Reversible Circuit Rewriting with Simulated Annealing

Nabila Abdessaied* Mathias Soeken*† Gerhard W. Dueck ‡ Rolf Drechsler*†
*Cyber-Physical Systems, DFKI GmbH, Bremen, Germany
†Department of Mathematics and Computer Science, University of Bremen, Germany
‡Faculty of Computer Science, University of New Brunswick, Canada
{nabila,msoeken,drechsle}@informatik.uni-bremen.de, gdueck@unb.ca

***Abstract*—This paper presents a rule based approach to optimize the quantum cost of reversible circuits using circuit rewriting rules that handle positive and negative controls. Since incremental optimization cannot guarantee optimality, we consider the application of simulated annealing to find further sub-circuits that could be replaced with smaller ones.**

Experimental evaluations show that simulated annealing not only can significantly improve the quality of reversible circuits but also is more efficient than a comparable greedy approach. Using the rewriting rules combined with the proposed method quantum cost reductions by up to 80% can be achieved.

I. Introduction

Synthesis describes techniques of finding a reversible circuit that realizes a given reversible function. Typically, synthesis approaches are not aware of the technology in which the circuit is applied. In order to optimize synthesis results with respect to some target technology, post-synthesis optimization approaches are used to reduce the circuits with respect to a given cost, e.g., transistor costs in CMOS architectures or quantum costs in quantum computing architectures.

Due to the inherent complexity of reversible circuits, usually local optimization strategies are implemented, i.e., sub-circuits are analyzed for possible reductions. The most employed optimization approaches can be categorized into *rule-based optimization* and *template matching*.

Rule-based optimization (see, e.g., [1]–[3]) suggests a specific set of sub-circuits together with cheaper replacements. Rule-based approaches are typically motivated based on some synthesis approaches and exploit often reoccurring circuit structures. The approaches are not complete, i.e., they cannot guarantee an optimal circuit after optimization. However, the approaches are greedy meaning that optimization is only applied if a reduction can be achieved. Consequently, they cannot escape local optima.

Template matching approaches (see, e.g., [4])are more powerful than rule-based approaches. A template is a generic circuit that realizes the identity function. Generic means that *one* template *represents* a (theoretically infinite) *set* of identity circuits. If a sub-circuit matches one part of an instance of a template it can always be replaced with the inverse of the remaining part, since they represent the same function. Optimization with templates is also greedy, however, it is still unknown whether template matching is complete.

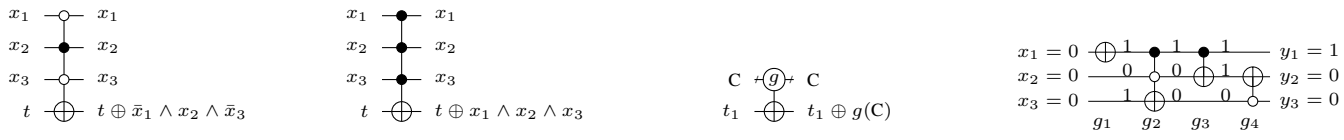
978-1-4673-9140-5/15/$31.00 2015 IEEE

In order to find suitable sub-circuits to apply a rule or a template, *moving rules* [5] have been proposed that allow gate movement without changing the function. These moving rules change the order of gates but not the gates themselves. Moving rules are not complete, i.e., one cannot necessary obtain all function-preserving permutation of gates in the circuit by repeatedly applying the moving rules. Since the rule-based and template matching approaches are incremental greedy algorithms, they can achieve further improvements on synthesized circuits but cannot guarantee optimality. In fact, these approaches can guarantee only local optimization but not global optimization.

Recently, *circuit rewriting* [6] has been proposed that exploits negative control to rewriting sub-circuits while preserving the function. Circuit rewriting is based on a small set of elementary rules which can be extended to more complicated rules. It is conjectured that circuit rewriting is complete based on a very small set of rules, i.e., one can rewrite a circuit into all other circuits that represent the same functionality. Circuit rewriting is not greedy, i.e., intermediate steps may increase the cost. In fact, no guided approach for circuit optimization using the rewrite rules has been proposed so far.

