Ant Colony Optimization Implementation for Reversible Synthesis in Walsh-Hadamard Domain

Krzysztof Podlaski[1][0000−0002−2883−0773]

Faculty of Physics and Applied Informatics, University of Lodz, Lodz, Poland
podlaski@uni.lodz.pl

Abstract. Reversible circuits are one of the technologies that can provide future low energy circuits. The synthesis of an optimal reversible circuit for a given function is an np-hard problem. The meta-heuristic approaches are one of the most promising methods for these types of optimization problems. In this paper, a new approach for ACO reversible synthesis is presented. Usually, authors build an ACO system with the use of truth table or permutation representation of the reversible function. In this work, a Walsh spectral representation of a Boolean function is used. This allows dividing search spaces into smaller "promising" areas with well-defined transition operations between them. As a result, we can minimize the enormous search space and generate better solutions than obtained by ACO synthesis with classical reversible function representation. The proposed approach was applied to benchmark reversible functions of 4,5 and 6 variables and compared to other meta-heuristic results and best-known solutions.

Keywords: aco · reversible circuits · synthesis · spectral methods · walsh spectrum

1 Introduction

One of the most important requirements for the development of new electronic devices is power consumption. With rapid minimization and growing demand for computation power, the low-power design is under constant research. It is well-known that with any loss of information the energy is dissipated [15]. On that base, the reversible circuits, i.e the circuits that do not lose information during computation, are recognized as one of the promising alternatives for future low-power design [3, 10]. It should be mentioned that reversible logic is strictly connected with another promising technology - quantum computing.

Reversible circuits can be synthesized with the use of basic gates like not, cnot, Toffoli. However, this synthesis is very different from the synthesis of classical circuits [23]. Many heuristic methods of reversible synthesis have been proposed in the literature, to name a few: transformation based algorithm [19, 20], cycle-based algorithm [24], decision diagram based algorithms [28]. Most of the known methods are very redundant or can be applied to a reversible function

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50426-7_18

with a small number of inputs [14]. Some of the authors proposed also the use of meta-heuristic methodologies like genetic algorithms [11,27], particle swarm optimization [4,18], and ant colony optimization [17]. All presented meta-heuristic approaches use truth-table or permutation representation of a reversible function, in this paper a new ant colony optimization approach that uses the spectral representation of a reversible function is used.

The paper is organized as follows. In Section 2 basic concepts connected to reversible logic are introduced. Section 3 contains a description of Walsh-Hadamard spectral methods. Section 4 is devoted to the general ACO system while in section 5 a detailed description of the algorithm is presented. Section 6 contains a discussion of the results obtained. The concluding remarks are included in the last Section 7.

2 Reversible Logic

In this Section, the basic definitions and ideas connected to reversible logic are presented for the convenience of the reader.

Definition 1 (Balanced function). *A Boolean function* $f : \{0,1\}^n \to \{0,1\}$ *is called balanced if it takes the value 1 the same number of times as 0.*

Definition 2 (Reversible function). *A mapping* $f : \{0,1\}^n \to \{0,1\}^n$ *is called a reversible function if it is bijective.*

Definition 3 (Component function). *A reversible function* $f(x), x \in \{0,1\}^n$ *can be considered as a vector of Boolean functions* $f = (f_1, f_2, ..., f_n)$*, each of these functions* f_i *will be called component functions.*

Table 1: Truth table of an exemplary $3 * 3$ reversible function f, the columns in/out uses decimal signal encoding, while two middle ones represent signals encoded as a binary string.

in	x_1	x_2	x_3	f_1	f_2	f_3	out
0	0	0	0	0	0	1	1
1	0	0	1	0	1	0	2
2	0	1	0	0	0	0	0
3	0	1	1	1	1	1	7
4	1	0	0	1	0	1	5
5	1	0	1	0	1	1	3
6	1	1	0	1	1	0	6
7	1	1	1	1	0	0	4

Any reversible function can be represented in many ways for example as a truth table, binary decision diagram, additionally, every reversible function is bijective onto, and as such, it can be written as a permutation of input signals.

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50426-7_18

From the truth table representation of a reversible function (Table 1), we can see that the function F is a bijection, every input signal appears once as an output. Every output $f(1) = f(001) = 010 = 2$ is an element of a set of all possible inputs, that is why the function can be represented as a permutation $[1, 2, 0, 7, 5, 3, 6, 4]$. All of these representations are equivalent. The component functions f_1, f_2, f_3 are connected to appropriate columns in the truth table.

