented by probability coefficients Pms, Pma, and Pmr:

1. Pin Change (Pms): A gate pin in a randomly selected gene (QuantumState) is stochastically selected and replaced by another pin in the same gene.

2. Gate Addition of the selected basis (Pma). If the circuit satisfies the condition – there is a gene with more free pins (not occupied by gates) than the pins of the chosen basis gate – then with a probability Pma, a gate is added to the circuit at the first available position.

3. Gate removal: With probability Pmr, a random gate in the circuit is deleted.

It is worth noting that:

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

$$Pms + Pma + Pmr = 1. \quad (4)$$

Tabu List: After mutation, the uniqueness of chromosomes is evaluated, and a decision is made about adding them to the population. If the number of such chromosomes in the population exceeded 10 %, the individual was not included. If the number of identical individuals in the population exceeds 12 %, partial population removal occurs with the addition of new random instances [23].

3.2. Ant Colony Optimization

ACO and GA are heuristic optimization algorithms that mimic the organization of an ant colony. An artificial ant, in this context, represents one cycle of the path search from the colony's home to the food source, constructed around two path evaluation criteria: visible distance (Vd) and pheromone quantity (P). The algorithm performs separate depth and breadth path search cycles. This search, in turn, consists of sequential path selections until the colony's home is found. To select the next path, the weight Wn of each path n adjacent to the current location must first be calculated using the formula (5).

$$Wn = \alpha \cdot P(n) + \beta \cdot Vd(n), \quad (5)$$

where α and β are coefficients that regulate the extent to which the pheromone quantity on path n and the visible distance of this path affect the weight value, respectively.

The next step of the algorithm is the return of the ant at the end of the traversed path to the food source. The pheromone value P on path n is performed using formula (6).

$$P(n) = PE(n) \cdot er + dp, \quad (6)$$

where P(n) – is the amount of pheromone currently present on path n, dp – is the amount of pheromone P that the ant may to deposit; er – is the evaporation rate, a configured coefficient in the range (0, 1) responsible for simulating the gradual evaporation of existing pheromones left by previous generations of ants.

Applying ACO to Circuit Synthesis. In ACO implementations specialized for synthesis, it is assumed that the ant starts its journey with an empty circuit, i.e., one located at the food source. The next step initiates a path search cycle that gradually selects subsequent paths. In this case, these paths are logic gates. The search ends when the circuit formed by combining all gates of the path tour realizes the desired logic circuit. Thus, the home is the location of the desired circuit.

Returning, the ant leaves pheromone P along the path. These are calculated as the reciprocal of the path length tour (7), where gc is the gate count function. Thus, ants attract subsequent generations to shorter paths and hence to circuits with fewer gates. This criterion serves as an additional criterion for circuits based on gate count.

$$P = 1/gc(tour). \quad (7)$$

A logic gate is designed using two vectors containing a list of control and target bits used by the gate and the type of operation it applies. For example, a Toffoli gate can be represented as g(t,[c0,...,cn]), where t is the index of the target bit pin, and [c0,...,cn] is a vector, where cn determines the role of pin n with one of the following values: 0 means the pin is a positive control line, 1 means the pin is a negative control, and 2 means the pin is not used by the gate [24].

Based on the list of gates, we calculate the truth table of the solution circuit to analyse the logical function it realizes. However, storing truth tables in computer memory is inefficient due to their large volume. Therefore, as an alternative form of the truth table, we use permutation vectors, i.e., a vector where the index position corresponds to the decimal encoding of the input value, and the number in this position corresponds to the decimal encoding of the output value [25].