In this paper, we present rule based optimization approaches using the rewrite rules from [6] instead of the moving rules [5]. Two quantum cost optimization approaches are presented. The first method aims to reduce the cost by applying a greedy approach, whereas the second method is based on simulated annealing. The application of simulated annealing can attain not only local optimum but also global optimum [7], hence it can find further sub-circuits to be replaced with smaller ones. The rewrite rules support this freedom by allowing a higher flexibility in gate movement. As confirmed by an experimental evaluation, improvements on quantum cost of up to 17% in the first case and up to 30% in the latter case can be observed. This clearly demonstrates that the simulated annealing approach outperforms the greedy approach on optimizing reversible circuits.

The remainder of the paper is structured as follows: first the basics on reversible circuits are recaptured in Sect. II. The next



(a) MPMCT gate (b) MCT gate (c) MPMCT gate, g is product term (d) Reversible circuit

Fig. 1: Reversible circuitry

section outlines the simulated annealing approach. Section IV introduces the rewriting rules and Sect. V gives a detailed description of the implementation of the presented approach, and experimental results are evaluated and interpreted in Sect. VI. The paper is concluded in Sect. VII.

II. Background

To keep this paper self-contained, this section briefly introduces the basics on reversible logic, quantum circuits, and their cost metrics.

A. Reversible Logic

Let $\mathbb{B} = \{0, 1\}$ denote the *Boolean values*. Then we refer to $\mathcal{B}_{n,m} = \{f \mid f \colon \mathbb{B}^n \to \mathbb{B}^m\}$ as the set of all *Boolean multiple-output functions* with n inputs and m outputs.

Definition 1 (Reversible function): A function $f \in \mathcal{B}_{n,m}$ is called *reversible* if f is bijective, i.e., if each input pattern is uniquely mapped to an output pattern, and vice versa. Otherwise, it is called *irreversible*. Clearly, if f is reversible, then $n = m$.

Reversible functions are realized by reversible circuits that consist of at least n lines and are constructed as cascades of reversible gates from some gate library. The most common gate library consists of multiple control Toffoli gates [8].

Definition 2 (Toffoli gate): Given a set of variables $X = \{x_1, \ldots, x_n\}$, a *mixed-polarity multiple-control Toffoli (MPMCT)* gate $\mathrm{T}(C, t)$ has *control lines* $C = \{x_{j_1}, x_{j_2}, \ldots, x_{j_l}\} \subset X$ and a *target line* $t \in X \setminus C$. The gate maps $t \mapsto t \oplus g(x_{j_1}, x_{j_2}, \ldots, x_{j_l})$ where g is defined as:

$$g : (x_{j_1}, x_{j_2}, \cdots, x_{j_l}) \mapsto (x_{j_1}^{p_1} \wedge x_{j_2}^{p_2} \wedge \cdots \wedge x_{j_l}^{p_l})$$

with each *literal* $x_{j_i}^{p_i}$ is either a propositional variable $x_{j_i}^1 = x$ or its negation $x_{j_i}^0 = \bar{x}$. All remaining other lines are passed through unaltered. *Multiple-control Toffoli gates (MCT)* are a subset from MPMCT gates in which the product terms in h can only consist of positive literals.

Example 1: Figure 1(a) shows a Toffoli gate with mixed polarity control lines, Fig. 1(b) shows a Toffoli gate with n positive controls. The control lines are either denoted by solid black circles to indicate positive controls, white circles to indicate negated controls or represented by a one product terms Boolean function g as depicted in Fig. 1(c). The target line is denoted by $\bigoplus$. Figure 1(d) shows different Toffoli gates in a cascade forming a reversible circuit. The annotated values demonstrate the computation of the gate for a given input assignment.

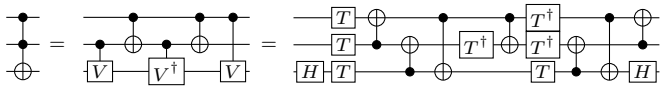


Fig. 2: Mapping a 2-controlled Toffoli gate to NCV and Clifford+T quantum gates

B. Cost Metrics

The quantum cost of a reversible or a quantum circuit is varying with respect to the quantum library used in the technology mapping. When using the NCV library, then the quantum cost is called the *NCV-cost* while it is called T-*depth* when the Clifford+T library is used. The motivation for that cost measure originates from the fact that the T gate is significantly larger compared to the other gates in the circuit.