A reversible circuit is a circuit that realizes a reversible function, i.e. it performs a bijective mapping of an n-bit input signal onto an n-bit output signal, the mapping is defined by a given reversible function. The circuit can be reversible if all internal operations are reversible, which means all building blocks of a reversible circuit have to be reversible themselves. The classical digital circuits are based on gates like AND and OR, these gates are not reversible, moreover, their functions are not balanced, this implies that these gates cannot be used in a reversible circuit. Additionally, in reversible circuit fan-outs are forbidden, this implies that a reversible circuit is a cascade of reversible gates [22]. The most often used library of basic reversible gates is known as multiple control Toffoli gates (MCT) and contains three types of gates: not, cnot, Toffoli.

1. T1(s) - not gate, negates the signal on line s,
2. T2($k; s$) - controlled not gate, negates the signal on line s if the signal on control line k is equal to 1,
3. T3($k, l; s$) - Toffoli gate, negates the signal on line s if the signals on controlled lines k and l are equal to 1,
4. Tm($k_1, ..., k_{m-1}; s$) - generalized Toffoli gate, negates the signal on line s if the signals on all $m - 1$ controlled lines $k_1, ..., k_{m-1}$ are equal to 1,

Each of the MCT gates is self-inverse. It is known that any $n * n$ reversible function can be implemented with the use of the gates from the MCT library. Moreover, the number of different gates one can use to synthesize $n*n$ reversible circuit increase with the number n of input binary variables, for example for 4*4 reversible domain, there are 32 available MCT gates: 4*T1, 12*T2 and T3, 4*T4.

2.1 Gate Cost

The simplest approach to evaluate the quality of a circuit is a gate count (GC), this measure, however, treats all gates in the same way. It is rather obvious that different gates will have different implementation cost, this implementation cost can differ for technology used. The MCT gate library contains many different gates, from the simplest ones T1 (not gate) to a very complex like T5 (generalized Toffoli gate with 5 input lines and 4 lines are control ones). In the literature, there are a few approaches to describe the sophistication of reversible gates. The most recognized measures of gate costs are the so-called quantum cost (QC) [2] and T-count [1, 21]. All these measures are connected to the representation of reversible gates in quantum gates. QC and T-count measures grow rapidly with the growth of the number of control lines. For gates used in the paper: T1, T2 have QC=1, T-count=0, T3: QC=5, T-count = 7, T4: QC = 13, T-count=16. In

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50426-7_18

this paper, quantum cost measure is used as it has similar behavior comparing to T-count and has a nonzero cost to two basic gates not and cnot. The following synthesis procedure is designed to optimize the QC of obtained circuits.

The process of reversible synthesis of a given function f is a task of generating the optimal sequence of reversible gates, that transforms input signals into outputs with the agreement with function f. In this paper, the optimal sequence means the sequence with the lowest total quantum cost. However the meta-heuristic methodology is used and by default part of the solutions can be suboptimal, the goal of the presented approach is to find solutions as near to optimal as possible.

3 Walsh-Hadamard Transform

In the previous section two different representation of reversible function was mentioned (truth table and permutation). In this paper, an additional - spectral representation is used. In the domain of Boolean functions a few generalized Fourier type transforms are well-known, i. e. Reed-Muller, Arithmetic, and Walsh [26]. Each of these transforms can be used to define spectral representation with different properties and was used for some time in the theory of Boolean functions. In this paper the Walsh transform in Hadamard order is used, called also Walsh-Hadamard transform.

Definition 4 (Walsh transform). *In n variable Boolean domain the Walsh-Hadamard transform is defined by the Kronecker product $\otimes$ of basic Walsh matrix*

$$W(n) = \bigotimes_n W(1), \qquad \text{where } W(1) = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}. \tag{1}$$

To apply Walsh-Hadamard transform to a Boolean function, we have to apply integer encoding of Boolean function.

Definition 5 (Integer encoding). *Integer encoding of a Boolean value x is defined as follows:*

$$x \rightarrow \begin{cases} 1 & \text{when } x = 0 \\ -1 & \text{when } x = 1. \end{cases} \tag{2}$$

Definition 6 (Walsh spectrum). *The Walsh-Hadamard spectrum of n variable Boolean function $f(x_1, x_2, ..., x_n)$ is represented as a vector of integer values S_f defined as:*

$$S_f = W(n) f^c. \tag{3}$$

where: f^c is a column truth-vector of function f in integer encoding.

