Ubiquity Symposium

What is Computation?

Computation and Fundamental Physics

by Dave Bacon, University of Washington

Editor's Introduction

In this seventh article in the ACM Ubiquity symposium, What is Computation?, Dave Bacon of University of Washington explains why he thinks discussing the question is as important as thinking about what it means to be self-aware.

Peter J. Denning
Editor

Ubiquity Symposium

What is Computation?

Computation and Fundamental Physics

by Dave Bacon, University of Washington

No matter how you slice it, computers have changed our lives. When we say this, what immediately comes to mind is the way in which ubiquitous computer technology has transformed our communicating, working, playing, and (witness online dating sites) even our loving. Another interpretation of "how computers have changed our lives" is the possibility that computation itself is essential to even defining who we are. This is the main idea behind "strong artificial intelligence," where one argues that human brains are nothing more than computing machines. Or, as the Post-It note I had on my wall in graduate school pithily read: "Either 1) computers will take over the world or 2) they already have." If strong artificial intelligence is true, then the question "What is computation?" is important to defining what it means to be intelligent and self-aware. Thus defining computation would seem relevant to a very fundamental question in the study of the mind.

Here, I would like to push this argument one step further and argue that the question "What is computation?" is not mere taxonomy, nor simply an aid in our anthropocentric obsession with explaining our thinking process, but instead is relevant to some of the deepest questions in the nature of the physics of our universe. Understanding what computation is may be more essential to us than we might have imagined—in fact, it may be tied to the very fabric of how nature works!

The Reliability of Computation

The starting point for this speculative point of view is the reliability of computation. In 1937, when Alan Turing formalized the notion of computation that we now call the Turing machine (Turing, 1937), one remarkable (but often overlooked) fact was that the feasibility of a machine that acted reliably enough to behave like the infinitely extendable machine Turing conceived, was not at all a technological given. When the first general purpose computer, the ENIAC, was constructed in the mid 1940s, some argued that the machine would not work because the

vacuum tubes it was made up would fail at too high of a rate. Indeed the ENIAC did have a tube failure about once every two days (Randall, 2006). Imagining how one could build larger, and more reliable computers was not at all obvious. The invention of the transistor in 1948 and the subsequent invention of the integrated circuit in 1958, solved these problems, which then led to an amazing half century of ever increasing computing power as described by Moore's law. Today, however, you may have noticed that computer clock speeds have stopped increasing and, while transistor sizes continue to shrink, we are fast approaching the limit where a single switch in a computer is carried out by only a few atoms. And when you get down to such small atomic systems, the problem of reliable computation from unreliable parts again begins to rear its head. Which leads us to ask, "Why is computation possible at all?" Our generous universe comes equipped with the ability to compute, but how and why is this?

The history of the understanding how unreliable devices can be made reliable is made up of at least two main directions. The first relies on a fundamental theoretical proof while the second, in contrast, is relevant to how modern computers work in practice. For the first direction, the seminal work is that of John von Neumann (von Neumann, 1956). Von Neumann considered a very simple model of unreliable computation in which every component of a logical circuit could fail to operate properly with some independent probability. He then asked the question of whether such a logical circuits could be engineered into a device that failed with vanishingly small probability. Such a device would then be considered as enacting a "robust computation." At first such a result may seem hopeless—if you are relying on the output of a single bit in the logical circuit then there is always a non-zero probability that the logic gate that outputs this last bit may fail. To circumvent this seemingly hopeless situation, von Neumann used the simple coding strategy, a la Claude Shannon's information theory (Shannon, 1948), of redundancy. Instead of placing the information into a single fragile bit, we instead take a bundle of bits and "interpret" them as being either zero or one depending on whether the vast majority of bits in this bundle are zero or one. This is one of the central tenets of dealing with unreliable bits: don't. Instead deal with lots of unreliable bits and "reinterpret" what you call your information.

