Classical Symbolic Retrodictive Execution of Quantum Circuits

Jacques Carette, McMaster University
Gerardo Ortiz*, Indiana University
Amr Sabry, Indiana University

February 20, 2022

Retrodictive quantum theory [3], retrocausality [1], and the time-symmetry of physical laws [10] suggest that partial knowledge about the future can be exploited to understand the present. We demonstrate the even stronger proposition that, in concert with the computational concepts of *demand-driven lazy evaluation* [7] and *symbolic partial evaluation* [6], retrodictive reasoning can be used as a computational resource to dequantize some quantum algorithms, i.e., to provide efficient classical algorithms inspired by their quantum counterparts.

Symbolic Execution of Classical Programs Applied to Quantum Oracles. A well-established technique to simultaneously explore multiple paths that a classical program could take under different inputs is *symbolic execution* [2, 4, 5, 8, 9]. In this execution scheme, concrete values are replaced by symbols which are initially unconstrained. As the execution proceeds, the symbols interact with the program constructs and this typically introduces constraints on the possible values that the symbols represent. At the end of the execution, these constraints can be solved to infer properties of the program under consideration. The idea is also applicable to quantum circuits as the following example illustrates.

Let $[\mathbf{n}]$ denote the finite set $\{0, 1, \ldots, (n-1)\}$. In Simon's problem, we are given a 2-1 (classical) function $f : [\mathbf{2^n}] \rightarrow [\mathbf{2^n}]$ with the property that there exists an a such $f(x) = f(x \oplus a)$ for all x; the goal is to determine a. The circuit in Fig. 1 implements the quantum algorithm when $n = 2$ and $a = 3$. In the circuit, the gates between barrier (1) and barrier (2) implement the quantum oracle $U_f(x, 0) = (x, f(x))$ that encapsulates the function f of interest. A direct classical simulation of the quantum circuit would need to execute the U_f block four times, once for each possible value $|00\rangle$, $|01\rangle$, $|10\rangle$, and $|11\rangle$ for the top two wires. Instead, let us introduce two symbols x_0 representing the top wire and x_1 representing the wire below it, and let's proceed with the execution symbolically. The state at barrier (1) is initially $|x_0x_100\rangle$. At the first CX-gate, we symbolically calculate the result of the target wire as $x_0 \oplus 0 = x_0$ evolving the state to $|x_0x_1x_00\rangle$. Going through the next three CX-gates, the state evolves as $|x_0x_1x_0x_0\rangle$, $|x_0x_1(x_0 \oplus x_1)x_0\rangle$, and $|x_0x_1(x_0 \oplus x_1)(x_0 \oplus x_1)\rangle$ at barrier (2). At that point, we have established that the bottom two wires are equal; the result of their measurement can only be 00 or 11. Since the function is promised to be 2-1 for all inputs, it is sufficient to analyze one case, say when the measurement at barrier (3) produces 00. This measurement collapses the top wires to $|x_0x_1\rangle$ subject to the constraint that $x_0 \oplus x_1 = 0$ or equivalently that $x_0 = x_1$. We have thus inferred that both $x_0 = x_1 = 0$ and $x_0 = x_1 = 1$ produce the same measurement result at barrier (3) and hence that $f(00) = f(11) = f(00 \oplus 11)$ which reveals that a is 11 in binary notation.

Since the quantum circuit between barriers (1) and (2) is reversible, we can perform a mixed predictive and retrodictive symbolic execution to make the flow of information conceptually clearer. We start a forward classical simulation with one arbitrary state at barrier (1), say $|0100\rangle$. This state evolves to $|0100\rangle$, then $|0100\rangle$ again, then $|0110\rangle$, and finally $|0111\rangle$. In this case, the result of measuring the bottom two wires is 11. Having produced a possible measurement at barrier (3), we start a retrodictive execution to find out what other input states might be compatible with this future measurement. To that end, we execute the circuit backwards with the symbolic state $|x_0x_111\rangle$; that execution evolves to $|x_0x_11(1 \oplus x_1)\rangle$, then $|x_0x_1(1 \oplus x_1)(1 \oplus x_1)\rangle$, then $|x_0x_1(1 \oplus x_1)(1 \oplus x_0 \oplus x_1)\rangle$, and finally $|x_0x_1(1 \oplus x_0 \oplus x_1)(1 \oplus x_0 \oplus x_1)\rangle$. At this point, we must reconcile the constraints with the initial condition of the bottom two wires which are

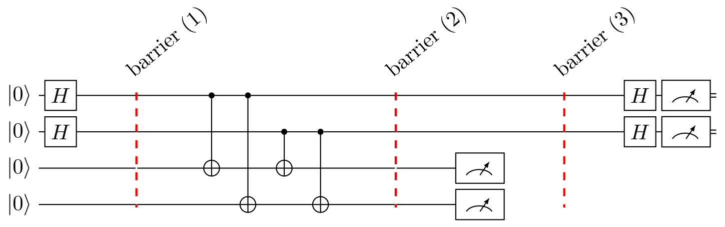
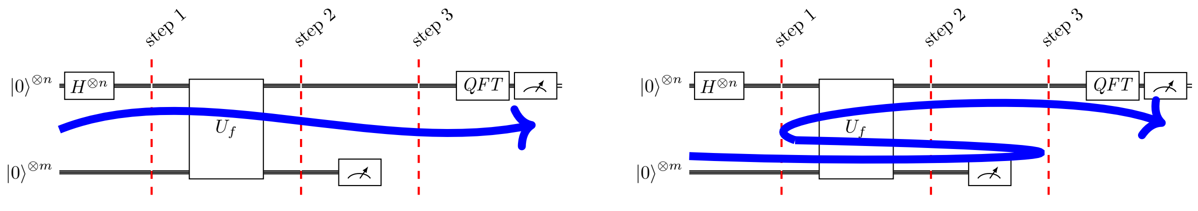


