International Journal of Computer and Information Technology (ISSN: 2279 – 0764)
Volume 05– Issue 01, January 2016

Template Matching with Ranking for Toffoli Circuits

Md Zamilur Rahman
Departmentof Mathematics and Computer Science
University of Lethbridge
Lethbridge, AB, Canada
Email: mdzamilur.rahman [AT] uleth.ca

Jacqueline E. Rice
Departmentof Mathematics and Computer Science
University of Lethbridge
Lethbridge, AB, Canada

Abstract—**Circuit realizations generated by reversible logic synthesis approaches may not be optimal, thus it is common to apply post-synthesis optimization techniques. This paper proposes an algorithm that uses a ranking system for identifying the best match with circuit-reduction templates. These templates incorporate both positive and negative control Toffoli gates. A reduction in quantum cost was achieved for 86 of the 110 circuits. On average a 21.34% reduction in quantum cost was achieved, and in some cases up to 53.58% reduction was obtained.**

Keywords-reversible logic; Toffoli gate; template matching; quantum cost; gate count

I. INTRODUCTION

Research into reversible logic synthesis has begun to attract much attention due to the potential use of reversible logic in areas such as quantum computing [20], optical computing [3], and nanotechnology [16]. In particular, heatgeneration and dissipation are serious problems in today's traditional circuit technologies. In 1961 Landauer observed that the amount of energy dissipated for each lost bit of information is $KTln2$, where $K$is Boltzmann's constant $1.3807 \times 10^{-23} JK^{-1}$and T is the temperature [8]. Over millions of operations this becomes a significant amount of energy. However,in [1] Bennett showed that in order to not dissipate energy a system must be logically reversible. Research on reversible circuits may also be attracting attention due to the discovery of powerful quantum algorithms in the mid-1990s [20], as quantum circuits are inherently reversible. Interested readers can refer to [20] for a detaileddiscussion of quantum computing. In reversible circuits no information is lost as the underlying functions are all bijective. Thus fan-out and feedback operations are not allowed. Such features of reversible circuits prevent the use of existing algorithms and tools for circuit synthesis and optimization, thus leading to the need for logic synthesis approaches that are specifically targeted to reversible circuits. After an initial logic synthesis approach is applied the resulting circuit is often not optimal, leading to the need for an optimization phase as shown in Figure 1. The main focus of this paper is to offer an improved optimization phase incorporating the application of templates. This work builds on the proposal from [22], in which a set of templates for positive and negative control Toffoli gates was proposed.

The remainder of the paper is organized as follows. The following section gives an overview of the Toffoli gate and the cost metrics of a reversible circuit, followed by a description of template matching as a post-synthesis optimizationapproach. In section 3 we describe the basic template matching algorithm, the moving rule version that was proposed in [22], and the rank-based algorithm that is proposed in this work. Section 4 gives the results based on benchmarks as compared to both non-optimized circuits and to other techniques from the literature. Section 5 concludes the paper and provides possible directions for future work.

II. BACKGROUND

[13] states that if an n-input n-output function (gate) is a bijection then it is reversible. In other words, a reversible function (gate) has the same number of inputs and outputs and there is a one-to-one mapping between its input and output vectors. Traditional logic gates other than the NOT gate are not reversible. Reversible gates that have been proposed include Toffoli [24], Fredkin [7], and Peres [21] gates. In this work we focus only on the family of Toffoli gates.

A. Toffoli gates

An n-bit Toffoli gate or Multiple Control Toffoli (MCT) gate is a reversible gate with n inputs and n outputs where $(i_1, i_2, \ldots, i_n)$ is the input vector, $(o_1, o_2, \ldots, o_n)$is the output vector, and $o_j = i_j$ where $j = (1, 2, \ldots, n-1)$ and $o_n = i1,i2,\ldots,in-1 \oplus in$. The first $n-1$ bits are known as controls and the last n^{th} bit is known as the target. The MCT gate passes all the inputs to the outputs unchanged and inverts the target bit when all control bits have the value 1.When $n = 1$there are no controls and this gate is known as a NOT gate. When $n = 2$ the gate is known as a controlled-NOT (CNOT) gate or Feynman gate. For the sake of simplicity, we assume that the nth bit is the target; however, thetarget bit could be any of the n bits with which the gate interacts.

A negative-control Toffoli gate is a MCT gate that may have one or more negative controls. The gate maps the n inputs $(i_1, i_2, \ldots, i_n)$ to the n outputs $(o_1, o_2, \ldots, o_n)$ where $o_j = i_j$ for $j = (1, 2, \ldots, n-1)$ and $o_n = \overline{i_1}, i_2, \ldots, i_{n-1} \oplus i_n$where $\overline{i_1}$ is a negative control. Like the original MCT gate a MCT with negative controls gate passes all the inputs to the outputs unchanged; however, the target bit is inverted when all positive controls have value 1 and negative controls have value 0.