The Restorative Organ

Just encoding information is not enough to obtain reliable computation: encoding may prolong the lifetime of a bit, but eventually the encoded information will be destroyed. To overcome this, von Neumann introduced a "restorative organ" that served to take encoded information with some erred bits and fix as many of those bits as possible. Of course the logic gates he used for this restorative organ were themselves subject to failure, so it is not at all obvious that this is possible. Further complicating this picture one must be able to compute on the encoded

information in a way that also does not add too many errors to the encoded information. But get around these obstacles von Neumann did, and in the end he produced a theorem out of it: If the rate of error of your logical gates is below a certain threshold value, then a robust computation can be enacted with a failure of probability as small as you like using only a few more gates (logarithmically more gates as a function of one over the error probability).

Von Neumann's constructions are all fine and dandy if you are a theoretician, but in the real world, it doesn't seem as if we need these constructions, or at least it doesn't seem that we need them for devices such as our silicon based integrated circuits. This is the second approach to robust computing: how it occurs in practice. Hard disks, to take an example, use some minimal error correction, but not so much that it dominates the hardware of the device. How is this possible? The answer to how our real world devices achieve robust computation is perhaps best summed up by a phrase coined by the lat Rolf Landauer: "information is physical" (Landauer, 2002). By this expression, Landauer did not mean that information itself is a basic construct of physics, but instead that information really only exists when the physics of a device allows it. Another way to put it is that not all physical systems are suitable for computation. This is of course, quite obvious, but imagine this question from the perspective of a physicist: what are the physical mechanisms that allow for a device to be a computer? How does one take a physical system and decide whether or not it is capable of robust computation?

Consider, for example, information stored in a hard drive. This information is encoded into the direction of the spins in a magnetic domain: whether the majority of spins in the domain are pointing up or down, say, gives a bit the value zero or one. Individual spins in these domains, however, are fragile: each individual spin is subject to noise from its environment that can act to change the direction of this spin. The domain is able to store information, however, by essentially using the two tricks of von Neumann. First, the information is not encoded into a single spin, but instead is spread out across many spins and the majority vote of these spins represents the information. Second the system essentially performs error correction: spins locally feel the direction of their neighbors and energetically favor alignment. Thus when the environment flips a spin, this requires some energy for violating neighborly concordance, and the flipped spin will quickly relax back to its neighbor's direction in order to minimize energy. As long as the environment does not induce too much noise, i.e., the system is not too hot, then the lifetime of the encoded majority of the spins is thus made much much longer than the lifetime of an individual spin. This is simply von Neumann's error correcting restorative organ enacted by the physics of the device.

Computation has a Lifetime

So what does this all have to do with the question at hand: "What is computation?" The first are the two realizations that computation has a lifetime and that in the wild it can exist in various states of digitization. For devices truly deserving the moniker computer, the lifetime of the information is longer than the tasks the device is designed to solve. This lifetime is defined as the time until the stored and computed on information is randomized. Similarly for devices that we want to call computers the digitization of information should be high. In our example of the hard-drive, the fact that one takes the majority vote of the spins along a particular direction is an example of a digitization resulting in one bit of digital information (note that physically spins can point along any continuous direction.) However, one could imagine digitizing the signal differently: for example by binning the total number of spins pointing along the up direction into four different bins one could encode two bits of digital information. But clearly there is a limiting point in which relying on single analog signals will run into the problem first enumerated by von Neumann: such single systems, if they fail with some fixed probability, fix the overall lifetime of the information.

But is there more to the question of "What is computation?" than just the above taxonomic clarifications? Here we will follow a speculative line of thought considered originally in a 1999 paper by Walter Ogburn and John Preskill (Ogburn & Preskill, 1999). The main subject of this paper was the field of topological quantum computation. Quantum computers are devices that manipulate information that exploits the counter-intuitive properties of quantum theory. Quantum computers offer an interesting example of the notions of robust computing: quantum information is even more fragile than classical information. Indeed when quantum computers were first shown to be able to outperform classical computers at certain tasks a critique of the field was that because quantum information is so fragile, no large-scale quantum computer could, in principle, ever be built. However, just as von Neumann showed that classical computing is robust with faulty devices, quantum computing theorists realized that a similar result could be obtained for quantum computers (Aharonov & Ben-Or, 1997) (Knill, Laflamme, & Zurek, 1998). Thus, in theory at least, quantum computers can be built assuming that the basic operations of a quantum computer can be performed with high enough fidelity. What Ogburn and Preskill were considering in their paper was a method for doing fault-tolerant quantum computing based upon prior work by Alexei Kiteav (Kitaev, 2003) which linked physical theories known as topological quantum field theories to fault-tolerant quantum computing. But what is remarkable about this paper is the final concluding section where the authors make a bold speculation.

