An Introduction to Quantum Cellular Automata

Joshua Horowitz

December 17, 2008

1 Introduction

Computer scientists have invented a plethora of fascinating machines to study. They are called computational models, and they come in many strange and varied shapes: Turing machines, register machines, circuits, finite automata, and many more.[1] When people began wondering what the computational potential of quantum physics was, they needed *quantum*-computational models, and the most straightforward way to get these was to take one of the old, classical computational models and modify it with quantum weirdness. As could be expected, this game of "quantization" (putting 'Q's in front of things) has become a commonplace part of quantum computer science.

Some quantized models have attained more prominence than others. For instance, the quantum circuit model has become standard for discussing issues of computational complexity, and the importance of Turing machines as a basis for classical computation has carried over to an interest in quantum Turing machines. But alternatives certainly exist, and each provides a new, interesting perspective on how computation works in a quantum universe.

In this report, I will discuss the *quantum cellular automaton* (QCA), a model which is exciting not only for its simple and elegant structure, but also for its "physical" flavor. Quantum cellular automata were physically inspired, and can thus shed light on the behavior of physical systems, and conversely, they carry the hope of providing a physical implementation of quantum computation.

Furthermore, just as a large class of classical computational models have convergent capabilities, making the concepts of "computable" and "efficient" much more universal than would be expected, quantum cellular automata appear to fit in with the rest of the quantum computational models: they are quantum-computationally universal. As such, quantum cellular automata encompass the whole universe of quantum computation, and one can be sure that studying them will shed light on the entirety of the field.

2 Classical Cellular Automata

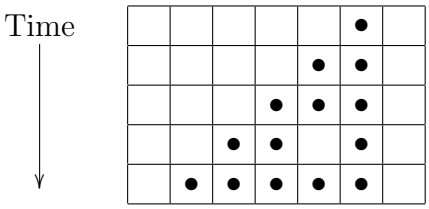
To start with, what is a classical (non-quantum) cellular automaton? It is a very simple computer which stores data on a tape[2] and updates every cell on the tape simultaneously through some local

[1] Wikipedia lists over 90 articles in its category "Computational models" [6].

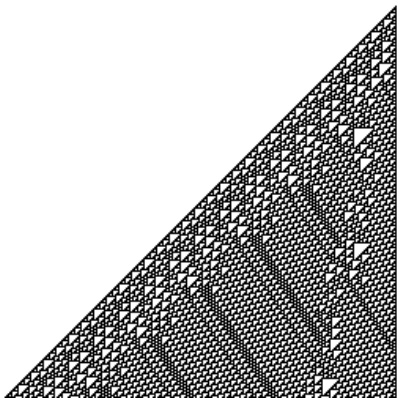
[2] It is certainly possible to define cellular automata on higher-dimensional grids (or even more general topologies), but the one-dimensional case is powerful and rich enough for this short introduction.

update rule. For example, we can make a cellular automaton with two cell states, ☐ and ●, and the update rule:

What this means is that, at every time step, a cell "looks" at its left and right neighbors and decides, based on its own state and its neighbors' states, what its state in the next time step will be. We can "run" this rule, starting with an almost-blank tape:



Starting with the single marked cell, a pattern spreads to the left. If we continued running the rule, a complicated pattern would emerge on our space-time diagram, full of triangles in strange formations:[3]



One way of thinking about a cellular automaton is that it is a "powerfully stupid" Turing machine; stupid, because there is no centralized head keeping track of a state in a program, but powerful, because the entire tape performs computational work at once. However you think these two forces balance one another, it is still surprising to learn that the above two-state CA (known as Rule 110) is in fact computationally universal. By programming it with some easily-computable input, it will perform any computation that a Turing machine can. Thus, cellular automata fit into the grand class of computational models which, though they appear different on the surface, have the exact same computational power.

It would be silly, however, to just forget about cellular automata as a result of this deep equivalence, for cellular automata have unique characteristics which distinguish them from other models and give us good reason to study them. As I mentioned before, one of these is their

[3]Image from [7].

distinctly "physical" flavor. The update rule is local, and homogenous across the tape, much like a law of physics should be local and homogenous across space. For this reason, there has been much interest in modeling physical systems with cellular automata. Conversely, if one could find a physical mechanism which effectively performed the update rule of a cellular automata, one could use this mechanism to implement the automaton in the real world, and perform actual computations.

This discussion becomes all the more exciting when one realizes that the physical laws of our universe are in fact quantum in nature. One is confronted with the question: How would a quantum cellular automaton work?

3 Quantum Cellular Automata

People often trace the history of quantum computing to a famous 1982 paper by Feynman entitled "Simulating physics with computers" [2]. It is interesting to note that, in this inaugural paper, Feynman mentioned quantum cellular automata as a possible way to make a universal quantum computer. The discussion was casual, however, and it took over ten years for the concept to be formalized.