The common gate library NCV and the Clifford+T gate libraries are composed of the elementary gate set {NOT, CNOT, V, V†} and {NOT, CNOT, H, Z, S, S†, T, T†}, respectively. The two libraries are universal for quantum computation, however only the gates of the Clifford+T library can be implemented in a fault-tolerant way.

Definition 3 (NCV-Cost): The *NCV-cost* is the total number of elementary gates used in a quantum circuit.

Definition 4 (T-depth): The T-*depth* is the number of T-stages in a quantum circuit where each stage consists of one or more T or $T^\dagger$ gates that can be performed concurrently on separate qubits. The total number of incorporated T or $T^\dagger$ gates in the whole circuit is denoted by T-*count*.

Example 2: Consider a Toffoli gate with two control lines as shown in Fig. 2. A functionally equivalent realization in terms of quantum gates using the NCV library is depicted in the second network. The NCV-cost of this circuit is 5 since there are 5 elementary gates in the circuit. The third cascade represent the quantum realization of a Toffoli gate using the Clifford+T library. It has a T-*count* of 7 and T-*depth* of 3. The reversible circuit depicted in Fig. 1(d) has an NCV-cost of 9 and a T-*depth* of 3.

Based on [9], the following table summarizes the quantum cost for MCT gates where c denotes the number of controls:

Number of controls	< 2	2	3	4	≥ 5
NCV-cost	1	5	20	50	$40(c-3)$
T**-depth**	0	3	12	30	$24(c-3)$

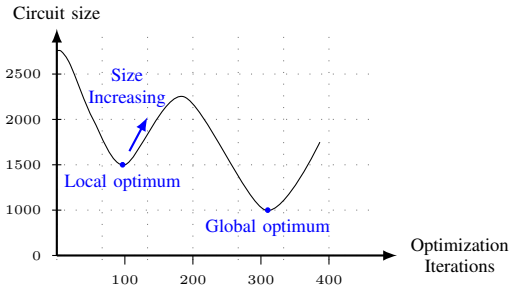


Fig. 3: Greedy heuristics via simulated annealing algorithm

III. Simulated Annealing

Assume that S is a solution to a given problem. Let M be a "move" that can be performed on S. M will have an effect on the "cost" of S. The objective of optimization is to to repeatedly perform "moves" to S such that the cost of S is reduced. The greedy algorithm is a simple heuristic optimization technique. With the problem at hand, it would only perform moves that lower the cost. The obvious problem with such an approach is that it can get stuck in a local optimum (see Fig. 3). Kirkpatrick et. al. [7] suggest an optimization technique called *simulated annealing* that is based on statistical mechanics.

The idea of simulated annealing is simple. Randomly select moves (or transformations) on a given solution. Each move will have an effect on the cost of the solution. The moves can be cost-increasing, cost-decreasing, or cost-neutral. Cost-decreasing moves are always accepted. Simulated annealing uses the concept of *temperature* (it plays a significant role in statistical mechanics) to deal with cost-increasing moves. Initially the temperature is high. Cost increasing moves are accepted with a probability that depends on the temperature. Initially, many cost-increasing moves are accepted. As the temperature decreases, fewer and fewer moves are accepted. Finally, the procedure stops when the system reaches a stable state, that is, no moves are accepted.

The process of reversible circuit optimization is well suited for simulated annealing. Gates can be moved within the circuit. The move may be associated with an increased cost of the circuit. On the other hand, a gate may be moved adjacent to an other gate that is identical. In this case both gates can be removed since Toffoli gates are self inverse, hence the cost will be reduced. In dealing with reversible circuits, the NCV-cost or the T-depth may be considered as cost metric during the simulated annealing process.

Simulated annealing has been used in the synthesis of reversible circuits, e.g, in [10], the authors have presented a simulated Annealing based Quine-McCluskey approach to synthesize a reversible circuit. Also, the synthesis algorithm presented in [11] have used simulated annealing to transform ESOP cubes.

IV. Rewriting Rules

Most of the existing synthesis approaches for reversible functions do not generate optimal circuit realizations with respect to quantum cost. Therefore optimization approaches are applied to reduce the quantum cost. These post synthesis techniques attempt to apply reduction rules by deleting identical gates or replacing cascades of gates with smaller ones [4]. To do so, the gates are rearranged to match the reduction rules. The moving rules were originally defined in [5] as the following property and illustrated in Fig. 4(a).