We use $\oplus$ to represent the target line, $\cdot$ to indicate a positive control, and $\circ$to indicate a negative control line. A

Toffoli gate can also be written as $T(C;t)$where C is the set of controls and t is the target line. The size of a Toffoligate refers to the number of controls plus target. Figure 2 illustrates different versions of the Toffoli gate.

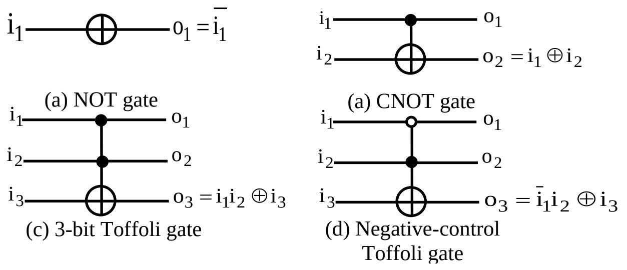


Figure 1. Toffoli gates.

Reversible circuits are created by cascading a series of reversible gates in such a way that the desired computation takes place.

B. Cost Metrics

A given reversible function may be realized in different ways, resulting in different circuits. The following two metrics are commonly used in evaluating the cost of different realizations. Gate count is the simplest way to evaluate different reversible circuits, and refers to a simple count of the number of gates in a circuit. It does not, however, consider the complexity of the circuit. For instance, if one circuit consists of two 6-input Toffoli gates and an equivalent circuit consists of three 2-input Toffoli gates, the first might seem preferable as it has fewer gates but these gates are significantly more complex in terms of quantum cost. The quantum cost of a gate is defined as the number of basic quantum operations needed to realize the gate [11]. Any reversible gate can be decomposed into basic quantum (1×1 and 2×2) gates. The NOT gate has a quantum cost of 1, as does the CNOT gate, while a 3-bit Toffoli gate has a quantum cost of 5. In general, as the number of controls for a gate increases so does the quantum cost.

The quantum cost of an n-bit negative control Toffoli gate with at least one positive control is exactly the same as the cost of an n-bit Toffoli gate. When all the controls are negative, an extra cost of 2 is required if the gate is to be implemented with zero or $(n-3)$ additional lines (referred to as garbage lines, as their values are not of interest at the output of the circuit) are used. An additional cost of 4 is required when only one garbage line is used [15].

C. Logic Synthesis

Logic synthesis is the process of converting a logic function into a high level circuit design in terms of gates. In reversible logic we refer to this as a cascade of gates. Reversible logic synthesis tools are used to generate a cascade of gates that computes the desired function. The circuit realizations obtained from different logic synthesis approaches may not be optimal in terms of the number of gates used, the quantum cost of those gates, and/or the number of lines (bits) required. Post-synthesis optimization phases may be applied in order to further reduce these costs as shown in Figure 1.

Figure 2. General flow in reversible logic synthesis approaches

An irreversible function can be embedded into a reversible function by adding constant inputs and garbage outputs [11]. A variety of synthesis approaches are available including those described in [6] and [17].

D. Template Matching

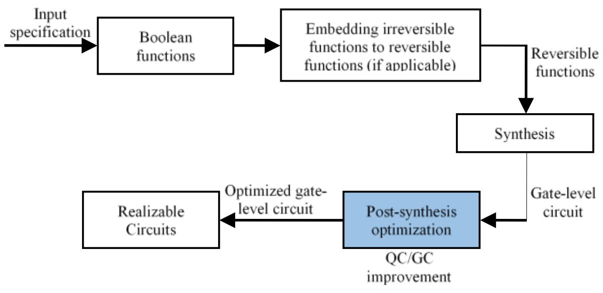
One post-synthesis optimization approach is template matching. If a circuit is non-optimal then it may be possible to decrease the size and quantum cost by replacing sequences of gates with shorter sequences that are equivalent in functionality. This is known as template matching [17].

The basic process of template matching is as follows: a circuit is examined to find a subsequence of gates (more than half) from a sequence that computes the identity; if such a subsequence is found then the matched sequence of gates in the circuit can be substituted with remaining sequence of gates in the identity circuit. The reader is directed to [13] for further details on the original template matching approach. These identity circuits are referred to as templates. One approach to template matching is to define all the templates up to a certain size for a given gate library. For instance, all Fredkin-Toffoli templates with less than six gates are given in [12], and all Toffoli gate templates of size up to 7 and some templates of size 9 can be found in [14].