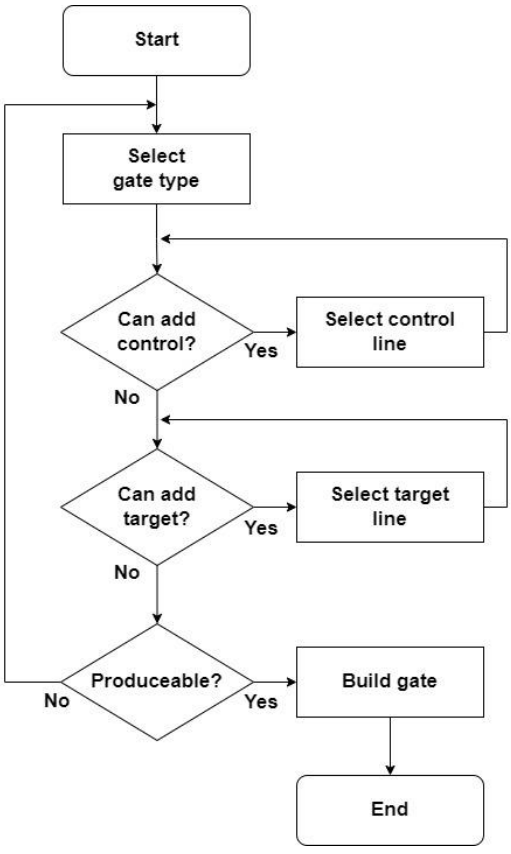


Fig. 4. Modified path selection algorithm

Extending ACO for Combined Basis Synthesis. In this study, unlike previous works dedicated to the

application of ACO to reversible network synthesis [17, 25], we consider for the first time the possibility of synthesizing circuits using the Fredkin gate and combinations of different types of gates as a basis. The following updates are proposed to the described standard structure of the algorithm to extend the synthesis capabilities of ACO:

1. It is proposed to include a new function in the path selection algorithm that determines the type of gate to be used. This function is added before selecting gate pins. Moreover, the process of pin selection now depends on the requirements defined by the chosen gate type, in terms of the number of control and target bits, constraints on where and how they can be placed in the circuit, etc. Thus, the path selection algorithm is illustrated in Fig. 4.

A graph of the paths that ants can traverse considering the proposed improvement is presented in Fig. 5.

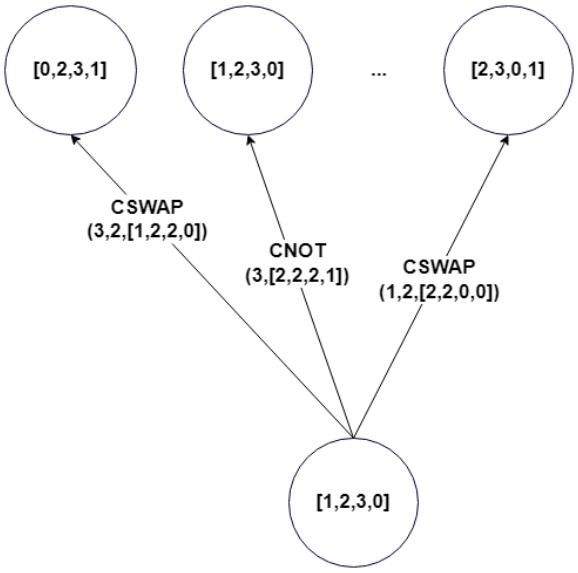


Fig. 5. Improved ACO Graph

2. Extending the pheromone deposit model to store variables on how often a certain type of gate placed in a specific part of the circuit appears in more successful ant routes. Subsequent generations of ants can use these pheromones, provided by their predecessors, for further movement. The pheromones used to select control and target lines should remain the same as in standard implementations. As a result, the functions selectGateType, selectControlLines, and selectTargetLines use separate sections of the pheromone collection phers (Fig. 5) for pins and gate types.

3. Using a universal method for modeling circuits that is independent of the gate type. As a result, any basic gate can be represented in the proposed algorithm without needing to modify it. The traditional implementation has a limitation in that it always expects the gate to have exactly one target line. To solve this problem, we change the model from g(t,[c0 ,..., cn]) to g([t0, ..., tm], [c0, ..., cn]), where [t0, ..., tm] is a vector containing a list of numbers or identifiers of the circuit lines to be considered as target bits.