Example 1. Let $f = (0, 0, 1, 1, 0, 1, 0, 1)^T$ be a truth-vector, then a vector of the form $f^c = (1, 1, -1, -1, 1, -1, 1, -1)^T$ represents integer encoded version of f.

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50426-7_18

In 3 variable domain Walsh-Hadamard transform $W(3)$ have the form:

$$W(3) = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{pmatrix}. \tag{4}$$

Walsh-Hadamard spectrum S_f of function f is $S_f = (0, 4, 4, 0, 0, -4, 4, 0)^T$.

The elements of spectral vector S_f are often called Walsh coefficients of function f in Hadamard order. In the Walsh domain of three variables, these coefficients are often designated as $S_f = (S_0, S_3, S_2, S_{23}, S_1, S_{13}, S_{12}, S_{123})^T$. These spectral coefficients represent the correlation of function values with input variables. The zero-order coefficient S_0 represents the difference of the number of occurrence of 0 and 1 values in a truth table column, for balanced functions S_0 is always 0. The first order coefficients $\{S_1, S_2, S_3\}$ represent the correlation of function values with the values of input variables x_1, x_2, x_3 respectively. The second-order coefficients $\{S_{12}, S_{13}, S_{23}\}$ are connected with the correlation of values of the function f and xor products: $x_1 \oplus x_2$, $x_1 \oplus x_3$, $x_2 \oplus x_3$. The third-order coefficient S_{123} represents the correlation of function f and $x_1 \oplus x_2 \oplus x_3$.

In the general case we can write properties of Walsh coefficients:

- all coefficients s have an integer value, $-2^n \leq s \leq 2^n$,
- a sum of absolute values of any two coefficients cannot exceed 2^n,
- S_0 - is connected to a constant part of the function, this coefficient is equal to 0 for balanced functions,
- S_i represents the correlation of the function in consideration with the value of variable x_i,
- S_{ij} represents the correlation of the function and xor product $x_i \oplus x_j$, where $i \neq j$ $1 \leq i, j \leq n$,
- $S_{ij...m}$ represents the correlation between the function and xor product $x_i \oplus x_j \oplus ... \oplus x_m$.

Definition 7 (Reverse Walsh transformation). *From Walsh-Hadamard spectrum S_f an original integer encoded Boolean function f^c can be obtained by reverse Walsh-Hadamard transform in the form:*

$$f^c = W^{-1}(n)S_f, \qquad \textit{where: } W^{-1}(n) = 2^{-n}W(n). \tag{5}$$

As reversible function can be treated as a vector of component functions, one can always derive Walsh transformation of a reversible function by application of Walsh transform to each of component functions independently.

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50426-7_18

Definition 8 (Walsh spectrum of reversible function). *The Walsh spectrum of $n*n$ reversible function F can be obtained by application of Walsh matrix to integer encoded function F^c*

$$S_F = W(n)F^c. \tag{6}$$

It should be noted that the spectral form $S_F = S_{f_1}, S_{f_2}, ...S_{f_n}$ of a reversible function $F = (f_1, f_2, ..., f_n)$ is a vector of spectral columns of component functions, i. e. $S_{f_i} = W(n)f_i^c$.

3.1 Spectral Invariant Operations

Definition 9 (Spectral invariant operations). *An operation on Boolean function f that preserve absolute values of Walsh spectral coefficients of the function is called a spectral invariant*

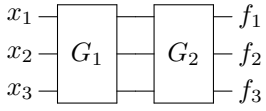
The set of spectral invariant operations have been used in Boolean logic for many years [9, 12, 13, 16]. The set of invariant operations is built from the following function transformations:

1. negation of the function - changes sign of all spectral coefficients,
2. negation of an input variable - changes the sign of spectral coefficients connected with this variable (for example when $x_2 \to \overline{x_2}$ then first-order coefficient change sign $S_2 \to -S_2$, and similarly appropriate second-order coefficients S_{12}, S_{13} and so on,
3. permutation of input variables - exchanges the coefficients connected with appropriate variables,
4. replacement of a variable x_i with $x_i \oplus x_j$ for $i \neq j$,
5. replacement of the function truth vector f with $f \oplus x_i$.

All invariant operations can be implemented by application of appropriate reversible gates, T1 gates implement operations 1 and 2 while using T2 gates one can build operations 4 and 5. The operations 3 can be implemented by a so-called swap gate, swap gate can be build from three T2 gates. The presented relations divide the set of all reversible gates into a set of spectral invariant operations (these gates have a quantum cost equal to 1) and the rest of reversible gates that can modify the spectrum of the function in consideration.