However, the approach described above did not incorporate negative control Toffoli gates. In [2], the authors defined positive/negative control Toffoli gates as PNC gates. They also suggest rules for merging, moving, and splitting of PNC gates within a circuit so that the overall functionality of the circuit is not affected. This allows simplification of the circuit when e.g. identity circuits (templates) can be identified. A simplification algorithm utilizing these ruleswas proposed in [2].

Templates and rules using both positive and negative control Toffoli gates were proposed in [4]. They introduced templates that allow for a substitution of a cascade of (positively controlled) Toffoli gates with a single but an equivalent (negatively controlled) Toffoli gate. In [4], 7 generalized rules were proposed for post-synthesis optimization to reduce both the number of gates and the quantum costs [4]. The proposed algorithm traverses the given reversible circuit and checks for any possible rules until no further reduction is possible. In [5], the authors proposed an optimization algorithm that uses merging and replacement rules to optimize the circuits and showed that their algorithm was able to improve upon the results from [4].

[23] defined the negative/positive Toffoli gate as a Mixed


Input specification
Boolean functions
Embedding irreversible functions to reversible functions (if applicable)
Reversible functions
Synthesis
Gate-level circuit
Post-synthesis optimization
QC/GC improvement
Optimized gate-level circuit
Realizable Circuits

Polarity Multiple Control Toffoli (MPMCT) gate and proposed reduction rules that can be applied to MPMCT gates. A process for applying these reduction rules was alsoproposed.

III. Proposed Approach

In this section we give an overview of the previous work that the proposed algorithm builds upon, and then describe the new algorithm and how it improves upon the previous work.

[22] describes how Toffoli gates can appear in various ways in a circuit. Basedon the target line, pairs of gates in a circuit can be categorized as:

- same or different size gates having the same target line, or
- same or different size gates having different target lines.

Different target line Toffoli gates can be further classified as:

- different target line Toffoli gates having targets on any line, or
- different target line Toffoli gates having targets always other than control lines.

The templates used in this work were developed by considering the various ways in which two Toffoli gates with the same target line can appear in a circuit. Templates 1-5 can be applied to two adjacent Toffoli gates $T_1(C_1;t_1)$ and $T_2(C_2;t_2)$ where C_i is the set of controls, $|C_1| = |C_2|$and t_i is the target, $|t_1| = |t_2|$. In templates 1-4, two gates share the same control line while in template 5 one of the controls of one gate is on a different line. Templates 6-7 can be applied to two different size Toffoli gates $T_1(C_1;t_1)$ and $T_2(C_2;t_2)$ where C_i is the set of controls, $|C_1| > |C_2|$ or $|C_1| < |C_2|$, and t_i is the target, $|t_1| = |t_2|$. In template 6 the two gates may differ, but only by at most 1 line. In template 7, the difference in the size of two Toffoli gates is at least 1. In all cases we are interested in Toffoli gates that have the same target line. Details of the templates are given in [22].

A. Basic Template Matching Algorithm [22]

A basic template matching algorithm can be implemented as follows. Consider two adjacent gates g_1 and g_2 from the gate list of a circuit. This algorithm maintains two separate gate lists; the original list of gates, and a new list of gates that at the end of the algorithm will replace the original list.

1. ifg_1 and g_2 have the same target line then we begin searching for templates
   a. if g_1 and g_2 match any of the templates then replace g_1 and $g_2$withthe equivalent gates from that template (i.e. $\acute{g}_1, \acute{g}_2, \ldots$) and addthe new gates at the end of the new gate list and then move on toconsider the next two gates (i.e. g_3 and g_4) in the original gate list;go to step 1.
   b. if no match is found then add $g_1$at the end of the new gate list, g_2 and g_3 become the gates under consideration; go to step 1.
2. else add g_1 and g_2 at the end of new gate list and consider the next twogates (i.e. g_3 and g_4) in the original gate list; go to step 1.

This algorithm is iterated until no further reduction is possible in quantumcost i.e., after each iteration the quantum cost of the new gate list is comparedto the quantum cost of the old gate list. If there is a reduction in quantum cost,then the new gate list becomes the old gate list, and a new iteration begins.

B. Improved Algorithm [22]

The ability to rearrange gates within a circuit without changing the functionality increases the possibilities for matching more templates. Gate rearrangements are generally performed based on the moving rule [10]. The moving rule preserves the functional behavior of a circuit while moving gates within the circuit. In the example circuit shown in Figure 3a the gate count for this circuit is 7 and the quantum cost is 15. After rearranging gates and applying templates the gate count of the new circuit is 7 and quantum is 11. The gate rearrangements and templates applied are shown in Figure 3. The basic template matching techniques along with the moving rule are described below.