Fundamental Physics

To understand this speculation, a little background from fundamental physics is necessary. Quantum theory is a foundational theory of physics. It sits, along with special relativity, as the base upon which the different physical theories rest. Thus one takes the physics of electrodynamics and puts it together special relativity and quantum theory to form quantum electrodynamics. So far physicists have figured out how to merge the physics of three fundamental forces, the electromagnetic force, the weak force, and the strong force with quantum theory and special relativity in what is known as the standard model of particle physics. But the fourth force we know exists, gravity, has resisted merging with quantum theory. This is the basic problem of quantum gravity. In the mid 1970s Stephen Hawking made a startling claim (Hawking, 1976) based upon extrapolating quantum theory into a regime where gravity is important. He claimed that if one merged quantum theory and gravity one was necessarily led to a situation in which information can be destroyed in a black hole. In contrast quantum theory, while allowing information to leak into an environment, does not allow for the quantum information to be explicitly destroyed. This was a radical suggestion and is the basis of what is now termed the "black hole information paradox."

Ogburn and Preskill noted that this idea, that information could be destroyed by processes at a black hole, might have a novel solution in light of the realization that classical and quantum computers could be built in spite of noise induced by an environment. In particular they noted that Hawking's result is particular odd because it implies that at very short timescales (faster than the so-called Planck time, about 5 times 10^{-44} seconds), where virtual black hole production is thought to be a dominating process, if Hawking is correct, then information must be destroyed in vast quantities. Yet at longer time scales we know that quantum theory without information loss appears to be correct. In order to reconcile this, Ogburn and Preskill suggested that perhaps nature itself is fault-tolerant. At short time scales, information is repeatedly being destroyed, but at longer scales, because of some form of effective encoding and error correction, non-information destroying quantum theory is a good approximation to describing physics. This was especially relevant to the subject of the paper, topological quantum computing, where a physical theory—that of topological quantum field theory—gave rise to models for robustly storing quantum information.

This is, in many respects, just another step down the line of argument that the universe is a computer of some sort. This point of view has been advocated by many researchers, most notably by Ed Fredkin (Fredkin, 1992), Stephen Wolfram (Wolfram, 2002) and Seth Lloyd (LLoyd, 2006). In many respects this point of view may be nothing more than a result of the fact that the notion of computation is the disease of our age—everywhere we look today we see examples of computers, computation, and information theory and thus we extrapolate this to

our laws of physics. Indeed, thinking about computing as arising from faulty components, it seems as if the abstraction that uses perfectly operating computers is unlikely to exist as anything but a platonic ideal. Another critique of such a point of view is that there is no evidence for the kind of digitization that characterizes computers nor are there any predictions made by those who advocate such a view that have been experimentally confirmed.

The notion that the universe is at its most fundamental level constantly destroying information, but that quantum theory holds at a larger length scales, is a variation on the theme of the universe as a deterministic digital computer. But it is one that has not been considered nearly as closely. And indeed, it seems possible that it could lead to predictions in a manner that the mere idea of the universe as a digital computer cannot. To understand how this might be, we note that if there is a digital component to our physics, then there is a timescale or length scale associated with this discreteness. Unfortunately there are quite restrictive bounds on what such a digitization can look like: many of our theories have been validated to very high precision. Of course one can always stick the digitization in at short enough timescales or small enough length scales, but there is no way to get at testable predictions beyond simply probing these shorter time scales with higher energy physics.

Fault Tolerance

But the case for fault-tolerant computing might be different. Indeed one of the important features of the models considered by Ogburn and Preskill is that they are in some way spatially and temporally local, and yet they provide some protection of quantum information. It is exactly this locality that means that they are easily meshed with our existing understanding of physics. Thus one can ask: Are there consequences of requiring that information destroying processes get alleviated at large scales by some local naturally fault-tolerant physics? While I don't know the answer, there is at least some interesting precedence that this could lead to novel predictions about fundamental physics.