The proposed additions are not exclusively intended to use a combined basis of gates during synthesis but to introduce the necessary flexibility into the ACO implementation for reversible device synthesis. As a result, this version of ACO can be used for circuit synthesis on any reversible basis. Using a combined basis increases the space of possible solutions, which complicates the synthesis. In this paper, we present the results of circuit synthesis on different bases and compare them with circuits synthesized by the GA algorithm.

4. Discussion and simulation results

To evaluate the effectiveness of reversible device synthesis using the above methods, we used the full reversible one-bit adder and circuits based on test functions [25] as test cases. Fig. 6,a shows the genetic synthesis results of a quantum circuit for a full one-bit adder. During the synthesis process, a better solution was found compared to the known ones [28, 30], using two constant inputs 0, 1; G1, G2 – garbage outputs; A, B – input bits; Cin, Cout – carry registers; and S – the sum. The resulting circuit contains only Fredkin gates; thus, it is fault-tolerant (reversible and parity-preserving) and has hardware complexity Gc = 5 with a quantum cost Qc = 25 (the quantum cost of a Fredkin gate is 5).

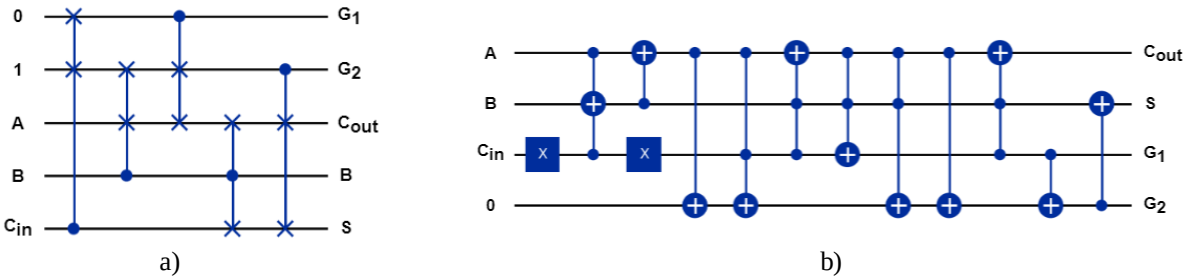


Fig. 6. Full one-bit adder: a) GA produced circuit, b) ACO produced circuit

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

Table 1

Full adder comparison results

	ACO		GA	
	proposed	[26]	proposed	[12]
Cost	37	12	25	25
Delay	12	4	5	5
Gates count	13	4	5	5
Garbage outputs	2	2	2	2
Auxiliary inputs	1	1	2	2
Basis	NCT		Fredkin gate	

For the described GA, the following parameters were applied: chromosome length N = 5; number of pins Q = 5; maximum number of generations G = 1200; population size Cmax = 100; $\alpha = 0.9$; $\beta = 0.1$; number of elite individuals passed unchanged to the next generation E = 8; mutation probability Pm = 0.1; probability of changing the gene gate line Pms = 0.4; probability of adding a new gate to the circuit Pma = 0.1; probability of removing a gate from the circuit Pmr = 0.5; crossover probability Pc = 0.7; and solution synthesis time = 2.1747836 s. Fig. 6,b shows the result of synthesizing the one-bit adder circuit using ACO on the NCT basis. Synthesis parameters: number of ants = 35, number of iterations = 20, pheromone influence coefficient = 3, visible distance influence coefficient – (-1), pheromone evaporation coefficient = 0.4. The circuit synthesis time was approximately 2 s. The comparative characteristics of the synthesized adders are presented in Table 1.

As can be seen, the circuit synthesized using GA replicated the results of the previous work [12]. At the same time, the circuit synthesized by ACO on the NCT basis is less optimal than that in [26].