Moving Rule

Two adjacent gates $g(C_1;t_1)$ and $g(C_2;t_2)$ in a reversible circuit can be interchanged iff$C_1 \cap t_2 = \Phi$ and $C_2 \cap t_1 = \Phi$, i.e. the target of each gate is not a control of the other gate [10]. From Figure 3a and 3b we can that the moving rule allows the first CNOT gate $T(x_0;f_1)$ to pass the second CNOT gate $T(x_1;f_0)$because the target of the first gate is not the control of the second gate. This movement allows the application of template 6 on gates 2 and 3 and generates a new Toffoli gate with positive and negative controls.

Basic Algorithm with Moving Rule

It is possible to incorporate the moving rule into the basic algorithm as proposedin [22]:

Consider two gates g_1 and g_2 from the gate list of a circuit.

1. if g_1 and g_2 have the same target line then we can check for templatematches:
   a. if g_1 and g_2 match any of the templates then replace g_1 and g_2 withthe equivalent gates from that template (i.e. $\acute{g}_1, \acute{g}_2, \ldots$) and addthe new gates to the new gate list and then move on to consider thenext two gates (i.e. g_3 and g_4) in the original gate list; go to step 1.
   b. if no match is found for any template then apply the moving rule:
      i. if g_1 can pass g_2 then interchange g_1 and g_2; add g_2 into the newgate list, g_1 and g_3 become the gates under consideration; go tostep 1.
      ii. else add g_1 and g_2 to the new gate list and consider the next twogates from the

original gate list (i.e. g_3 and g_4); go to step 1.

2. else apply moving rule to g_1 and g_2
   a. if g_1 can pass g_2 then interchange g_1 and g_2; add g_2 into the new gatelist, g_1 and g_3 become the gates under consideration; go to step 1.
   b. else add g_1 and g_2 to the new gate list and consider the next twogates from the original gate list (i.e. g_3 and g_4)in the circuit; go tostep 1.

The algorithm continues until no further reduction is possible in quantumcost. After each iteration the quantum cost of the new gate list is compared tothe quantum cost of the old gate list. If there is a reduction in quantum cost,then the new gate list becomes the old gate list and a new iteration begins.

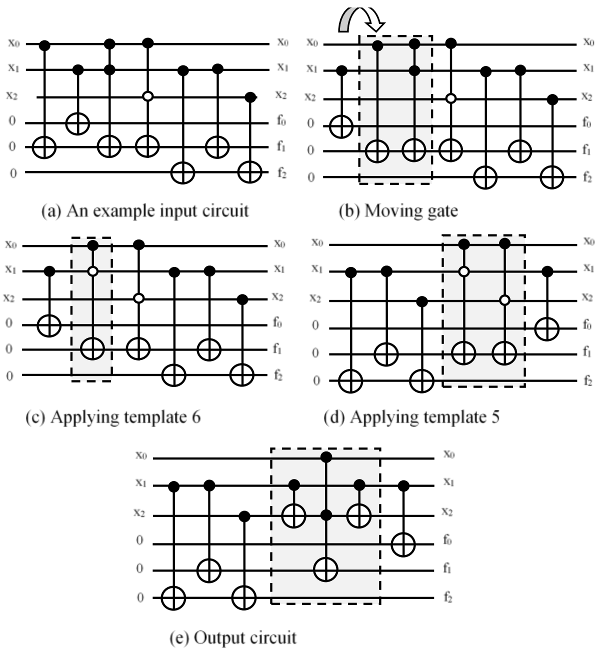


Figure 3. Illustration of applying moving rule

C. *Rank-based Template Matching Algorithm*

Neither of the previous two algorithms searched for the templates that offeredthe best match. In this section we propose a new algorithm which considers rankwhile applying templates. The rank of each template is found by consideringthe quantum cost savings that can be achieved when applying that template.

Quantum Cost Savings in Template

The template ranking strategies are summarized in Table 1 and explained below.

TABLE I. QUANTUM COST OF SAVINGS OF DIFFERENT TEMPLATES

Templates	*QC savings*	*Min. QC savings*	*Rank*
Template 1	$2p$to$m$	2 to 1	2
Template 2	$2x$to0	2 to 0	1
Template 3	$2x$ to$y$	10 to 1	3
Template 4	$2x$ to$2p+y$	10 to 3	4
Template 5	$2x$ to$2p+x$	10 to 7	5
Template 6	$y+x$ to$x$	6 to 5	6
Template 7	$x+y$ to$2q+y$	18 to 15	7

Template 1: Template 1 describes the case when a cascade of two CNOTgates can be replaced by a single NOT gate. If m and p is the quantum cost ofthe NOT and CNOT gate, respectively, then the quantum cost is reduced from$2p$ to m.