3.2 Walsh-Hadamard Spectrum and Reversible Gates Operation

As was shown above, the simplest reversible gates T1 and T2 are connected to spectral invariant operations. That means the rest of the reversible gates have to modify the spectrum of a Boolean function. Every reversible function can be represented as a permutation matrix, in particular, any reversible gate is connected to a permutation matrix. Suppose we have a reversible circuit of the form presented below.



ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50426-7_18

The function $F = (f_1, f_2, f_3)$ is represented by such a reversible gates G_1 and G_2, and can be written as:

$$F = G_2 G_1 I, \tag{7}$$

where I represent $3 * 3$ identity function and G_1, G_2 are permutation matrices representing reversible gates G_1, G_2. Applying Walsh-Hadamard transform to the resulting function F we have

$$WF = WG_2G_1I = WG_2W^{-1}WG_1W^{-1}WI, \tag{8}$$

$$S_F = WG_2G_1I = WG_2W^{-1}WG_1W^{-1}S_I = \widetilde{G_2}\widetilde{G_1}S_I, \tag{9}$$

where S_I denote Walsh spectral representative of identity function and $\widetilde{G_2}$ and $\widetilde{G_1}$ are Walsh-Hadamard representatives of reversible gates G_2, G_1 respectively. For any n the S_I is the simplest spectrum matrix $n * 2^n$, in each column, there is only one nonzero value. For $n = 3$ S_I have the form:

$$S_I = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 8 \\ 0 & 8 & 0 \\ 0 & 0 & 0 \\ 8 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}. \tag{10}$$

Taking into account permutation matrices of reversible gates T1, T2, T3 we can easily using matrices W and W^{-1} derive their representatives in the spectral domain, below representatives of selected gates in Walsh domain are presented:

$$T1(1) \rightarrow \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{pmatrix}, \quad T2(1;2) \rightarrow \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}, \tag{11}$$

$$T3(1,2;3) \rightarrow \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 & -\frac{1}{2} \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 & -\frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & -\frac{1}{2} & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & -\frac{1}{2} & 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 & \frac{1}{2} \end{pmatrix}. \tag{12}$$

As all T1 are diagonal and T2 a permutation matrices, their application to S_I will not modify absolute values in each column, these operations can only

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50426-7_18

change their positions. All other gates T3, T4, ..., Tn will change the values in the spectrum and very often change number or zeros in the spectral columns, the more zero values are in the spectrum the more linear the function is.

In Walsh domain, the functions are treated as a spectral matrix, all the possible input signals are taken into account, singular input is connected to a matrix row. The columns of the spectral matrix are connected to the spectrum of component functions. The action of the reversible gate on the actual state is represented as matrix state multiplication, which means all columns in the state matrix are treated independently, i.e. all component functions can be treated the same way.

4 Ant Colony Optimization

Ant colony optimization (ACO) is a biologically inspired meta-heuristic algorithm introduced by Dorigo in [5, 7]. The main idea of the algorithm is based on the social behavior of ants during a food search. The communication in the colony is based on pheromone residue that is left by each colony member when traveling between nest and food source. At first, ACO was introduced to solve traveling salesman problem later the method was applied to many other combinatorial problems [6, 8]

Usually, the ACO algorithm is defined on a graph that represents states in a search space, each edge in the graph represents actions (transitions between states) that can be taken during the walk of an artificial ant. Every action (edge) has an assigned value that represents the cost of the choice (very often the distance between the states).

The optimization procedure used in this paper is based on a colony of m artificial ants, in an iteration, every ant walks independently on a problem graph, at each node the ant has to choose an edge that leads to the nest state. When an ant reaches the end, either by reaching the final state node or the limit of steps in one iteration, pheromones residues on visited nodes are updated. The pheromone update rule and decision reasoning depend on ACO implementation.

