Quantum Logic Synthesis by Symbolic Reachability Analysis

William N. N. Hung[1], Xiaoyu Song[2], Guowu Yang[2], Jin Yang[1], and Marek Perkowski[2]
[1]Intel Corporation, Hillsboro, Oregon, 97124, USA
[2]Portland State University, Portland, Oregon, 97201, USA

ABSTRACT

Reversible quantum logic plays an important role in quantum computing. In this paper, we propose an approach to optimally synthesize quantum circuits by symbolic reachability analysis where the primary inputs are purely binary. We present an exact synthesis method with optimal quantum cost and a speedup method with non-optimal quantum cost. Both our methods guarantee the synthesizeability of all reversible circuits. Unlike previous works which use permutative reversible gates, we use a lower level library which includes non-permutative quantum gates. Our approach obtains the minimum cost quantum circuits for Miller's gate, half-adder, and full-adder, which are better than previous results. In addition, we prove the minimum quantum cost (using our elementary quantum gates) for Fredkin, Peres, and Toffoli gates. Our work constitutes the first successful experience of applying satisfiability with formal methods to quantum logic synthesis.

Categories and Subject Descriptors: B.6.3 [Logic Design]: Design Aids **General Terms:** Design, Algorithms.

Keywords: Reversible Logic, Quantum Computing, Formal Verification, Model Checking, Satisfiability.

1. INTRODUCTION

Reversible logic is needed in the synthesis of quantum computing circuits [5,7,12]. The synthesis of reversible logic circuits using elementary quantum gates [1, 21] is different from classical (non-reversible) logic synthesis. There are some works [13, 16, 19, 23] on reversible logic synthesis using permutative reversible gates (Toffoli [7], Fredkin [21] or Feynman gates). However, these gates have different quantum costs (e.g. the cost of Feynman is lower than Toffoli). So finding the smallest number of gates to synthesize a reversible circuit does not necessarily result in a quantum implementation with the lowest cost (in terms of quantum gates). In this paper, we focus on synthesizing reversible circuits to quantum implementations with the lowest cost. These circuits include common reversible gates that can be used at higher levels of logic synthesis or for technology mapping. We reduce the quantum logic synthesis problem to multiple-valued logic synthesis, which simplifies the search space and algorithm complexity. We formulate the quantum logic synthesis problem via symbolic reachability analysis [2, 14]. Our method not only guarantees to find a quantum implementation (for reversible circuits), but also the lowest quantum cost in the synthesized result. In contrast to previous works, which either use permutative reversible gates to design permutative circuits or universal quantum gates to design quantum circuits, we use a subset of quantum gates to design permutative circuits.

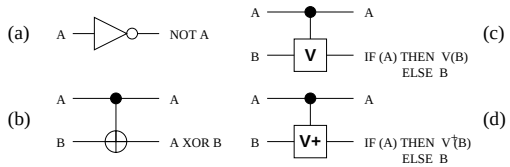

Figure 1: Elementary quantum logic gates

2. BACKGROUND

It has been shown [1, 21] that any quantum logic can be constructed using elementary quantum XOR, controlled-V, controlled-V^+, or NOT gates, as shown in Fig. 1. The NOT gates are also known as inverters. The quantum XOR gates are also called Feynman gates or controlled-NOT (CNOT) gates. The controlled-V [17] gate's data output is the same as its data input (B) when its control input (A) value is 0 (FALSE). When its control value is 1 (TRUE), the data output becomes *V(input)*. Similar rules apply to the controlled-V^+ gate, except that its data output becomes V^+*(input)*. According to [17], the values V and V^+ are constructed such that $V \times V = V^+ \times V^+ = \text{NOT}$. For any unitary matrix X (e.g. V), X^+ is its hermitian matrix where $X \times X^+ = I$ (identity). For quantum implementation, the cost of 2-qubit gates (Fig. 1 b,c,d) far exceeds the cost of 1-qubit gates (Fig. 1a). Hence, in a first approximation the quantum cost of 1-qubit gates is usually ignored in the presence of 2-qubit implementations [1, 5].