Template 2: Template 2 can be applied to two n-bit Toffoli gates with thesame controls. In this case the two gates negate each other and the gate countand quantum cost savings is 100%.

Template 3: In template 3 two n-bit Toffoli gates are replaced by one$(n-1)$-bit Toffoli gate. If the quantum cost of a n-bit Toffoli gate is x and thatof the $(n-1)$-bit Toffoli gate is y, then the quantum cost is reduced from $2x$ to$y$.

Template 4: Template 4 can be applied to two n-bit Toffoli gates with theconditions described in [22] where $n \geq 3$. The cascade is replaced by two CNOTgates and one $(n-1)$-bit Toffoli gate where $n \geq 2$. If x is the quantum costof an n-bit Toffoli gate, p is the quantum cost of a CNOT gate, and y is the quantum cost of an $(n-1)$-bit Toffoli gate, then the quantum cost is reducedfrom $2x$ to $2p+y$and the template is given a rank of 4.

Template 5: In template 5 the cascade of two n-bit Toffoli gates is replacedby two CNOT gates and one n-bit Toffoli gate. If x and p is the quantum costof an n-bit Toffoli gate and a CNOT gate, respectively, then the quantum costis reduced from $2x$ to $2p+x$.

Template 6: Template 6 can be applied to two different size Toffoli gateswith the conditions described in [22]. If the quantum cost of an n-bit Toffoligate is x and an $(n-1)$-bit Toffoli gate is y, then the quantum cost is reducedfrom $y+x$to x and the template is given a rank of 6.

Template 7: Template 7 can be applied to the various situations discussedin [22]. If x is the quantum cost of an n-bit Toffoli gate, q is the quantum cost ofa3-bit Toffoli gate, and y is the quantum cost of an $(n-1)$-bit Toffoli gate, thenthe quantum cost is reduced from $x+y$to $2q+y$and the template is assigneda rank of 7.

As an example of how these rankings are used, we consider an arbitrary circuit shown in Figure 4a where the circuit has three Toffoligates. All the gates have the same target line. We can see that gates 1 and 2satisfy the conditions to apply template 4. Now we set rank 4 for gates 1 and 2and save it. According to the moving rule gate 1 can pass gate 2 since none ofthe controls of either gate is the target of the other gate. As shown in Figure 4b,we can see that gates 1 and 3 satisfy the conditions to apply template 7. We setrank 7 for gates 1 and 3. Now we compare the new rank (7) with the previousrank (4). The new rank is higher than the previous thus we apply template 4on gates 1 and 2 and replace with the new set of gates as shown in Figure 4c.In each iteration for each gate in a circuit and considering the moving rule wecompare the rank and take the best pair that offers the best savings in quantumcost after applying different templates.

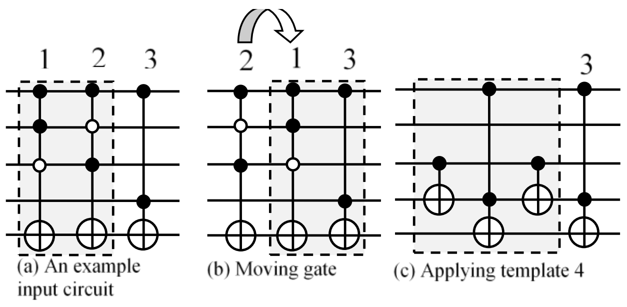


Figure 4. Illustration of rank-based template matching algorithm

Basic Rank-based Algorithm

The algorithm is as follows: consider two gates (g_i and g_{i+1}, $(1 \leq i \leq n)$ where i is the index of a gate and n is the number of gates in a circuit) from a circuit.

1. If g_i and g_{i+1} have the same target, then the algorithm searches for templates and sets the rank as described in Table 1.
2. If g_i can pass g_{i+1}, then we consider g_i and the next gate after g_{i+1} (i.e. g_{i+2}) from the circuit. If g_i and g_{i+2} have the same target, then thealgorithm searches for templates and sets the rank.
3. Now we compare the new rank with the previous rank and update theprevious rank with the new rank if the new rank is less than the previousrank at this stage.
4. Next, if gate g_i can also pass g_{i+2}, then we consider g_i with the next gateof g_{i+2} and check conditions to apply templates and update the previousrank with the new rank.
5. After considering g_i with all the other gates in the circuit, we get the bestrank for g_i and g_j, $(i+1 \leq j \leq n$.
6. We then apply the template on g_i and g_j and replace the gates with thenew set of gates (i.e. $\acute{g}_1, \acute{g}_2, \ldots$).