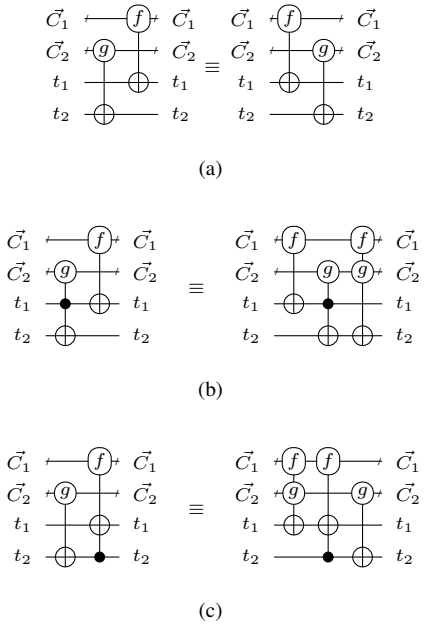


Fig. 4: Rewriting rules

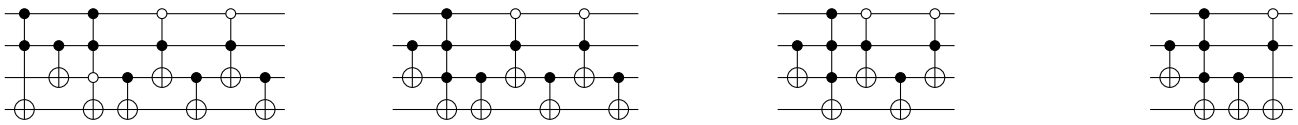
Moving rule. Two adjacent gates can be interchanged if and only if the target for each gate is not a control for the other gate, i.e., in a reversible circuit, gate $T_1(C_1, t_1)$ can be interchanged with gate $T_2(C_2, t_2)$ if and only if $t_2 \notin C_1$ and $t_1 \notin C_2$.

This moving rule is very restrictive, therefore, the conventional moving rule was extended as described in [12] allowing further moving abilities for each gate in the circuit. The extended moving rule is defined in the following property.

Extended moving rule. A gate can be moved from one end of a cascade of gates to the other end if its controls are on lines that are invariant with respect to the cascade and its target is on a non-controlling line.

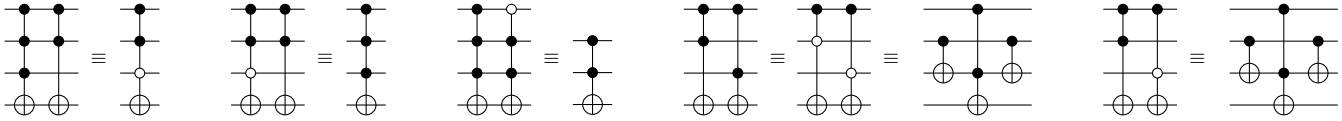
To identify whether a line is invariant, an algorithm called *line labelling procedure* (LLP) labels the line segments in a circuit. A label is an assignment of numbers to each line after a gate. If the label is the same at the beginning and the end of the segment, then the line is invariant.

Rewriting rules. However, the above moving rules are restraining the movement of gates into a circuit. On the other hand, the moving rules introduced in [6] are general for both MCT and MPMCT cascades and have more freedom for gate rearrangement. Based on the rules, we extract three scenarios for moving a gate from one position to another as shown in Fig. 4.



(a) Original realization (b) Moving rules (c) Extended moving rules (d) Rewriting rules

Fig. 5: Circuit optimization wrt. different moving rules



(a) (b) (c) (d) (e)

Fig. 6: Applied reductions

Example 3: Consider the reversible circuit depicted in Fig. 5(a). Its NCV-cost is 39 while its T-depth is 21. When this circuit is optimized using the conventional moving rules, then the gate in position 1 could be moved to position 2. Hence the gates in position 2 and 3 could be merged into one gate as shown in the circuit depicted in Fig. 5(b). No more reductions are possible. The obtained circuit has an NCV-cost of 34 and a T-depth of 18. But if we consider the extended moving rules, one can move from the circuit in Fig. 5(b) also the gate in position 3 to the position 7 since its control line is invariant and its target line does not contain any controls between position 3 and 7. Thus the moved gate would be removed with its neighbour because they form an identity circuit. The obtained circuit is shown in Fig. 5(c) and has NCV-cost of 32 and a T-depth of 18. Now if we use the rewriting rules, one can move the gate in position 3 from the circuit in Fig. 5(c) to position 4 by applying the rule shown in Fig. 4(c). As a result the moved gate form an identity with gate in position 5 and get both removed. The resulting circuit is presented in Fig. 5(d). It has an NCV-cost of 27 and T-depth of 15.