Table 2 presents the synthesis results of various reversible logic functions using ACO and GA compared with the results reported by other authors. The functions are presented as permutation vectors, as described above. The synthesis results of the reversible functions correspond to the existing counterparts and, in some cases, improve them [25, 26]. Comparing the two algorithms, we see that ACO performs better in synthesizing simple functions ([0,1,2,3,4,6,5,7], [0,1,2,4,3,5,6,7]). For the function [1,2,3,4,5,6,7,0] both algorithms achieved parity in the obtained parameters, while for the remaining functions GA synthesized circuits with better parameters.

The synthesized circuits were exported to QASM (Quantum Assembly Language) and verified in the IBM Quantum lab [10]. The design of the produced circuits, as shown in Table 3.

Comparing the obtained results with [26], we found that, in some cases, GA synthesizes better results. Most circuits include swap gates, which can be replaced with direct data reading from the corresponding pin in hardware implementation to improve the characteristics of synthesized circuits.

Table 2

Comparing the synthesis results by gate counts

Function	ACO		GA	
	proposed	[25]	proposed	[26]
[0,1,2,3,4,6,5,7]	1	3	3	3
[0,1,2,4,3,5,6,7]	3	3	4	5
[1,0,3,2,5,7,4,6]	3	3	2	4
[1,2,3,4,5,6,7,0]	3	3	3	3
[1,2,7,5,6,3,0,4]	16	6	9	7
[3,6,2,5,7,1,0,4]	12	6	8	7
[4,3,0,2,7,5,6,1]	9	5	6	6
[7,0,1,2,3,4,5,6]	5	3	3	3

5. Conclusion

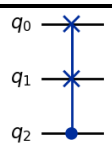
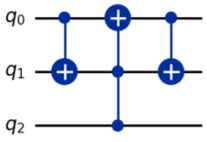
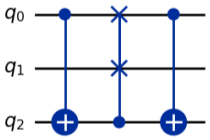
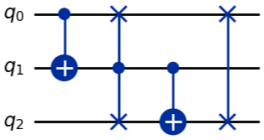
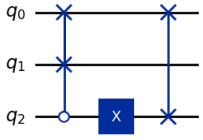
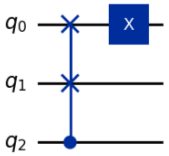
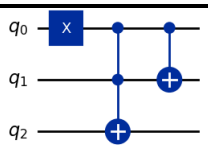
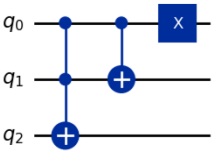
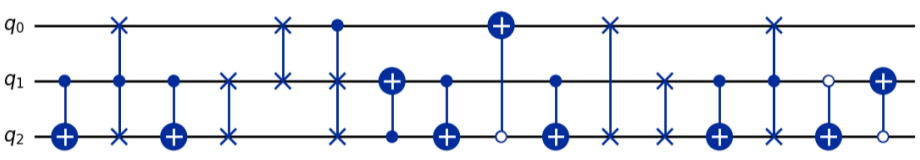
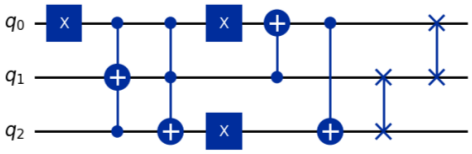
This study presents a novel matrix-coding method for chromosomes used in a genetic algorithm (GA), incorporating a Tabu list to address the problem of stagnation. We propose several modifications to the data representation model, structure, and working principles of the ant colony optimization (ACO) algorithm for reversible circuit synthesis. These modifications allow the effective synthesis of devices using arbitrary gate bases. Developed an improved pheromone storage and usage structure that enables multi-criteria navigation in the solution space. The efficiency of the reversible device synthesis was evaluated using the proposed implementations of the GA and ACO algorithms. The synthesis of a reversible full one-bit adder circuit using the GA demonstrated high efficiency, achieving an optimal solution in the 100th generation with hardware complexity Gc = 5 and quantum cost Qc = 25 and a synthesis time of 2.1747836 s. The ACO method showed a quick resolution of the one-bit adder synthesis problem, with a synthesis time of approximately 2 s, and visualization of the primary ant path indicated a typical search depth of approximately 10 possible solutions.