In this way, the algorithm searches for all possible matches and replaces thegate list with the new gate list with highest rank.

Modification 1

In the rank-based algorithm described in section 3.3, we applied templates basedon the highest ranking template matching the gates under consideration, gate g_i and g_j . We did not consider that the gate g_j paired with gate g_k can have ahigher rank. In this algorithm we consider the highest rank among all allowablegate pairs before applying templates. The previous algorithm visits all the gatesin a circuit to check the conditions and select the best template. In this case, atthe beginning, the algorithm indexes all the gates by the target and generatessub-gate lists for each different target. It then goes on to preprocess each gatefor all the sub-gate lists and store ranks and the pair gate. The next step is toassign rank to gates by considering the moving rule and then applying templatesbased on the rank listed in Table 1. This algorithm iterates over each gate fromeach sub-gate list and thus reduces the number of iterations compared to theprevious approaches. For some of the cases we get slightly improved resultscompared to the results listed in Table 2, but the average results are almost same(16.39% improvement over non-post-processed circuits). One reason behind thesimilar results despite the additional gates being considered is that One of theobservations to get almost similar results is that this algorithm applies templatesamong all the gates. If we apply template 4, 5, or 7 and then if one of the controlsof the next gates has a target on this line, then the gate cannot move. However,if we apply template 1, 2, 3, or 6 before applying template 4, 5, or 7, then allthe gates after these gates can move in the circuit by following the moving rule;this increases the chances to apply more templates.

Modification 2

Based on the results from the previous algorithm, we re-ranked the templatesby considering the number of increased/decreased gates in the template. If weapply templates 1, 2, 3, and 6, the number of gates decreases to 1; on theother hand, if we apply templates 4, 5, and 7, the number of gates increasesto 3. Based on this observation we re-ranked template 6 as rank 4, template4 as rank 5, and template 5 as rank 6 and kept templates 1, 2, 3, and 7 asrank 2, 1, 3, and 7, respectively. The algorithm works in the same way as thealgorithm described in the previous section. Overall we get slightly improvedresults (17.22% improvement over non-post-processed circuits) compared to theapproach proposed in section 3.3 (16.64%) and section 3.3.3 (16.39%).

IV. Experimental Results and Discussion

The templates and moving rule algorithm from [22] and the new rank-based algorithm were implemented in Java. Tests were run on an Intel Core 2 Duo CPU T6670 @ 2.20GHz_2 system running Ubuntu 13.04 with 2GB main memory for 110 benchmark circuits. All benchmarks used in this work were obtained from RevLib [25] and preprocessed by applying the improved shared cube synthesis approach from [19]. All the resulting circuits are verified using QMDD (Quantum Multiple-valued Decision Diagrams) [18]. Using QMDD, we compare the resulting circuits (after applying templates) with the input circuits in order to ensure that the behavior of the circuit has not been modified.

The run time of all the tested benchmarks was under 114,299 milliseconds(ms), and our experiments showed that each circuit was iterated over at most 7 times.

When comparing the rank-based algorithm with the moving rule algorithm [22] 56 benchmarks showed improvement. The results are summarized in Table 2. Both the moving rule and rank-based results are compared to the circuits obtained from the improved shared cube synthesis approach [19] in terms of quantum cost and gate count. In this table PrevGC/PrevQC (column 2 and 3) refers to the gate count/quantum cost obtained from the circuit generated by the improved shared cube synthesis approach. The % Improvement for the moving rule and rank-based algorithms indicate the improvement in size as compared to the shared cube synthesis. For the improved 56 circuits the average quantum cost reduction is 23.80% compared to the improved shared cube synthesis approach.