At the start of the procedure, each node has been given an initial value τ_0. During the update procedure we allow the pheromones to evaporate in time.

$$\tau_{ij} \leftarrow \rho\tau_{ij}, \tag{13}$$

where $\rho \in [0, 1)$ is an evaporation parameter. The pheromone update procedure is done after evaporation, at the end of each iteration, every ant from the colony update pheromone by an additional deposit $\Delta\tau_{ij}^k$:

$$\Delta\tau_{ij}^k(t) = R/C^k(t), \tag{14}$$

where R is a constant and $C^k(t)$ represents the cost of the solution obtained by k-th ant in an iteration t, usually, C^k represents the sum of costs assigned to edges used in the solution. The final pheromone update procedure, that takes

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50426-7_18

into account evaporation and new deposits have the form:

$$\tau_{ij}(t+1) = \rho\tau_{ij}(t) + \sum_{k} \Delta\tau_{ij}^{k}(t). \tag{15}$$

When ant moves over the graph the choice of an outgoing edge can be taken either randomly or based on actual pheromones residue and partial heuristic reasoning. In the procedure, a parameter ζ, the probability of pure random choice at each graph node, is used. In the case of pheromone-based choice the decision is based on weighted probability:

$$p_{ij}(t) = \frac{\tau_{ij}(t)^{\alpha}\eta_{ij}^{\beta}}{\sum_{i,j}\tau_{ij}(t)^{\alpha}\eta_{ij}^{\beta}}, \tag{16}$$

where $p_{ij}(t)$ is the probability of choosing j edge when an ant is in node i in iteration t, $\tau_{ij}(t)$ represents the actual pheromone deposit in node i attached to edge j at the moment t. The parameter η_{ij} represents heuristic reasoning, this part always depends on the problem to be solved. Two additional parameters α, β describe the relative importance of the pheromone and heuristic factors.

5 Algorithm Implementation

In this paper ACO is used to optimize the process of reversible function synthesis, moreover, all the reversible functions are represented with the use of the Walsh-Hadamard spectrum. The procedure starts with an initial function to be synthesized. Each ant will try to convert the function spectrum into a spectrum S_I by application of reversible gates. The solution is a sequence of gates that represents a function in consideration.

In the proposed ant colony optimization procedure state (node of the graph) is connected to the spectrum of a reversible function, while the actions (edges of the graph) are the gates that can be added to the solution. Each edge has an attached value that represents the quantum cost of the appropriate gate.

5.1 Heuristic Reasoning for Designed ACO

As was mentioned before in the process of decision making the heuristic knowledge about the word of reversible functions is taken into account (16). The target state S_I has only one nonzero value, that means during synthesis procedure we should maximize the number of zeros of the actual state.

Suppose we have two states F_1, F_2 represented by spectral matrices S_{F_a} and S_{F_b}. Additionally, there is the reversible gate G_x that transforms one state into another one, i. e. $S_{F_a} = \widetilde{G_x} S_{F_b}$. The heuristic factor for edge G_x connected to the state F_a will have the form:

$$\eta_{ax} = 1 + \frac{\texttt{count\_zeros}(S_{F_b}) - \texttt{count\_zeros}(S_{F_a})}{\texttt{min}\left(\texttt{count\_zeros}(S_{F_b}), \texttt{count\_zeros}(S_{F_a})\right)} \tag{17}$$

$$\pm \frac{\sigma}{\texttt{qc}(G_x)}, \tag{18}$$

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50426-7_18

where σ is a scaling parameter, `count_zeros`(.) represent the function that returns the number of zeros in the argument, `min`(.,.) is the function that returns the minimum value from given arguments, `qc(.)` represents the quantum cost of a reversible gate. The sign $\pm$ corresponds to the sign of $\texttt{count\_zeros}(S_{F_b}) - \texttt{count\_zeros}(S_{F_a})$.

The form of the heuristic factor (17) represents the main parameters that have to be taken into account during the synthesis procedure. The main part $\frac{\texttt{count\_zeros}(S_{F_b})-\texttt{count\_zeros}(S_{F_a})}{\texttt{min}(\texttt{count\_zeros}(S_{F_b}),\texttt{count\_zeros}(S_{F_a}))}$ depends on a change of the number of zeros in two connected states S_{F_a} and S_{F_b}, if the application of the gate increase the number of zeros the factor has positive value when number of zero decreases the factor is negative and so decrease the probability of choosing the edge. The second element of the heuristic factor $\pm\frac{\sigma}{\texttt{qc}(G_x)}$ distinguishes the gates that generate state with the same number of zeros in the spectrum but differs in quantum cost.

5.2 Pheromone Update Procedure

In the pheromone update procedure, the deposit from k-th ant given by equation (14), the sum of quantum costs of all the gates in the solution found by an ant is used for assessment of the cost of the solution C^k.

5.3 Additional Parameters of ACO Used in the Implementation

In the implementation of the ACO algorithm, the values of parameters that were used are presented in Table 2.