Comparison of the methods confirmed that GA was more effective for complex functions, whereas ACO was more efficient for simpler functions. The synthesis results in a set of functions indicate that both algorithms have advantages. Comparison with existing solutions suggests the potential to improve current solutions in some cases. Overall, the synthesis of reversible devices using GA and ACO was effective for creating reliable and fast circuits suitable for various high-performance and accurate computational systems.

Future research directions. In future, we will focus on developing high-performance encryption devices based on synthesized reversible logic circuits using meta-heuristic methods.

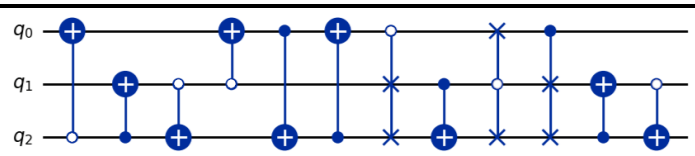
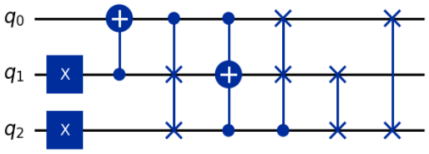
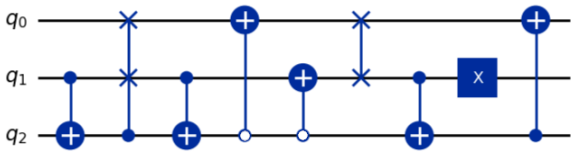
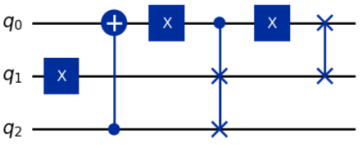
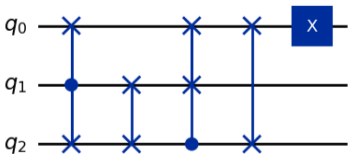
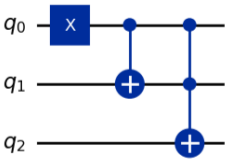
Table 3

Synthesis results

Function	Algorithm	Qc	Delay	Circuits
[0,1,2,3,4,6,5,7]	ACO	5	1	
	GA	7	3	
[0,1,2,4,3,5,6,7]	ACO	7	3	
	GA	8	4	
[1,0,3,2,5,7,4,6]	ACO	8	3	
	GA	6	2	
[1,2,3,4,5,6,7,0]	ACO	7	3	
	GA	7	3	
[1,2,7,5,6,3,0,4]	ACO	31	16	
	GA	17	8	

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

Continuation of theTable 3

[3,6,2,5,7,1,0,4]	ACO	30	12	
	GA	20	7	
[4,3,0,2,7,5,6,1]	ACO	15	9	
	GA	10	5	
[7,0,1,2,3,4,5,6]	ACO	13	5	
	GA	7	3	

Contributions of authors:

Taras Kyryliuk – development of model, software, verification, writing, original draft preparation, analysis of results;

Mykhailo Palahuta – methodology, development of model, software, verification, writing, original draft preparation, analysis of results;

Vitaly Deibuk – supervision, conceptualization, formulation of tasks, analysis of results.

Conflict of interest

The authors declare that they have no conflict of interest to this research, whether financial, personal, authorship, or otherwise, that could affect the research and its results presented in this paper.

Financing

This research was conducted without financial support.

Data availability

This study has data associated with the data repository.

Use of Artificial Intelligence

The authors confirm that they did not use artificial intelligence methods in their work.

All the authors have read and agreed to the published version of the manuscript.