In this paper, we adopt the quantum gate cost evaluation introduced originally in [21]. Each 2-qubit gate has a quantum implementation cost of 1; and each symmetric gate pattern (shown in Fig. 2) has a cost of 1.

Given a reversible function, the quantum logic synthesis problem is to synthesize the function using the above

DAC 2004, June 7–11, 2004, San Diego, California, USA.

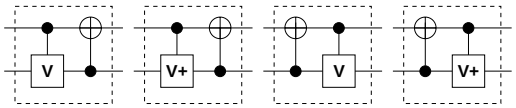

Figure 2: Merged 2-qubit gates

elementary quantum logic gates with the minimum cost. Various heuristic methods have been applied to find low cost quantum implementations (using the elementary gates) for the functionality of the Fredkin [21], Toffoli [20] and Peres [18] gates. Yet, nobody has been able to prove that they have the lowest cost implementation (based on the cost evaluation criteria above).

We solve the quantum logic synthesis problem through reachability analysis. Symbolic reachability analysis is a well known technique in formal verification [14]. Its basic idea is to find all the reachable states of a finite state machine (FSM). Using a symbolic representation, we can check if an invariant (property) is true for all reachable states. This technique is used in invariant checking [14] where the state space is traversed exhaustively against an invariant. We use the state-of-the-art satisfiability (SAT) based bounded model checking (BMC) [2] to check invariants. If the invariant is false, it can automatically generate a counter-example. We can find the shortest counter-example by starting with a zero bound and gradually incrementing the bound. If the invariant is true and enough time is given, this method can also check that the bound is sufficiently large and establish the proof. SAT based applications have been successfully deployed in industry [4, 11, 22].

3. SYMBOLIC FORMULATION

We consider each "quantum wire" of the quantum circuit as a superposition of $|1\rangle$ and $|0\rangle$, denoted as 1 and 0, respectively. We are interested in synthesizing quantum circuits with pure binary inputs (1 and 0). The values of these signals (quantum wires) are modified after passing through the elementary gates (Fig. 1). There are six possible output values when we apply binary (1 and 0) inputs to one of those elementary gates: 0, 1, V_0, V_1, $V_0{}^+$, $V_1{}^+$, where V_0 represents *V(input)* when input is 0, and similarly for $V_1, V_0{}^+, V_1{}^+$. These values are used as input values to gates in subsequent stages. We want to synthesize our circuit such that the inputs of XOR and NOT gates and the "control" input of controlled-V and controlled-V^+ will always be pure binary (0's and 1's), i.e., their input values cannot be V_0, etc. As shown in Section 2, given the above six possible values at the data input of the controlled-V or controlled-V^+, their corresponding data output has the same set of six possible values. Hence the input/output of every quantum gate in the circuit can be represented using the above six values. Since V and V^+ have a dual relationship (Section 2), we have $V_0 = V_1{}^+$ and $V_1 = V_0{}^+$. Thus, it suffices to represent signals in the circuit using four values: $0, 1, V_0, V_1$. In this way, we reduce the problem of quantum circuit synthesis, (that would normally use unitary matrices and Hilbert space to represent signals), to a simpler synthesis problem in mixed binary/quaternary algebra. This is a general approach to efficiently synthesize a subclass of quantum circuits.

Suppose we intend to synthesize a $n \times n$ reversible function R, using the 2 qubit quantum gates as described in