In the following, we will consider the rewriting rules for optimizing reversible circuits and show through the experimental results its efficiency in reducing the quantum cost of reversible circuits.

V. Algorithms

To show the advantages of applying simulated annealing to reduce the quantum cost for reversible circuit, we compare it with the well known approach for optimizing reversible circuits based on exhaustive search. To do so, we introduce in this section as a first step a greedy approach combined with the rewriting rules. Then, as a second step, we define the simulated annealing approach. Both approaches are applied to gate cascade with a common target. Note that gates can be rearranged to create such a cascade using rewriting rules. Each MPMCT optimization procedure finds possible reductions in the circuit by moving gates across the circuit and making them adjacent. The gates may either be cancelled when they are identical or may be reduced to a less expensive cascade using the five different rules sketched in Fig. 6.

A. Greedy Approach

Given is a reversible circuit $G = T_1(C_1, t_1) \dots T_k(C_k, t_k)$ with k gates over variables $x_1, \dots, x_n$. This algorithm optimizes the circuit by applying a greedy approach.

For each gate, this technique searches over the circuit for a gate that has the same target. A found gate can be merged with the requested gate only when the rewriting process reduces the quantum cost of the targeted reversible circuit. After a reduction is applied, the optimization restarts the same process from the first gate of the circuit.

B. Simulated Annealing Approach

Given is a reversible circuit $G = T_1(c_1, t_1) \dots T_k(c_k, t_k)$ with size k over variables $x_1, \dots, x_n$. This algorithm optimizes the circuit by applying simulated annealing. For the computation, we are making use of the variables k,T, frozen, l, and Δ_{cost} to denote the size of the circuit, the used temperature, the stopping criterion, the number of generated perturbation, and the cost variable, respectively. The remaining variables (i, j, and optimized) are used to control the algorithm loops.

The algorithm is listed in Algorithm 1. We have chosen the initial temperature and the number of perturbation as factors of the number of gates in the initial circuit while the stopping criterion is set to 0 regardless of the size of the circuit and should not exceed the value 5 (see lines 2, 3, and 4). The algorithm generates, for a predetermined number of times l (line 9), two different positions of gates (line 10 and 11) denoted by loc and pos. Then, it calculates the rewriting cost for rearranging them together (line 12). If the cost is decreased then the solution is accepted, i.e, the gates are merged together (line 14). Otherwise the solution is accepted with a certain probability (line 18). After each loop the temperature T is

Algorithm 1: Simulated annealing

Input: Reversible circuit G
Output: Optimized circuit G'

```
1  k ← Size(G)
2  T ← 10k
3  frozen ← 0
4  l ← 100k
5  while frozen > 5 do
6      j ← i + 1
7      optimized ← False
8      i ← 0
9      for i = 0 to l do
10         pos ← Random(0,k)
11         loc ← Random(0,k)
12         Δ_cost ← RewriteCost(G, pos,loc)
13         if Δ_cost < 0 then
14             RewriteMerge(G, pos, loc)
15             optimized ← True
16         else
17             q ← Random(0,1)
18             if q < e^(−Δ_cost/T) then
19                 RewriteMerge(G, pos, loc)
20                 optimized ← True
21             end
22         end
23     end
24     T ← 0.8T
25     if optimized then
26         frozen ← 0
27     else
28         frozen ← frozen + 1
29     end
30 end
```

decreased (line 24) and the stopping criterion frozen is reset to 0 when the circuit has changed (line 26) otherwise the variable is incremented (line 28). This process is repeated until the stopping criterion reaches the value 5.