References

1. Moore, G. E. Cramming more components onto integrated circuits, reprinted from electronics, volume 38, number 8, April 19, 1965, pp.114 ff. *IEEE Solid-State Circuits Society Newsletter*, 2006, vol. 11, iss. 3, pp. 33–35. DOI: 10.1109/n-ssc.2006.4785860

2. Shauly, E. N. CMOS leakage and power reduction in transistors and circuits: process and layout considerations. *Journal of Low Power Electronics and Applications*, 2012, vol. 2, iss. 1, pp. 1–29. DOI: 10.3390/jlpea2010001

3. Vos, A., Baerdemacker, S., & Rentergem, Y. *Synthesis of Quantum Circuits vs. Synthesis of Classical*

Reversible Circuits. Springer International Publishing, Cham. 2018. 109 p. DOI: 10.1007/978-3-031-79895-5

4. Bennett, C. H. Logical reversibility of computation. *IBM Journal of Research and Development,* 1973, vol. 17, iss. 6, pp. 525–532. DOI: 10.1147/rd.176.0525

5. Weste, N., & David, H. *CMOS VLSI design: a circuits and systems perspective*. Pearson Education, Limited, 2013. 743 p

6. Akshata, S., Arpitha, E., Deepashree, V. A., & Pushpa, L. N. D. Low power comparator design using reversible logic gates – adiabatic circuits. *International journal of engineering research & technology*, 2018, vol. 6, iss. 13 Available at: https://www.ijert.org/research/low-power-comparator-design-using-reversible-low-power-comparator-design-using-reversible-IJERTCONV6IS13214.pdf (accessed 16 April 2024)

7. Szymański, Z., & Pawłowski, M. Symmetric block encoder based on reversible circuits. R. S. Romaniuk and M. Linczuk, eds. *Photonics applications in astronomy, communications, industry, and high-energy physics experiments,* 2018. DOI: 10.1117/12.2501438

8. Ying, Z., Feng, C., Zhao, Z., Soref, R., Pan, D., & Chen, R. T. Integrated multi-operand electro-optic logic gates for optical computing. *Applied Physics Letters*, 2019, vol. 115, iss. 17, 171104. DOI: 10.1063/1.5126517

9. Wang, B., Xie, Y., Zhou, S., Zhou, C., & Zheng & X. Reversible data hiding based on DNA computing. *Computational Intelligence and Neuroscience*, 2017, pp 1–9. DOI: 10.1155/2017/7276084

10. Wille, R., Van Meter, R. & Naveh, Y. IBM's qiskit tool chain: working with and developing for real quantum computers. *2019 design, automation & test in Europe conference & exhibition (DATE),* Florence, Italy. IEEE, 2019, pp. 1234-1240. DOI: 10.23919/date.2019.8715261

11. Dovhaniuk, O. & Deibuk, V. Synthesis and implementation of reconfigurable reversible generalized Fredkin gate. *2021 IEEE 12th International Conference on Electronics and Information Technologies (ELIT),* Lviv, Ukraine, 2021, pp. 165-169. DOI: 10.1109/elit53502.2021.9501129

12. Deibuk, V. & Grytsku, I. Optymal'nyy syntez zvorotnykh kvantovykh sumatoriv z dopomohoyu henetychnykh alhorytmiv [Optimal synthesis of reversible quantum summators using genetic algorithm]. *Journal of Computing*, 2013, vol. 12, iss. 1, pp. 32–41. Available at: https://computingonline.net/computing/article/download/585/547/0 (accessed 16 April 2024) (In Ukrainian).

13. Gendreau, M. & Potvin, J.-Y, Tabu search. *Handbook of metaheuristics.* Springer International Publishing, Cham, 2018, pp. 37-55. DOI: 10.1007/978-3-319-91086-4

14. Held, M. & Karp, R. M. The traveling-salesman problem and minimum spanning trees: Part II. *Mathematical Programming,* 1971, vol. 1, iss. 1, pp. 6–25. DOI: 10.1007/bf01584070

15. Korb, O., Stützle, T. & Exner, T. E. An ant colony optimization approach to flexible protein–ligand docking. *Swarm Intelligence*, 2007, vol. 1, iss. 2, pp. 115–134. DOI: 10.1007/s11721-007-0006-9