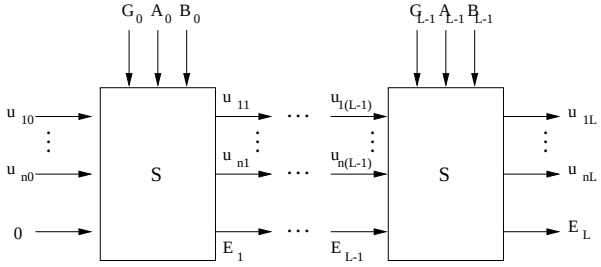

Figure 3: L-2Syn problem

Fig. 1 b,c,d. The synthesized result should be a cascade of L stages. Each stage consists of one of the above quantum gates. Since the function applies to n qubits, and the quantum gates at each stage are 1-qubit or 2-qubit gates, the synthesized result should indicate to which qubits the gates are connected. For each stage i, we use g_i to represent the gate selection (Fig. 1 b,c,d), and we use A_i and B_i to indicate the two qubits that the gate is connected to, i.e., $A_i, B_i \in \{1, \ldots, n\}$. We require: $A_i \neq B_i$. As shown in Fig. 3, the input of stage i is $\vec{U}_i$, where $\vec{U}_i = u_{1i}u_{2i}\cdots u_{ni}$, such that $u_{qi} \in \{0, 1, V_0, V_1\}$ for $q = 1, \ldots, n$. The output of stage i is $\vec{U}_{i+1}$.

$$u_{q(i+1)} = \begin{cases} u_{A_i i} \oplus u_{qi} & (q = B_i) \wedge (g = \oplus) \\ V(u_{qi}) & (q = B_i) \wedge (g = \text{ctl-}V) \wedge u_{A_i i} \\ V^+(u_{qi}) & (q = B_i) \wedge (g = \text{ctl-}V^+) \wedge u_{A_i i} \\ u_{qi} & \text{Otherwise} \end{cases}$$

The input values V_0 and V_1 are meaningful only at particular inputs of the controlled-V and controlled-V^+ gates. We create a boolean signal E_i to represent whether the gate has been erroneously configured with the V_0 or V_1 values in the current (i-th) synthesis stage or any previous synthesis stages: $E_{i+1} = E_i \vee (u_{A_i i} \notin \{0, 1\}) \vee ((g_i = \oplus) \wedge (u_{B_i i} \notin \{0, 1\}))$. For better quantum cost, let us use a different gate select G_i by adding the merged gates in Fig. 2:

$$(\vec{U}_{i+1}, E_{i+1}) = S(G_i, A_i, B_i, \vec{U}_i, E_i) \quad (1)$$

Definition 1. **(L-2Syn)** The synthesis of reversible function R using 2 qubit gates in L stage cascade is to find G_i, A_i, B_i, (where $A_i \neq B_i$) such that $E_0 = E_L = 0$ and $\vec{U}_L = R(\vec{U}_0)$ for all boolean inputs of $\vec{U}_0$. It is equivalent to:

$$\exists G_0 \exists A_0 \exists B_0 \ldots \exists G_{L-1} \exists A_{L-1} \exists B_{L-1} \cdot \left(\bigwedge_{i=0}^{L-1} A_i \neq B_i \right) \wedge \left(\forall \vec{U}_0 \in \{0, 1\}^n \cdot (E_0 = E_L = 0) \wedge (\vec{U}_L = R(\vec{U}_0)) \right) \quad (2)$$

Note that E_0 is not an input constant to the reversible logic circuit.

THEOREM 1. *For any reversible function R that is realizable without inverters, its quantum logic implementation with the minimum cost is equivalent to solving the L-2Syn with the smallest L.*

PROOF. Each stage of the L-2Syn solution has a quantum cost of 1. Thus the minimum quantum cost is L. □

We can also modify the above formulation using 1-qubit (inverters) and other 2-qubit gates.