VI. Experimental Results

In this work, we proposed rule based approaches for optimizing MPMCT circuits using rewriting rules. The first is using a greedy search algorithm while the second is using a simulated annealing algorithm. The proposed ideas described above in Sect. V have been implemented in the open source toolkit *RevKit* [13]. The experimental evaluation has been carried out on an Intel Core i5 Processor with 4 GB of main memory using the benchmarks taken from [14], [15]. We have observed that our approaches lead to reversible circuits with smaller NCV-cost and T-depth.

The experimental results presented graphically in the plot depicted in Fig. 7 show the NCV-cost of optimized reversible circuits with respect to different optimization algorithms. The values of x-axis and the y-axis denote the name of

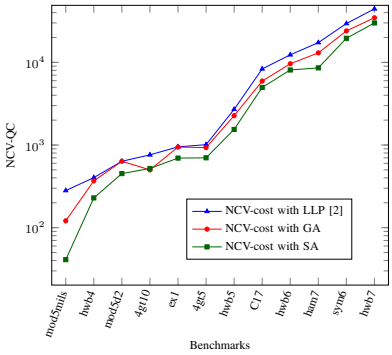


Fig. 7: Evaluation of Optimization approaches wrt NCV-cost

Benchmarks and the NCV-cost, respectively. The plot contains three different scenarios: the NCV-cost of optimized reversible circuits based on LLP approach [2] (in blue), the NCV-cost of circuits optimized with greedy approach (in red), and the NCV-cost of optimized circuits based on simulated annealing (in green). One can clearly see that the greedy approach outperforms the LLP approach [2] based on extended moving rules. While the simulated annealing approach produces circuits with the smallest NCV-cost in comparison with the other approaches. In the rest of the paper we consider only the results of the greedy and simulated annealing approaches.

Our results capture the following values: (1) the results of the greedy optimization approach (GA), (2) the results of optimized reversible circuits using simulated annealing approach (SA), (3) improved quality with respect to NCV-costs and T-depths of resulting circuits from the greedy approached compared to the original benchmarks (GA/OB), (4) the quality of optimized circuits using simulated annealing compared to the original circuits (SA/OB), and (5) the improvement of the simulated annealing approach over the greedy approach (SA/GA).

In Table I, the results are summarized as follow: for each benchmark we show the name (ID), the number of lines (L), the NCV-cost (NCV), and the T-depth (TD). In addition to these metrics, for each approach, we add the required runtime in seconds (Time). The NCV-cost and the T-depth reduction and improvement are provided in the columns denoted by ΔNCV, I_{NCV}, ΔTD, and I_{TD}, respectively.

The results from the simulated annealing approach are given in third, fifth, and sixth columns of Table I. Our proposed second approach leads to significant *T-depth* and NCV-cost reductions. Over all circuits, reductions up to 184589 NCV gates can be obtained. Also, it enables further improvements of the overall *T-depth*. The *T-depth* is reduced by 30% on average and in the best case (*mod5mils*) by 86%.

As can be clearly seen, the effect of incorporating the simulated annealing algorithm for reversible circuit optimization is significant. By rearranging the gates in a random way, further reductions are applied which confirms the idea outlined in Sect. III. Therefore this approach outperforms the greedy approach for most of the functions. Consider as an