Table 2: Parameters used in implemented ACO algorithm

Name	Short description	Value
α	The parameter connected to the influence of pheromone in decision making (16)	1
β	The parameter connected to the influence of heuristic in decision making (16)	1
ζ	The probability of random choice at every node	0.5
m	The size of the colony	30
R	The scale factor used for deposit (14)	1
ρ	The pheromone evaporation factor (13)	0.8
σ	The scale factor in heuristic rule (17)	1

5.4 Creation of ACO System

The ACO is based on the graph that represents the search space. It has to be mentioned that the graph in consideration is an enormous one, for n variables,

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50426-7_18

there are $2^n!$ reversible functions. It is impossible to build and keep in memory all possible functions as nodes and transition gates edges. For that reason, our search space will be built during the optimization process, whenever an ant visits a new node, important node information will be created, that means assign all edges, generate all heuristic factors, initialize pheromone deposit, etc. This lazy node initialization allows us to save memory and speed up the initialization phase. In the starting initialization phase, only the target state (node connected to S_I) and a state connected to a function f in consideration are created. The node connected to f takes the role of the nest, it means at the beginning of a new iteration every ant starts from this node.

During the first tests of the algorithm, it was noticed that the ACO behaves better (gives better results) when more than one final state was given. For that reason, in the initialization phase, more than one target state was created. For every gate a state that can be obtained from state S_I by application of a single gate was initialized, moreover, this states had a very simple assignment of the pheromones for edges: 1 for the edge that leads to S_I and 0 to others. This creates an additional set of states that directly leads to the final state. This procedure could be extended to next states with the distance of two or three gates from S_I, it has to be noted that while for $n = 4$ there are only 32 nearest neighbors of S_I the number of neighbors in a distance of 3 gates can be estimated to $5000 - 10000$ and it not seem reasonable to initialize them all.

6 Numerical Results

The proposed algorithm has been implemented in python, and run to synthesize selected benchmark functions. At first ACO algorithm was used to synthesize one of functions used in publication [17]: $f = [2, 9, 7, 13, 10, 4, 2, 14, 3, 0, 12, 6, 8, 15, 11, 15]$. On that function, the impact of algorithm parameter values was analyzed and their final values selected (see Table 2).

In the literature, authors use many different functions as well as different cost measures to test developed synthesis algorithms. Therefore, the algorithm was tested against two sets of functions: one the set of known benchmarks and results obtained via heuristic algorithms implemented in revkit tool [25] (Table 3), the other set of functions taken from [17] in order to compare two ACO based approaches (Table 4). The results from Table 3 shows that the results obtained are not always optimal, however, most of them are near-optimal.

In the article that uses the ACO approach [17], we can find results of synthesis that takes into account the number of gates. In order to be able to compare both approaches, the cost measure was changed from quantum cost (QC) to the number of gates in sequence (GC). For the cases presented in Table 4 the cost of all gates was set to 1 instead of quantum cost.

The results are shown in Table 4 show that the results obtained are comparable, in more difficult tasks often better to those obtained in [17]. This could be the result of function representation used, as the Walsh spectrum contains more

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50426-7_18

Table 3: Results of the presented ACO algorithm for selected benchmark functions compared with circuits obtained with revkit tool [25].

Benchmark function	ACO Walsh		Known Solutions		
	GC	**QC**	GC	**QC**	
4_49	35	88	36	84	[29]
4b_15g_1	27	67	34	90	[29]
4b_15g_2	32	91	35	83	[29]
4b_15g_3	25	81	32	96	[29]
4b_15g_4	26	80	35	91	[29]
4b_15g_5	35	102	34	80	[29]
aj-e11	25	53	34	90	[29]
App2.2	22	83	26	86	[29]
decode42	25	53	34	90	[29]
dmasl	24	68	30	78	[29]
gyang	30	96	16	76	[29]
hwb4	20	40	21	37	[29]
mini-alu	13	45	12	68	[29]
mod10_171	22	72	13	65	[29]
msaee	26	64	38	98	[29]
nth_prime4	18	60	26	70	[29]
oc5	31	127	26	82	[29]
oc6	30	66	36	88	[29]
oc7	20	54	32	92	[29]
oc8	24	54	38	98	[29]
hwb5	34	81	33	71	[27]
hwb6	48	105	47	107	[24]
nth_prime6_inc	59	281	57	485	[27]

Table 4: Results of the presented ACO algorithm compared to the results from [17] when minimal gate count is taken as the goal of the synthesis.