TABLE II. RESULTS FROM THE MOVING RULE AND RANK-BASED ALGORITHM

Circuit	[19]	[22]		rank-based	
	GC/QC	*GC/%Impr.*	*QC/%Impr.*	*GC/%Impr.*	*QC/%Impr.*
cm85a_127	48/2206	54/-12.50	1232/44.15	55/-14.58	1024/53.58
4mod5_8	4/21	4/0.00	13/38.10	5/-25.00	10/52.38
adr4_93	41/645	41/0.00	489/24.19	38/7.32	330/48.84
bw_116	287/637	94/67.25	387/39.25	77/73.17	332/47.88
C7552_119	89/399	32/64.04	250/37.34	29/67.42	210/47.37
decod_137	89/399	32/64.04	250/37.34	29/67.42	210/47.37
0410184_85	218/7636	256/-17.43	5662/25.85	257/-17.89	4631/39.35
add6_92	153/5135	159/-3.92	3714/27.67	154/-0.65	3267/36.38
rd73_69	43/856	52/-20.93	619/27.69	51/-18.60	581/32.13
ham15_30	114/263	46/59.65	183/30.42	45/60.53	179/31.94
clip_124	78/3824	80/-2.56	2803/26.70	85/-8.97	2726/28.71
3_17_6	11/28	9/18.18	22/21.43	7/36.36	20/28.57
sym6_63	13/721	16/-23.08	571/20.80	17/-30.77	521/27.74
sqrt8_205	22/466	23/-4.55	393/15.67	24/-9.09	339/27.25
urf2_73	479/8742	254/46.97	7453/14.74	272/43.22	6395/26.85
dc1_142	31/127	18/41.94	97/23.62	19/38.71	93/26.77
dc2_143	51/1084	39/23.53	906/16.42	38/25.49	794/26.75
cycle10_2_61	42/1273	46/-9.52	1004/21.13	45/-7.14	934/26.63
max46_177	42/4524	52/-23.81	3540/21.75	54/-28.57	3324/26.53
hwb8_64	480/8195	261/45.63	7158/12.65	270/43.75	6172/24.69

In order to situate our work against other work in the literature we also compared the ranking algorithm's results with the results from [4] and [5]. Theresults are listed in Table 3 sorted in order of quantum cost improvement. The column RevLibGC/QC represents the gate count and quantum cost of the circuit obtained from RevLib. The gate count and quantum cost of these circuits are calculated based on the values in [9]. The column NewGC/QC gives the gate count and quantum cost resulting from the proposed rank-based template matching algorithm. Compared to [4] and [5], the average quantum cost reduction for the top 19 circuits is 13.19% and 9.40%, respectively.

TABLE III. RESULTS FROM THE MOVING RULE AND RANK-BASED ALGORITHMS

Circuit	RevLib [25]	[4]	[5]	rank-based	Impr.
	GC/QC	*GC/QC*	*GC/QC*	*GC/QC*	*GC/QC (%)*
sym9_148	210/4368	154/3668	143/3433	276/2189	40.32/36.24
sym6_145	36/777	31/647	23/517	48/408	36.94/21.08
max46_240	107/5444	51/4498	52/4538	112/3632	19.25/19.96
add6_196	229/6455	179/6005	167/5534	244/4581	23.71/17.22
clip_206	174/6731	111/6535	109/6119	173/5462	16.42/10.74
life_238	107/6766	57/5740	57/5744	99/5210	9.23/9.30
sym9_193	129/14193	58/12747	63/13090	124/11929	6.42/8.87
9symml_195	129/14193	58/12747	62/13026	124/11929	6.42/8.42
alu4_201	1063/55388	523/46413	529/46764	1013/43097	7.14/7.84
mux_246	35/1078	20/804	20/804	42/742	7.71/7.71
tial_265	1041/56203	516/47145	522/47556	981/44394	5.84/6.65
apla_203	80/3438	74/3438	64/3024	78/2839	17.42/6.12
f51m_233	663/37400	358/33333	355/32882	637/30981	7.06/5.78
dc2_222	75/1886	55/1789	53/1688	78/1592	11.01/5.69
mod5adder_306	96/292	84/281	70/270	72/265	5.69/1.85
hwb5_300	88/276	80/270	67/259	67/255	5.56/1.54
in2_236	405/23802	283/23146	250/20600	394/20289	12.34/1.51
alu2_199	157/5654	87/4776	87/4611	147/4561	4.50/1.08
cycle10_293	78/202	74/199	57/186	56/184	7.54/1.08
Average					**13.19/9.40**

V. CONCLUSION AND FUTURE WORK

The proposed approach for post-synthesis optimization consider templates that make use of both positive and negative control Toffoli gates and uses a ranking system to find the best possible match that will locally result in the highest savings in quantum cost. The approach has been experimentally demonstrated to obtain improvements over a popular non-optimized approach (shared cube), and also gives further improvement over comparable works in the literature ([4] and [5]).

Future work may pursue several avenues related to this work, including identifying additional templates, particularly for Toffoli gates with different target lines, and also improving the template matching algorithm. Of course, the issue of template matching with negative controls has not yet been thoroughly studied, and as we pursue this work a broader investigation will also be required. All positive control Toffoli gate templates of size 7 and some templates of size 9 are listed in [14]. Other templates for Toffoli gates with positive and negative controls were proposed in [2, 4, 5, 23]; however, finding and classifying a complete set of templates for positive and negative control Toffoli gate is another direction for further research. Garbage/line count reduction through the application of templates is also a possible research direction.

Acknowledgment

This research was funded by a grant from the Natural Sciences and EngineeringResearch Council of Canada (NSERC).