TABLE I: Experimental Results For Greedy and Simulated Annealing Approaches

Original Benchmark (OB)				Greedy Ap. (GA)			Simulated An. (SA)			GA/OB				SA/OB				SA/GA			
ID	L	NCV	TD	NCV	TD	Time	NCV	TD	Time	Δ_{NCV}	I_{NCV}	Δ_{TD}	I_{TD}	Δ_{NCV}	I_{NCV}	Δ_{TD}	I_{TD}	Δ_{NCV}	I_{NCV}	Δ_{TD}	I_{TD}
Aj-e11	4	141	84	131	78	0.0	91	54	0.9	10	7%	6	7%	50	35%	30	36%	40	31%	24	31%
mod5mils	5	281	168	121	72	0.0	41	24	0.2	160	57%	96	57%	240	85%	144	86%	80	66%	48	67%
hwb4	4	403	240	368	219	0.0	229	135	0.8	35	9%	21	9%	174	43%	105	44%	139	38%	84	38%
mod5d2	5	641	384	636	381	0.0	451	270	1.3	5	1%	3	1%	190	30%	114	30%	185	29%	111	29%
4gt10	5	760	456	500	300	0.0	520	312	0.8	260	34%	156	34%	240	32%	144	32%	-20	-4%	-12	-4%
ex1	5	955	573	945	567	0.0	695	417	3.0	10	1%	6	1%	260	27%	156	27%	250	26%	150	26%
4gt5	5	1010	606	930	558	0.0	700	420	1.1	80	8%	48	8%	310	31%	186	31%	230	25%	138	25%
hwb5	5	2700	1617	2265	1356	0.1	1544	924	4.2	435	16%	261	16%	1156	43%	693	43%	721	32%	432	32%
C17	6	8332	4998	5937	3561	0.3	4977	2985	12.7	2395	29%	1437	29%	3355	40%	2013	40%	960	16%	576	16%
hwb6	6	12340	7404	9620	5772	0.7	8090	4854	23.9	2720	22%	1632	22%	4250	34%	2550	34%	1530	16%	918	16%
ham7	7	17340	10404	13020	7812	2.0	8580	5148	30.9	4320	25%	2592	25%	8760	51%	5256	51%	4440	34%	2664	34%
sym6	7	29492	17694	24022	14412	4.1	19567	11739	79.2	5470	19%	3282	19%	9925	34%	5955	34%	4455	19%	2673	19%
hwb7	7	44435	26661	34645	20787	13.3	29995	17997	154.3	9790	22%	5874	22%	14440	32%	8664	32%	4650	13%	2790	13%
con1	8	99306	59580	80356	48210	45.4	68956	41370	508.4	18950	19%	11370	19%	30350	31%	18210	31%	11400	14%	6840	14%
z4	8	105380	63225	86990	52191	86.3	74260	44553	555.7	18390	17%	11034	17%	31120	30%	18672	30%	12730	15%	7638	15%
hwb8	8	150315	90189	118795	71277	289.3	103445	62067	1121.3	31520	21%	18912	21%	46870	31%	28122	31%	15350	13%	9210	13%
sqrt8	9	282759	169650	232954	139767	927.2	205669	123396	351.7	49805	18%	29883	18%	77090	27%	46254	27%	27285	12%	16371	12%
radd	9	292495	175494	241895	145134	1166.7	211416	126846	383.0	50600	17%	30360	17%	81079	28%	48648	28%	30479	13%	18288	13%
plus63	12	317821	190692	313871	188322	10183.9	311026	186615	342.4	3950	1%	2370	1%	6795	2%	4077	2%	2845	1%	1707	1%
urf1	9	381172	228702	321182	192708	2386.8	285702	171420	632.6	59990	16%	35994	16%	95470	25%	57282	25%	35480	11%	21288	11%
hwb9	9	449579	269745	377349	226407	4451.1	316135	189678	889.5	72230	16%	43338	16%	133444	30%	80067	30%	61214	16%	36729	16%
x2	15	511366	306816	499346	299604	3007.6	474546	284724	297.3	12020	2%	7212	2%	36820	7%	22092	7%	24800	5%	14880	5%
5xp1	10	517028	310212	433568	260136	6341.1	378968	227376	1544.3	83460	16%	50076	16%	138060	27%	82836	27%	54600	13%	32760	13%
root	10	536719	322029	447239	268341	4182.2	394789	236871	1498.0	89480	17%	53688	17%	141930	26%	85158	26%	52450	12%	31470	12%
max46	10	652290	391374	539680	323808	7096.7	484300	290580	2219.2	112610	17%	67566	17%	167990	26%	100794	26%	55380	10%	33228	10%
dist	10	677078	406242	560483	336285	10876.7	492489	295488	2360.5	116595	17%	69957	17%	184589	27%	110754	27%	67994	12%	40797	12%
9symml	10	902254	541347	759794	455871	3459.4	665089	399048	4862.9	142460	16%	85476	16%	237165	26%	142299	26%	94705	12%	56823	12%
sym9	10	912693	547611	781628	468972	3491.1	673958	404370	4548.1	131065	14%	78639	14%	238735	26%	143241	26%	107670	14%	64602	14%
urf3	10	962832	577698	815702	489420	4560.5	724112	434466	4708.2	147130	15%	88278	15%	238720	25%	143232	25%	91590	11%	54954	11%
sqr6	12	1157296	694374	979286	587568	4185.7	898006	538800	5980.3	178010	15%	106806	15%	259290	22%	155574	22%	81280	8%	48768	8%
rd84	11	1747516	1048503	1493116	895863	19181.5	1322236	793335	16395.8	254400	15%	152640	15%	425280	24%	255168	24%	170880	11%	102528	11%
Average										**40274**	**17%**	**24165**	**17%**	**65872**	**30%**	**39523**	**30%**	**25598**	**16%**	**15359**	**16%**