16. Erdoğan, G., Laporte, G. & Rodríguez Chía, A. M. Exact and heuristic algorithms for the Hamiltonian p -median problem. *European Journal of Operational Research*, 2016, vol. 253, iss. 2, pp. 280–289. DOI: 10.1016/j.ejor.2016.02.012

17. Min Li, Yexin Zheng, Hsiao, M. S. & Chao Huang. Reversible logic synthesis through ant colony optimization. *2010 design, automation & test in europe conference & exhibition (DATE 2010),* Dresden, Germany, IEEE, 2010. DOI: 10.1109/date.2010.5457190

18. Landauer, R. Irreversibility and heat generation in the computing process. *IBM Journal of Research and Development*, 1961, vol. 5. iss. 3, pp. 183–191. DOI: 10.1147/rd.53.0183

19. Bérut, A., Petrosyan, A. & Ciliberto, S. Information and thermodynamics: experimental verification of Landauer's Erasure principle. *Journal of Statistical Mechanics: Theory and Experiment*, 2015, vol. 2015, iss. 6. DOI: 10.1088/1742-5468/2015/06/p06015

20. Deibuk, V., Dovhaniuk, O., Kyryliuk, T. The Extended Fredkin Gates with Reconfiguration in NCT Basis. *Advances in Computer Science for Engineering and Education VI.* Springer, Cham, 2023, vol. 181, pp. 95-105. DOI: 10.1007/978-3-031-36118-0_9

21. Maslov, D., Dueck, G. W. & Miller, D. M. Techniques for the synthesis of reversible Toffoli networks. *ACM Transactions on Design Automation of Electronic Systems,* 2007, vol. 12, iss. 4, 42 p. DOI: 10.1145/1278349.1278355

22. Maslov, D. & Dueck, G.W. Comparison of the Cost Metrics for Reversible and Quantum Logic Synthesis, 2005, *arXiv preprint quant-ph/0511008*. Available at: https://arxiv.org/pdf/quant-ph/0511008 (accessed 16 April 2024)

23. Han, J., Zhang, X. & Wang, X. Application research of evolutionary algorithm in synthesis of reversible logic circuits. *Journal of Physics: Conference Series,* 2019. vol. 1237, iss. 2. DOI: 10.1088/1742-6596/1237/2/022083

24. Sarif, B. A. B., Abd-El-Barr, M., Sait, S. M. & Al-Saiari, U. Fuzzified ant colony optimization algorithm for efficient combinational circuits synthesis. *Proceedings of the 2004 Congress on Evolutionary Computation (IEEE Cat. No.04TH8753),* Portland, OR, USA, 2004, vol. 2, pp. 1317-1324. DOI: 10.1109/cec.2004.1331049

25. Podlaski, K. Ant colony optimization implementation for reversible synthesis in Walsh-Hadamard domain. *Lecture notes in computer science*,

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

Springer International Publishing, Cham, 2020, vol. 12141, pp. 230–243. DOI: 10.1007/978-3-030-50426-7_18

26. Sasamal, T. N., Gaur, H. M., Singh, A. K. & Mohan, A. Reversible circuit synthesis using evolutionary algorithms. *Lecture notes in electrical engineering*, Springer Singapore, Singapore, 2019, vol. 577, pp. 115–128. DOI: 10.1007/978-981-13-8821-7_7