Function	Permutation based ACO [17]	Walsh based ACO
[1, 0, 3, 2, 5, 7, 4, 6]	3	3
[7, 0, 1, 2, 3, 4, 5, 6]	3	3
[0, 1, 2, 3, 4, 6, 5, 7]	3	3
[0, 1, 2, 4, 3, 5, 6, 7]	4	3
[0, 1, 2, 3, 4, 5, 6, 8, 7, 9, 10, 11, 12, 13, 14, 15]	7	8
[1, 2, 3, 4, 5, 6, 7, 0]	3	3
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 0]	3	3
[0, 7, 6, 9, 4, 11, 10, 13, 8, 15, 14, 1, 12, 3, 2, 5]	4	4
[3, 6, 2, 5, 7, 1, 0, 4]	6	6
[1, 2, 7, 5, 6, 3, 0, 4]	6	6
[4, 3, 0, 2, 7, 5, 6, 1]	5	5
[7, 5, 2, 4, 6, 1, 0, 3]	5	5
[6, 2, 14, 13, 3, 11, 10, 7, 0, 5, 8, 1, 15, 12, 4, 9]	11	10
[2, 9, 7, 13, 10, 4, 2, 14 3, 0, 12, 6, 8, 15, 11, 15]	11	11
[6, 4, 11, 0, 9, 8, 12, 2, 15, 5, 3, 7, 10, 13, 14, 1]	13	12
[13, 1, 14, 0, 9, 2, 15, 6, 12, 8, 11, 3, 4, 5, 7, 10]	10	9

global information on the function and heuristic factor of ant decision policy is probably better suited to the synthesis task.

7 Conclusions

In the paper, a new approach for meta-heuristic reversible synthesis is presented. The most important change is connected with different function representation used. The Walsh-Hadamard spectrum of a function contains some global properties of the function, i.e. the correlation of the function values with input variables. This property allows the algorithm to create a choosing policy based on the linearity of the function, the number of zeros in spectral representation of the function in consideration. Additionally knowing that some of the simplest reversible gates preserve absolute values of the spectrum the artificial ants in the ACO algorithm more often use high-cost gate only when the gate simplifies the

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50426-7_18

function. The results of the presented algorithm were compared to best-known solutions, this comparison shows that even though the results obtained didn't always reach global optimal solutions, they were near the optimal ones. It has to be noted that the global optimal reversible circuits are known only for functions with the input variable numbers up to 4, for the functions with a higher number of inputs the global optimal solutions are not known. Therefore all methods that give a near-optimal solution for a high number of variables are important.

7.1 Future Areas of Research

The results obtained are promising, the extensions of the method presented are considered. As was mentioned all columns of component functions are treated independently, it is possible to store in the graph only component function states while all the decisions would be taken with the use of global knowledge, the final decision would be the sum of factors for each component function. This could lead to an exchange of information between similar functions that share the same component function spectral column. Additionally, it is possible to use two-directional synthesis, i.e. one nest can be placed in target node and search for a given function, while the other nest works the same way as presented, the pheromones could be exchanged between these two colonies. This could lead to better exploration and exploitation of search space in consideration.

References

1. Amy, M., Maslov, D., Mosca, M., Roetteler, M.: A meet-in-the-middle algorithm for fast synthesis of depth-optimal quantum circuits. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **32**(6), 818–830 (June 2013)
2. Barenco, A., Bennett, C.H., Cleve, R., DiVincenzo, D.P., Margolus, N., Shor, P., Sleator, T., Smolin, J.A., Weinfurter, H.: Elementary gates for quantum computation. Phys. Rev. A **52**, 3457–3467 (Nov 1995)
3. Conte, T.M., DeBenedictis, E.P., Gargini, P.A., Track, E.: Rebooting computing: The road ahead. Computer **50**(1), 20–29 (Jan 2017)
4. Datta, K., Sengupta, I., Rahaman, H.: Particle swarm optimization based reversible circuit synthesis using mixed control toffoli gates. Journal of Low Power Electronics **9**(3), 363–372 (2013)
5. Dorigo, M.: Optimization, learning and natural algorithms. Ph. D. Thesis, Politecnico di Milano, Italy (1992)
6. Dorigo, M., Blum, C.: Ant colony optimization theory: A survey. Theoretical Computer Science **344**(2), 243 – 278 (2005)
7. Dorigo, M., Maniezzo, V., Colorni, A.: Ant system: optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) **26**(1), 29–41 (Feb 1996)
8. Dorigo, M., Stützle, T.: Ant Colony Optimization. A Bradford book, BRADFORD BOOK (2004)
9. Edwards, C.R.: The application of the rademacher–walsh transform to boolean function classification and threshold logic synthesis. IEEE Transactions on Computers **100**(1), 48–62 (1975)