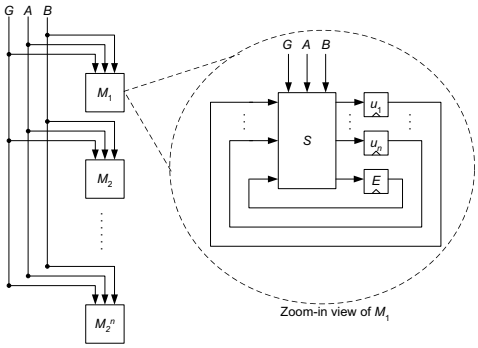

Figure 4: FSM for Reachability Analysis

4. REACHABILITY ANALYSIS

We construct a FSM shown in Fig. 4, use a bounded model checker [2] for temporally unrolling the FSM upto a specific bound, and invoke a SAT solver to find a counter-example. The FSM will be initialized at time $t = 0$. We use $\vec{\mu}_{M_h t}$ to denote the value of register vector $u_1, \ldots, u_n$ of machine M_h at time t, where $h = 1, \ldots, 2^n$. Similarly, we use $\varepsilon_{M_h t}$ to denote the value of register E of machine M_h at time t. We assume: $\forall t \geq 0$ $(A_t \neq B_t)$. From Fig. 4, we can see that the next state is computed from the current state and inputs through the combinational functional block S. $(\vec{\mu}_{M_h t+1}, \varepsilon_{M_h t+1}) = S(G_t, A_t, B_t, \vec{\mu}_{M_h t}, \varepsilon_{M_h t})$ We initialize the E register of every machine to 0 (FALSE): $\varepsilon_{M_h 0} = 0$ for $h = 1, \ldots, 2^n$. We also initialize the $\vec{U}$ registers of every machine to their corresponding patterns in a truth table: $\vec{\mu}_{M_1 0} = 0 \ldots 0$, $\ldots$, $\vec{\mu}_{M_{2^n} 0} = 1 \ldots 1$. We check the non-synthesizeability invariant: $inv(t)$

$$inv(t) = \neg \bigwedge_{h=1}^{2^n} ((\vec{\mu}_{M_h t} = R(\vec{\mu}_{M_h 0})) \wedge (\varepsilon_{M_h t} = 0)) \qquad (3)$$

THEOREM 2. *The function R is synthesizeable if and only if there exists a counter-example of $inv(t)$ at time $t = L$.*

PROOF. The existence of a counter-example to $inv(t)$ can be re-written by the assumption and initial condition as:

$$\exists G_0 \exists A_0 \exists B_0 \ldots \exists G_L \exists A_L \exists B_L \cdot \bigwedge_{t=0}^{L} (A_t \neq B_t) \wedge$$

$$\forall \vec{U}_{t=0} \in \{0,1\}^n \cdot (E_{t=0} = E_{t=L} = 0) \wedge (\vec{U}_{t=L} = R(\vec{U}_{t=0}))$$

Observe that the registers (E and $\vec{U}$) in Fig. 4 depend only on inputs of the previous time, making $G_L A_L B_L$ redundant in the above formula. Hence it is equivalent to (2). □

Quantum logic synthesis is equivalent to finding a counter-example to our formulation. By starting with a small bound and gradually increasing the bound, we can find the shortest counter-example, essentially the minimum cost quantum implementation of the function R. If R is not synthesizeable, the model checker will prove the invariant has no counter-example (Theorem 2). Alternatively, we can use BDD [3] based model checking [14] for the same purpose.

Our method works not only for reversible circuits, but also for non-reversible circuits by adding input constants (ancilla qubits) in order to convert a non-reversible logic function into a reversible one [7, 17]. Input constants are also needed in some reversible cases (e.g. [19]). We can add k input constants to the original $n \times n$ circuit, making it a $(n + k) \times (n + k)$ circuit with some minor changes, run it through our model checker and see if we can get a counter-example or a proof. If we get a proof, we can increment k until we eventually get a counter-example (happens for finite k [7]). A systematic way is to start with $k = 1$ and gradually increment k until we reach a counter-example.

Table 1: Quantum cost of common circuits

Circuit	Prior	Our	Optimum	Time (sec)
Miller	7	6	Yes	318.29
Fredkin	5	5	Yes	78.02
Peres	4	4	Yes	35.18
Toffoli	5	5	Yes	122.52
Half-adder	6	4	Yes	6.77
Full-adder	12	6	Yes	7 hours
Full-adder	12	9	No	140.83

5. NON-OPTIMAL SYNTHESIS

Industrial experience [4] suggests that the complexity of model checking is sensitive to the number of state retaining elements in a FSM. For our FSM in Fig. 4, there are $2n \times 2^n$ boolean state variables. However, n tends to be small due to physical limitations (the largest number [24] of qubits is 7). Nevertheless, we would like to speed up our synthesis process.