One of the first people to write down a real mathematical definition of a QCA was John Watrous, in a 1995 paper [5]. Since then, several alternative definitions have been given (to fix various unpleasant attributes of the model) but Watrous's QCA is still the most straightforward, and an excellent way to introduce the concept.

The basic idea behind Watrous's definition was to replace classical, deterministic transitions with quantum amplitudes. That is, instead of the rule $\boxed{\bullet}\boxed{\bullet}\boxed{\ } \longrightarrow \boxed{\bullet}$, we might have $\boxed{\bullet}\boxed{\bullet}\boxed{\ } \longrightarrow \sqrt{\tfrac{1}{3}}\boxed{\ } + \sqrt{\tfrac{2}{3}}\boxed{\bullet}$. We can formalize this, of course: Before, the (classical) update rule could be thought of as a map:

$$\delta : \underbrace{Q}_{\text{left}} \times \underbrace{Q}_{\text{old}} \times \underbrace{Q}_{\text{right}} \longrightarrow \underbrace{Q}_{\text{new}}.$$

(Here, Q is the set of cell states.) Now, the (quantum) update rule assigns amplitudes to every possible transition from old (and neighbors) to new:

$$\delta : \underbrace{Q}_{\text{left}} \times \underbrace{Q}_{\text{old}} \times \underbrace{Q}_{\text{right}} \times \underbrace{Q}_{\text{new}} \longrightarrow \underbrace{\mathbb{C}}_{\text{amplitude}}.$$

Based on these local transition amplitudes, we can compute global transition amplitudes from any given tape configuration to any other tape configuration:

$$\Delta(a, b) = \prod_i \delta(a_{i-1}, a_i, a_{i+1}, b_i).$$

Using these amplitudes, we have a single-time-step-evolution operator on the Hilbert space with tape configurations as basis elements:

$$T\,|a\rangle = \sum_b \Delta(a, b)\,|b\rangle\,.$$

This straightforward process has given us the global time-evolution operator (on the Hilbert-space extension of the set of classical tape configurations) from the local transition amplitudes. But we've moved very quickly, and missed an important problem. For this model to make any sense at all, the global time-evolution operator T must be unitary. Every quantum system must evolve according to some unitary transformation, and besides, being unitary is one of the best ways to guarantee that normalized states will go to normalized states (so we can interpret squared amplitudes as probabilities which sum to 1).[4] But in general, there is no reason for us to expect that T will be unitary. Trivially, we could have set δ to always be 0, giving $T = 0$. One might think that a simple normalization requirement on δ (like $\sum_q \delta(q_1, q_2, q_3, q)^2 = 1 \ \forall q_1, q_2, q_3$) would be enough to guarantee that T be unitary, but it is regrettably not that simple.

Watrous has a two-part solution to this problem. The first part is sort of a cop-out. It is to call QCAs which have δs giving rise to unitary Ts *well-formed* QCAs, and leave the question of determining whether a given QCA is well-formed or not to whoever wants to use that QCA. Fortunately, a few years after Watrous's paper, a polynomial-time algorithm was discovered which checks whether a QCA is well-formed or not (in our situation, it would run in $O(n^2)$ time) [1]. Still, the question of which QCAs are well-formed is a complex and subtle one.

The other part of Watrous's solution was to find a sufficiently large and interesting class of QCAs for which checking well-formedness is easier than in general. In particular, Watrous studied a creature called a *partitioned* quantum cellular automaton. As evidence that these do indeed form a sufficiently large and interesting class, he proved that they are, in fact, universal for quantum computation. Let us see how this works.

4 Partitioned QCAs and Universality

In a partitioned QCA, we think of the state of each cell as being made up of the states of three (somewhat imaginary) *subcells*. Here's a single cell with the state (α, β, γ):

$\in Q$		
α	β	γ
$\in Q_L$	$\in Q_M$	$\in Q_R$

$$Q = Q_L \times Q_M \times Q_R$$

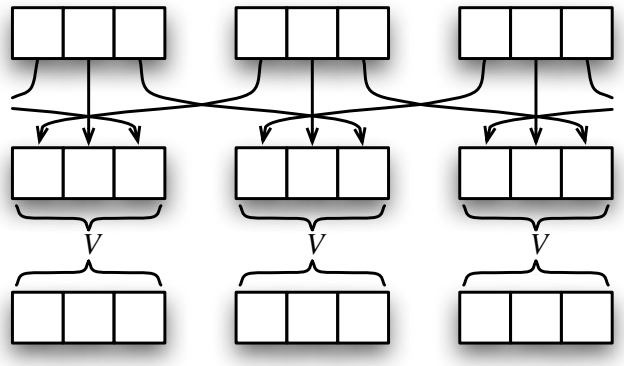
This alone means nothing; we're just writing our cells' states as tuples. The real thing which distinguishes a PQCA is that we restrict how its update rule can work. In particular, we demand that transition amplitudes depend only on

- the right substate of the left neighbor,
- the middle substate of the self, and
- the left substate of the right neighbor.