example the function *hwb4*, its realization is reduced by 8% when the greedy approach is applied. Then additional 38% of improvement is achieved by applying simulated annealing. In general, the latter approach leads to additional quantum cost reductions of 16% in average compared to realizations optimized via the greedy approach.

VII. Conclusion

In this paper we introduced optimization approaches for reversible circuits based on rewriting rules. We presented two different strategies; a greedy approach and a simulated annealing approach. On our set of functions we showed that significant reductions (with respect to the NCV-cost and T-depth) can be achieved, specially when simulated annealing is considered.

References

[1] M. Arabzadeh, M. Saeedi, and M. S. Zamani, "Rule-based optimization of reversible circuits," in *Asia and South Pacific Design Automation Conference*, 2010, pp. 849–854.

[2] M. Soeken, Z. Sasanian, R. Wille, D. M. Miller, and R. Drechsler, "Optimizing the mapping of reversible circuits to four-valued quantum gate circuits," in *International Symposium on Multiple-Valued Logic*, 2012, pp. 173–178.

[3] Z. Sasanian and D. M. Miller, "Reversible and quantum circuit optimization: A functional approach," in *Reversible Computation*. Springer, 2013, pp. 112–124.

[4] D. Maslov, G. Dueck, and D. Miller, "Simplification of Toffoli networks via templates," in *Symposium on Integrated Circuits and Systems Design*, 2003, pp. 53–58.

[5] D. M. Miller, D. Maslov, and G. W. Dueck, "A transformation based algorithm for reversible logic synthesis," in *Design Automation Conference*, 2003, pp. 318–323.

[6] M. Soeken and M. K. Thomsen, "White dots do matter: rewriting reversible logic circuits," in *Reversible Computation*. Springer, 2013, pp. 196–208.

[7] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *SCIENCE*, vol. 220, no. 4598, pp. 671–680, 1983.

[8] T. Toffoli, "Reversible computing." Springer, 1980.

[9] A. Barenco, C. H. Bennett, R. Cleve, D. DiVinchenzo, N. Margolus, P. Shor, T. Sleator, J. Smolin, and H. Weinfurter, "Elementary gates for quantum computation," *The American Physical Society*, vol. 52, pp. 3457–3467, 1995.

[10] M. Sarkar, P. Ghosal, and S. P. Mohanty, "Reversible circuit synthesis using aco and sa based quine-mccluskey method," in *International Midwest Symposium on Circuits and Systems*, 2013, pp. 416–419.

[11] K. Datta, A. Gokhale, I. Sengupta, and H. Rahaman, "An esop-based reversible circuit synthesis flow using simulated annealing," in *Applied Computation and Security Systems*. Springer, 2015, pp. 131–144.

[12] D. M. Miller and Z. Sasanian, "Lowering the quantum gate cost of reversible circuits," in *International Midwest Symposium on Circuits and Systems*, 2010, pp. 260–263.

[13] M. Soeken, S. Frehse, R. Wille, and R. Drechsler, "Revkit: A toolkit for reversible circuit design." *Journal of Multiple-Valued Logic & Soft Computing*, vol. 18, no. 1, 2012, RevKit is available at http://www.revkit.org.

[14] R. Wille, D. Große, L. Teuber, G. W. Dueck, and R. Drechsler, "RevLib: an online resource for reversible functions and reversible circuits," in *International Symposium on Multiple-Valued Logic*, 2008, pp. 220–225, RevLib is available at http://www.revlib.org.

[15] D. Maslov. Reversible logic synthesis benchmarks page. Available at http://webhome.cs.uvic.ca dmaslov/, last accessed January 2011.

[16] M. Soeken, R. Wille, C. Hilken, N. Przigoda, and R. Drechsler, "Synthesis of reversible circuits with minimal lines for large functions," in *Asia and South Pacific Design Automation Conference*, 2012, pp. 85–92.