THEOREM 3. *$\forall R \forall Q \exists P$ $R = Q \circ P$, where R, Q, P are all $n \times n$ reversible functions, and $\circ$ is function composition.*

PROOF. Since Q is reversible, we have function Q^{-1} such that $Q \circ Q^{-1} = I$, where I is the identity function. Hence there exists $P = Q^{-1} \circ R$, such that $Q \circ P = Q \circ Q^{-1} \circ R = I \circ R = R$ □

We devised a strategy to speed up the synthesis process at the expense of a higher circuit cost. Given an $n \times n$ reversible gate to synthesize, there are 2^n cases to be enumerated. We pick one of the inputs, say the first input, and consider only the cases where it is 0. Then we have 2^{n-1} cases. To perform reachability analysis, we construct the same FSM as shown in Fig. 4, but check it with a different invariant $inv'(t)$. This new invariant $inv'(t)$ checks that R is accomplished for only half of all possible input patterns (cases where the first input is 0). It is easier to find a counter-example for this new invariant, because only half of the cases has to be accomplished. We take a snap-shot of all register states at the end of this counter-example, and use it as the initial state of the FSM. We then run model checker again with our original invariant $inv(t)$. Since we started from a state fairly close to R, it is easier to generate a counter-example. According to Theorem 3, this method guarantees to generate the counter-example if the function that we want to synthesize is reversible.

6. EXPERIMENTS

We constructed our invariant checking formulations using NuSMV with BerkMin [8] on a 850MHz Pentium®III processor running Linux.

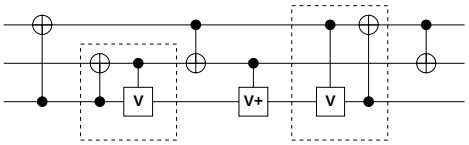

Figure 5: Miller's gate with quantum cost = 6

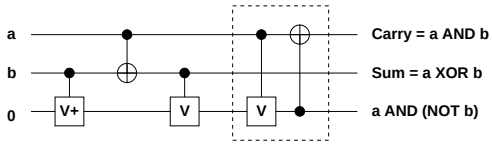

Figure 6: Half-adder with quantum cost = 4

Our results are summarized in Table 1. The "Prior" and "Our" columns indicate the best published quantum cost in previous literature and our synthesized quantum cost respectively. Our result for Miller's gate [15] is (cost=6) better than prior result (cost=7) [15, 25]. For the Fredkin [7], Peres [18] and Toffoli [1, 20] gates, our synthesized results have the same quantum costs as reported in prior literature [21, 25]. But nobody was able to show that the cost was minimum until now. In the past, people have been synthesizing the 2-bit adder using a Toffoli gate and an XOR gate (total cost of 6) [6, 9]. Our method proved that the minimum quantum cost is actually 4, as shown in Fig. 6.

Recent papers [13,16] used two Toffoli gates and two Feynman gates to implement a full-adder (cost=12). We proved that the minimum quantum cost for a full-adder is 6, shown in Fig. 7. To shorten the CPU runtime, we used a two-stage strategy (Section 5), and obtained a cost of 9 (Fig. 8). The CPU runtime is significantly reduced (from 7 hours to 140.83 seconds). Notice that the cost of this implementation can be reduced to 8 if we choose to omit the "propagate" logic (the last XOR gate).

We have also tried BDD based model checking [14], but the computation depends largely on variable ordering [10]. It is not as efficient as SAT based model checking.

7. CONCLUSION

We applied invariant checking, a formal verification technique, to the synthesis of quantum logic circuits. We reduced problems in quantum logic synthesis to those of multiple-valued logic synthesis, thus simplifying the search space and algorithm complexity. We created an optimal synthesis method and a speedup method with non-optimal quantum cost. Both our methods are guaranteed to synthesize the circuit. Our optimal synthesis method created minimum cost quantum circuits for Miller's gate, half-adder, and full-adder, which are better than previous results. We also proved the minimum quantum cost (using our elementary quantum gates) for Fredkin, Peres, and Toffoli gates. Our work is the first successful application of satisfiability with formal methods in quantum logic synthesis.