To anthropomorphize, a cell can only "see" the right side of its left neighbor, the middle of itself, and the left side of its right neighbor. Perhaps the inability of a cell to see its own facades fits in nicely with the social analogy.

[4] There is a point of possible confusion here: Since we have an infinite-dimensional Hilbert space, there are operators which preserve the norm without being unitary.

We can think of the action of a PQCA's update rule as a permutation of subcells, followed by some sort of cell-wise transformation V. This is illustrated in the following diagram:[5]



Now it is easy to see why a PQCA's well-formedness is simple to determine. What we have here is a quantum circuit. Every step of the PQCA's operation consists of a permutation of registers, followed by disjointly-acting three-register operations. If the three-register operation is unitary, the whole time-step operation will be unitary, and visa versa. Thus, checking well-formedness of a PQCA just comes down to checking whether the matrix determining the transition amplitudes is unitary.[6]

So PQCAs are a particularly nice and easy-to-analyze subclass of QCAs. The question is, can they do anything? As I mentioned before, the answer is that they can in fact to *anything*. To be exact, Watrous proved that, given any quantum Turing machine M, you can easily construct a PQCA which simulates M and gives the same answers to all inputs, taking at most a constant amount more steps than M to do so.

The proof is surprisingly straightforward. The structure of the Turing-machine simulation is reflected in the structure of the cell partitioning:

- Left subcells store left-moving heads (with their head-states).
- Middle subcells store the Turing machine's tape markings.
- Right subcells store right-moving heads (with their head-states).

Then the restriction on the PQCA's update rule gives us exactly as much freedom as we need to update the tape corresponding to one step of the QTM's operation (that is, to move the head, possibly change the head-state, and possibly change the tape marking on the cell it moves into). The amplitudes for various QTM transitions directly become the amplitudes for various PQCA transitions.

So PQCAs are at least as powerful as QTMs. Are they any more powerful? Watrous proved in his paper that the answer is no. Just as PQCAs can simulate QTMs, QTMs can simulate PQCAs.

[5]Diagram from [3].

[6]Thinking about QCAs in terms of quantum circuits helps to explain how it is that QCAs can be poorly-formed: Drawing out the circuit diagram for a general QCA requires that you use fan-out (branching wires), which is not allowed by the no-cloning theorem. A QCA will only be well-formed only if these fan-outs turn out to be unneeded, which is not necessarily easy to determine.

Furthermore, they can do so with at worst linear slowdown, so PQCAs and QTMs are equivalent in terms of both computation *and* complexity.

However, this equivalence was proven only between QTMs and the restricted class of *partitioned* QCAs. The question of whether non-partitioned QCAs are any more powerful than QTMs is undetermined.

5 Further Directions

As I mentioned, Watrous's model suffers from some genuine flaws. The lack of a general guarantee of well-formedness makes them difficult to work with. Furthermore, they lack many convenient properties of composability and invertibility. For these and other reasons, other models have been presented in the literature. They generally invoke more sophisticated machinery, such as homomorphisms of quasi-local C^*-algebras in the Heisenberg picture [4]. The theories around these more advanced models are certainly rich, and it remains to be seen what theory, if any, might serve to be useful in a physical implementation of quantum cellular automata.

Several possibilities for physical implementation have also been raised. Links have been found between QCA models and lattice gasses, spin chains, and quantum-dot configurations. Clearly, none of these connections have yet materialized into scalable universal quantum computers. If they had, said fact would probably be fairly well-publisized by now. But they remain intriguing candidates in the search.

Some scientists have controversially claimed that the universe is, at some fundamental level, a cellular automaton. Ironically, they generally try to brush aside the quantum nature of this fundamental reality, claiming that quantum phenomena can somehow come out of lower-level classical phenomena (contradicting several mathematical arguments that this is impossible). Perhaps people would take these computational physicists more seriously if they claimed the universe was a *quantum* cellular automata.

References

[1] C. Dürr, M. Santha, "A Decision Procedure for Unitary Linear Quantum Cellular Automata", SIAM Journal on Computing, v.31 n.4, p.1076-1089, 2002.

[2] R. Feynman, "Simulating physics with computers", Int. J. Theor. Phys. 21, 1982: pp. 467488.

[3] C. Pérez-Delgado and D. Cheung, "Local Unitary Quantum Cellular Automata", Phys. Rev. A 76, 032320, 2007.

[4] B. Schumacher and R. Werner, "Reversible quantum cellular automata", quant-ph/0405174.

[5] J. Watrous, "On one-dimensional quantum cellular automata", Proc. 36th FOCS, 1995: pp. 528537.

[6] `http://en.wikipedia.org/wiki/Category:Computational_models`.

[7] `http://upload.wikimedia.org/wikipedia/commons/f/fa/CA_rule110s.png`.