Received 14.06.2024, Accepted 18.11.2024

ВИКОРИСТАННЯ МЕТОДІВ ШТУЧНОГО ІНТЕЛЕКТУ ДЛЯ ОПТИМАЛЬНОГО СИНТЕЗУ ЗВОРОТНИХ МЕРЕЖ

Т. П. Кирилюк, М. І. Палагута, В. Г. Дейбук

У світлі невпинного прогресу в мініатюризації електронних пристроїв і необхідності скорочення споживання енергії стають очевидними технічні проблеми синтезу схемотехнічних рішень. Відповідно до закону Мура, зменшення розмірів транзисторів до атомного масштабу стикається з фізичними межами, що ускладнює подальший розвиток. Крім того, зменшення розмірів транзисторів спричиняє витоки струму, що призводить до збільшення теплового шуму, який може порушити належне функціонування цифрових пристроїв. Перспективним рішення цих проблем є застосування зворотної логіки в схемотехніці. Зворотна логіка дозволяє зменшити втрати енергії та інформації, оскільки логічні зворотні операції виконуються без втрати. Метою дослідження є синтез оптимальних зворотних схем на базі зворотних вентилів за допомогою еволюційних алгоритмів і порівняння їх з існуючими аналогами. Об'єктом дослідження виступають логічні схеми, побудовані з використанням зворотних вентилів, які можуть значно зменшити втрати енергії, що є критичним для сучасних і майбутніх електронних пристроїв. Синтез зворотних схем тісно пов'язаний з квантовими обчисленнями, де квантові вентилі також мають зворотну природу. Це дозволяє використовувати методи синтезу для створення квантових зворотних логічних обчислювальних пристроїв, що, у свою чергу, сприяє розвитку квантових технологій. Предметом дослідження є застосування еволюційних алгоритмів штучного інтелекту, зокрема генетичних алгоритмів і алгоритмів оптимізації колонії мурах, для оптимального синтезу зворотних схем. Як результат наведено детальний опис ключових понять вдосконалених алгоритмів, результатів моделювання та порівняння двох методів. Ефективність синтезу зворотного пристрою оцінювали за допомогою запропонованої реалізації генетичного алгоритму та алгоритму оптимізації мурашиної колонії. Отримані результати були порівняні з існуючими аналогами та верифікувалися за допомогою фреймворку Qiskit у лабораторії квантових обчислень IBM. Висновки описують розроблені алгоритми, що показують високу ефективність у вирішенні проблем оптимізації топології схеми. Було розроблено генетичний алгоритм, з багатокомпонентною мутацію та матричним підходом до кодування хромосом у поєднанні з пошуком Табу, щоб уникнути локальних оптимумів. Покращено алгоритми оптимізації мурашиної колонії включаючи кілька змін у запропонованій моделі представлення даних, структурі та принципах роботи алгоритму синтезу, що дозволяє ефективно синтезувати пристрої на основі NCT разом із вентилями Фредкіна. Було розроблено вдосконалену структуру для зберігання та використання феромонів, що дозволяє здійснювати багатокритеріальну навігацію в просторі рішень.

Ключові слова: квантові обчислення; зворотні схеми; синтез; штучний інтелект; мурашиний алгоритм; генетичний алгоритм; симуляція; моделювання.

Кирилюк Тарас Петрович – асп. каф. програмного забезпечення комп’ютерних систем, Чернівецький національний університет ім. Ю. Федьковича, Чернівці, Україна.

Палагута Михайло Ілліч – асп. каф. програмного забезпечення комп’ютерних систем, Чернівецький національний університет ім. Ю. Федьковича, Чернівці, Україна.

Дейбук Віталій Григорович – д-р фіз.-мат. наук, проф., проф. каф. комп’ютерних систем та мереж, Чернівецький національний університет ім. Ю. Федьковича, Чернівці, Україна.

Taras Kyryliuk – PhD Student of the Computer System’s Software Department, Yuriy Fedkovych Chernivtsi National University, Chernivtsi, Ukraine,
e-mail: kyryliuk.taras@chnu.edu.ua, ORCID: 0009-0004-5033-1200.

Mykhailo Palahuta – PhD Student of the Computer System’s Software Department, Yuriy Fedkovych Chernivtsi National University, Chernivtsi, Ukraine,
e-mail: palahuta.mykhailo@chnu.edu.ua, ORCID: 0009-0003-4896-903X.

Vitaly Deibuk - Dr. Sci. (Phys. & Math.), Professor, Professor at the Computer Systems and Networks Department, Yuriy Fedkovych Chernivtsi National University, Chernivtsi, Ukraine,
e-mail: v.deibuk@chnu.edu.ua, ORCID: 0000-0002-1216-0031.