References

[1] C. H. Bennett. Logical reversibility of computation. IBM Journal of Research and Development, 17(6):525-532, November 1973.

[2] Xueyun Cheng, Zhijin Guan, WeiWang, and Lingling Zhu. A simplification algorithm for reversible logic network of positive/negative control gates. In Fuzzy Systems and Knowledge Discovery (FSKD), 2012 9th International Conference on, pages 2442-2446, May 2012.

[3] R. Cuykendall and D. R. Andersen. Reversible optical computing circuits. Optics Letters, 12(7):542-544, 1987.

[4] K. Datta, G. Rathi, R. Wille, I. Sengupta, H. Rahaman, and R. Drechsler. Exploiting negative control lines in the optimization of reversible circuits. In Gerhard W. Dueck and D. Michael Miller, editors, Reversible Computation,volume 7948 of Lecture Notes in Computer Science, pages 209-220. Springer Berlin Heidelberg, 2013.

[5] K. Datta, I Sengupta, and H. Rahaman. A post-synthesis optimization technique for reversible circuits exploiting negative control lines. Computers, IEEE Transactions on, 64(4):pages 1208-1214, 2015.

[6] K. Fazel, M. A. Thornton, and J. E. Rice. ESOP-based Toffoli gate cascade generation. In Communications, Computers and Signal Processing, 2007. PacRim 2007. IEEE Pacific Rim Conference on, pages 206-209, 2007.

[7] E. Fredkin and T. Toffoli. Conservative logic. International Journal of Theoretical Physics, 21:219-253, 1982.

[8] R. Landauer. Irreversibility and heat generation in the computing process. IBM Journal of Research and Development, 44(1.2):261-269, 2000.

[9] D. Maslov. Reversible logic synthesis benchmarks page, http://www.cs.uvic.ca/~dmaslov/.

[10] D. Maslov. Reversible Logic Synthesis. PhD thesis, University of New Brunswick, 2003.

[11] D. Maslov and G. W. Dueck. Reversible cascades with minimal garbage. Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on, 23(11):1497-1509, 2004.

[12] D. Maslov, G. W. Dueck, and D. M. Miller. Fredkin/Toffoli templates for reversible logic synthesis. In Computer Aided Design, 2003. ICCAD-2003. International Conference on, pages 256-261, Nov 2003.

[13] D. Maslov, G. W. Dueck, and D. M. Miller. Toffoli network synthesis with templates. Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on, 24(6):807-817, 2005.

[14] D. Maslov, G. W. Dueck, and D. M. Miller. Techniques for the synthesis of reversible Toffoli networks. ACM Transactions on Design Automation of Electronic Systems, 12(4):42-1-42-28, September 2007.

[15] D. Maslov, G. W. Dueck, D. M. Miller, and C. Negrevergne. Quantum circuit simplification and level compaction. Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on, 27(3):436-444, March2008.

[16] R. C. Merkle. Reversible electronic logic using switches. Nanotechnology,4:21-40, 1993.

[17] D. M. Miller, D. Maslov, and G. W. Dueck. A transformation based algorithm for reversible logic synthesis. In Design Automation Conference, 2003. Proceedings, pages 318-323, 2003.

[18] D. M. Miller and M. A. Thornton. QMDD: A decision diagram structure for reversible and quantum circuits. In Multiple-Valued Logic, 2006. ISMVL 2006. 36th International Symposium on, pages 30{35, May 2006.

[19] N. M. Nayeem. Synthesis and Testing of Toffoli Circuits. Master's thesis, University of Lethbridge, 2011.

[20] M. Nielsen and I. Chuang. Quantum Computation and Quantum Information. Cambridge University Press, 2000.

[21] A. Peres. Reversible logic and quantum computers. Physical Review A,32(6):3266-3276, 1985.

[22] M. Z. Rahman and J. E. Rice. Templates for positive and negative control Toffoli networks. In Shigeru Yamashita and Shin-ichi Minato, editors, Reversible Computation, volume 8507 of Lecture Notes in Computer Science, pages 125-136. Springer International Publishing, 2014.

[23] Z. Sasanian. Technology Mapping and Optimization for Reversible and Quantum Circuits. PhD thesis, University of Victoria, 2012

[24] T. Toffoli. Reversible computing. Tech Memo LCS/TM-151, MIT Lab for Computer Science, 1980.

[25] R. Wille, D. Gro_e, L. Teuber, G. W. Dueck, and R. Drechsler. RevLib: An online resource for reversible functions and reversible circuits. In Int'l Symp. on Multi-Valued Logic, pages 220-225, 2008. RevLib is available at http://www.revlib.org.