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50426-7_18

10. Frank, M.P.: Throwing computing into reverse. IEEE Spectrum **54**(9), 32–37 (September 2017)
11. Hadjam, F.Z., Moraga, C.: Rimep2: Evolutionary design of reversible digital circuits. ACM Journal on Emerging Technologies in Computing Systems (JETC) **11**(3), 27 (2014)
12. Hurst, S.L.: The logical processing of digital signals, crane russak & company. Inc., New York, Edward Arnold, London (1978)
13. Karpovsky, M.G., Stanković, R.S., Astola, J.T.: Spectral logic and its applications for the design of digital devices. John Wiley & Sons (2008)
14. Kerntopf, P., Perkowski, M., Podlaski, K.: Synthesis of reversible circuits: A view on the state-of-the-art. In: Proceedings of the 12th IEEE Conference on Nanotechnology (IEEE-NANO). pp. 1–6. IEEE (2012)
15. Landauer, R.: Irreversibility and heat generation in the computing process. IBM Journal of Research and Development **5**(3), 183–191 (1961)
16. Lechner, R.J.: A transform approach to logic design. IEEE Transactions on Computers **100**(7), 627–640 (1970)
17. Li, M., Zheng, Y., Hsiao, M.S., Huang, C.: Reversible logic synthesis through ant colony optimization. In: Proceedings of the Design, Automation & Test in Europe Conference & Exhibition (DATE). pp. 307–310. IEEE (2010)
18. Manna, P., Kole, D.K., Rahaman, H., Das, D.K., Bhattacharya, B.B.: Reversible logic circuit synthesis using genetic algorithm and particle swarm optimization. In: Proceedings of the International Symposium on Electronic System Design (ISED). pp. 246–250. IEEE (2012)
19. Maslov, D., Dueck, G.W., Miller, D.M.: Techniques for the synthesis of reversible toffoli networks. ACM Trans. Des. Autom. Electron. Syst. **12**(4) (Sep 2007)
20. Miller, D.M., Maslov, D., Dueck, G.W.: A transformation based algorithm for reversible logic synthesis. In: Proceedings of the 40th annual Design Automation Conference. pp. 318–323. ACM (2003)
21. Miller, D.M., Soeken, M., Drechsler, R.: Mapping ncv circuits to optimized clifford+t circuits. In: Yamashita, S., Minato, S.i. (eds.) Reversible Computation. pp. 163–175. Springer International Publishing, Cham (2014)
22. Nielsen, M.A., Chuang, I.L.: Quantum Computation and Quantum Information. Cambridge University Press (2010)
23. Saeedi, M., Markov, I.: Synthesis and optimization of reversible circuits – a survey. ACM Computing Surveys **45**(2), 21:1–21:34 (2013)
24. Saeedi, M., Zamani, M.S., Sedighi, M., Sasanian, Z.: Reversible circuit synthesis using a cycle-based approach. J. Emerg. Technol. Comput. Syst. **6**(4), 1–26 (2010)
25. Soeken, M., Frehse, S., Wille, R., Drechsler, R.: Revkit: An open source toolkit for the design of reversible circuits. In: Reversible Computation - Third International Workshop, RC 2011, Gent, Belgium, July 4-5, 2011. pp. 64–76 (2011)
26. Stanković, R.S., Astola, J.T.: Spectral interpretation of decision diagrams. Springer (2003)
27. Wang, X., Jiao, L., Li, Y., Qi, Y., Wu, J.: A variable-length chromosome evolutionary algorithm for reversible circuit synthesis. Journal of Multiple-Valued Logic and Soft Computing **25**(6), 643–671 (2015)
28. Wille, R., Drechsler, R.: Bdd-based synthesis of reversible logic. Int. J. of Applied Metaheuristic Computing **1**(4), 25–41 (2010)
29. Wille, R., Große, D., Teuber, L., Dueck, G.W., Drechsler, R.: RevLib: An online resource for reversible functions and reversible circuits. In: Int'l Symp. on Multi-Valued Logic. pp. 220–225 (2008), RevLib is available at http://www.revlib.org

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50426-7_18