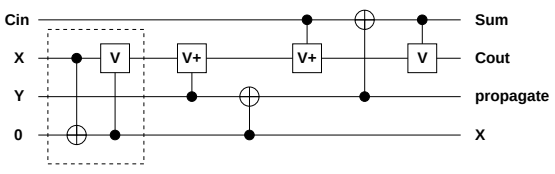

Figure 7: Full-adder with quantum cost = 6

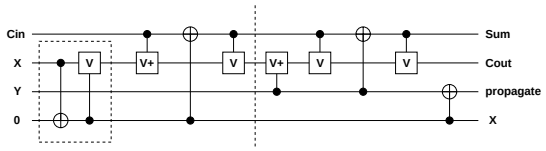

Figure 8: Full-adder with quantum cost = 9

8. REFERENCES

[1] A. Barenco et al. Elementary gates for quantum computation. *Physical Review A*, 52(5):3457–3467, 1995.

[2] A. Biere et al. Symbolic Model Checking using SAT procedures instead of BDDs. In *Proc. DAC*, 1999.

[3] R. E. Bryant. Graph-based algorithms for boolean function manipulation. *IEEE Trans. Computers*, 35(8), August 1986.

[4] F. Copty et al. Benefits of Bounded Model Checking at an Industrial Setting. In *Proc. Computer-Aided Verification*, 2001.

[5] D. Deutsch. Quantum computational networks. *Royal Society of London Series A*, 425:73–90, 1989.

[6] A. Ekert et al. Basic concepts in quantum computation. In *Coherent atomic matter waves*, volume 72 of *Lectures Notes in Les Houches Physics School*. Springer, August 1999.

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

[8] E. Goldberg and Y. Novikov. BerkMin: A Fast and Robust SAT Solver. In *Design Automation and Test in Europe (DATE)*, pages 142–149, 2002.

[9] J. Gruska. *Quantum Computing*. Osborne McGraw-Hill, April 1999.

[10] W. N. N. Hung et al. BDD Minimization by Scatter Search. *IEEE Trans. CAD*, 21(8), August 2002.

[11] W. N. N. Hung et al. Segmented Channel Routability via Satisfiability. *ACM Trans. Design Automation*, July 2004.

[12] K. Iwama et al. Transformation rules for designing CNOT-based quantum circuits. In *Proc. DAC*, 2002.

[13] A. Khlopotine et al. Reversible logic synthesis by iterative compositions. In *Int. Workshop on Logic Synthesis*, 2002.

[14] K. L. McMillan. *Symbolic model checking*. Kluwer Academic Publishers, 1993.

[15] D. M. Miller. Spectral and two-place decomposition techniques in reversible logic. In *Proc. IEEE Midwest Symp. Circuits and Systems*, August 2002.

[16] D. M. Miller et al. A transformation based algorithm for reversible logic synthesis. In *Proc. DAC*, 2003.

[17] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information*. Cambridge Univ. Press, 2000.

[18] A. Peres. Reversible logic and quantum computers. *Physical Review A*, 32:3266–3276, 1985.

[19] V. V. Shende et al. Reversible logic circuit synthesis. In *Proc. ICCAD*, 2002.

[20] T. Sleator and H. Weinfurter. Realizable universal quantum logic gates. *Physical Review Letters*, 74(20):4087–4090, 1995.

[21] J. A. Smolin and D. P. DiVincenzo. Five two-bit quantum gates are sufficient to implement the quantum Fredkin gate. *Physical Review A*, 53:2855–2856, 1996.

[22] X. Song et al. Board-level multiterminal net assignment for the partial cross-bar architecture. *IEEE Trans. VLSI*, 11(3):511–514, June 2003.

[23] X. Song et al. Algebraic characteristics of reversible gates. *Theory of Computing Systems*, 2004.

[24] L. M. K. Vandersypen et al. Experimental realization of Shor's quantum factoring algorithm using nuclear magnetic resonance. *Nature*, 414:883–887, Dec 2001.

[25] G. Yang et al. Majority-Based Reversible Logic Gate. In *Proc. Int. Symp. Representations and Methodology of Future Computing Technologies (RM2003)*, March 2003.