Figure 1: Circuit for Simon's Algorithm $n = 2$ and $a = 3$



(a) Conventional Flow

(b) Retrodictive Flow

Figure 2: Template quantum circuit

both $|0\rangle$. We conclude that $1 \oplus x_0 \oplus x_1 = 0$ or equivalently that $x_0 \neq x_1$. We have discovered that the measurement at barrier (3) is consistent with not just the state $|01\rangle$ we started with but also with the state $|10\rangle$. In other words, we have $f(01) = f(10) = f(01 \oplus 11)$ and the hidden value of a is revealed to be 11.

The example generalizes to many quantum algorithms that are expressed using circuits consisting of three stages: preparation, unitary evolution, and measurement in the Hadamard / Fourier basis as shown in Fig. 2(a). In the conventional execution model of quantum circuits, which is the conventional way to use quantum mechanics as a predictive theory, the U_f block receives both inputs and evolves in the forward direction to produce the outputs. Retrodictive reasoning suggests more creative ways to execute the U_f block as shown in Fig. 2(b). In this model, a forward classical execution is performed to determine a possible measurement result for the bottom register; using this information, a retrodictive classical execution is performed to determine the initial states of the first register that are consistent with this measurement. These states are then propagated forward to the final group of Hadamard / Fourier block.

Representing Wavefunctions Symbolically. In the quantum circuit for the Simon problem instance in Fig. 1, the Hadamard gates are only applied to control wires of CX-gates, never to wires that serve as targets of control gates. This property is actually common in many quantum algorithms and allows a rather simple symbolic representation of the wave function. An initial state $|0\rangle$ followed by Hadamard will be represented by a symbol x represented as unknown boolean value (instead of the superposition $(1/\sqrt{2})(|0\rangle + |1\rangle)$). Fig. 3 shows a circuit to generate the Bell state $(1/\sqrt{2})(|00\rangle + |11\rangle)$. By using the symbol x for the value of the top wire after the Hadamard gate, the input to the CX-gate is $|x0\rangle$ which evolves to $|xx\rangle$. The latter state, by sharing the same symbol in two positions, accurately represents the entangled Bell state.

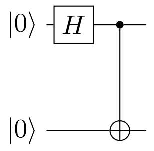


Figure 3: Bell State

we have enough to analyze many quantum algorithms; say we have an implementation and show result of running on many circuits

do communication protocols too ??

then get to $|-\rangle$

now to Bell, GHZ, and other entangled states graph state: H,H,CZ

00 00 01 01 10 10 11 -11

H control +/- distinction not important so use one class of vars H target +/- distinction important; use two classes of vars

References

[1] Yakir Aharonov and Lev Vaidman. “The Two-State Vector Formalism: An Updated Review”. In: *Time in Quantum Mechanics*. Ed. by J.G. Muga, R. Sala Mayato, and Í.L. Egusquiza. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 399–447.

[2] Roberto Baldoni, Emilio Coppa, Daniele Cono D’elia, Camil Demetrescu, and Irene Finocchi. “A Survey of Symbolic Execution Techniques”. In: *ACM Comput. Surv.* 51.3 (May 2018).

[3] Stephen M. Barnett, John Jeffers, and David T. Pegg. “Quantum Retrodiction: Foundations and Controversies”. In: *Symmetry* 13.4 (2021).

[4] Robert S. Boyer, Bernard Elspas, and Karl N. Levitt. “SELECT—a Formal System for Testing and Debugging Programs by Symbolic Execution”. In: *SIGPLAN Not.* 10.6 (Apr. 1975), pp. 234–245.

[5] Lori A. Clarke. “A Program Testing System”. In: *Proceedings of the 1976 Annual Conference*. ACM ’76. Houston, Texas, USA: Association for Computing Machinery, 1976, pp. 488–491.

[6] Yoshihiko Futamura. “Partial computation of programs”. In: *RIMS Symposia on Software Science and Engineering*. Ed. by Eiichi Goto, Koichi Furukawa, Reiji Nakajima, Ikuo Nakata, and Akinori Yonezawa. Berlin, Heidelberg: Springer Berlin Heidelberg, 1983, pp. 1–35.

[7] Peter Henderson and James H. Morris. “A Lazy Evaluator”. In: *Proceedings of the 3rd ACM SIGACT-SIGPLAN Symposium on Principles on Programming Languages*. POPL ’76. Atlanta, Georgia: Association for Computing Machinery, 1976, pp. 95–103.

[8] William E. Howden. “Experiments with a symbolic evaluation system”. In: *Proceedings of the National Computer Conference*. 1976.

[9] James C. King. “Symbolic Execution and Program Testing”. In: *Commun. ACM* 19.7 (July 1976), pp. 385–394.

[10] Satosi Watanabe. “Symmetry of Physical Laws. Part III. Prediction and Retrodiction”. In: *Rev. Mod. Phys.* 27 (2 Apr. 1955), pp. 179–186.