One such precedence concerns fault-tolerant cellular automata. A cellular automaton is a construction where information is stored in spatially separated cells and these cells are updated at discrete time steps depending on the state of the cell and the state of the neighboring cells. As a model of a faulty cellular automaton one can imagine interleaving deterministic perfectly functioning cellular automaton updates with a process that randomly and independently adds noise to the cells. Another model of faults introduces cells that don't function at all (think "manufacturing defects"). Now here is the interesting point. If one considers a simple cellular automaton with the first kind of faults on a two dimensional square grid, then there is a simple

cellular automaton rule, Toom's rule (Toom, 1980), which can be used to robustly store information. Notably, however, when one adds manufacturing faults to this system, this tolerance to errors disappears (McCann, 2007). Indeed it has been shown that for any cellular automaton rule from a broad class of such rules, on a square grid there is no way to robustly store information. But, it turns out that if one changes the spatial layout of the cells, then a fault-tolerant cellular automaton tolerant to both types of errors can be constructed. What is the change necessary to achieve this? It is to work on grids that can be embedded without distortion into a hyperbolic two-dimensional space (McCann & Pippenger, 2008). In other words, it seems that a consequence of the hypothesis that nature is fault-tolerant in two spatial dimensions (for classical computation) is that space must be hyperbolically curved! In this way a consequence of the large scale geometry of the universe emerges from a requirement that a cellular automata robustly store classical information in the presence of probabilistic and manufacturing faults. Indeed, in this case, it seems to get the answer wrong as we currently believe that our universe is actually curved in the opposite sense: it is believed that we live in a universe dominated by a de Sitter cosmology, which is elliptically and not hyperbolically curved.

The larger point, however, is that the idea that a noisy, faulty, or information destroying small scale physics may result in consequences for our large scale physics without our ever having to probe the kinds of time scales at which these processes reveal themselves. Thus it is possible that, in contrast to the notion that the universe is a computer, that the notion that the universe is a faulty computer may make experimentally testable predictions, even if the level at which these faults show themselves is as far down as the Planck scale.

To the question "What is computation?" we have answered that, from the perspective of physics, computation is a property of only some physical systems and that the idealized form of perfectly digital computation is an extremely useful emergent property, but unlikely to be fundamental. This in turn led us to consider whether nature itself might actually be much more dirty and noisy than we normally think, and that the existence of computers should give us pause when understanding how quantum theory can exist in a physics dominated by the destruction of information. Provocatively we have even suggested that proving theorems about fault-tolerant local systems might lead to evidence for nature being fault-tolerant without an appeal to the details of the actual physics. Truly, then, computation may lie at the heart of our understanding of the workings of the universe.

About the Author

Dave Bacon is an assistant research professor in the department of computer science and engineering at the University of Washington, where he is also an adjunct professor in the department of physics.

Bibliography

Aharonov, D., & Ben-Or, M. (1997). Fault-tolerant quantum computation with constant error rate. *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing* (pp. 176-188). New York: ACM Press.

Fredkin, E. (1992). Finite Nature. *Proceedings of the XXVIIth Rencotre de Moriond .*

Hawking, S. W. (1976). Breakdown of Predictability in Gravitational Collapse . *Physical Review D, 14*, 2460.

Kitaev, A. (2003). Fault-tolerant quantum computation by anyons. *Annals of Physics , 303*, 2--30.

Knill, E., Laflamme, R., & Zurek, W. H. (1998). Resilent quantum computation. *Science, 279*, 342-345.

Landauer, R. (2002). Information is inevitably physical. In A. J. Hey, *Feynman and computation: exploring the limits of computers* (pp. 77-92). Boulder, CO: Persueus Books.

LLoyd, S. (2006). *Programming the Universe: A Quantum Computer Scientist Takes On the Cosmos.* Knopf.

McCann, M. (2007). Memory in media with manufacturing faults. *PhD thesis, Department of Computer Science, Princeton Universit*y.

McCann, M., & Pippenger, N. (2008). Fault tolerance in cellular automata at high fault rates. *Journal of Computer and System Sciences, 74*, 910–918.

Ogburn, R. W., & Preskill, J. (1999). Topological Quantum Computing. *Lecture Notes in Computer Science, 1509*, 341-356.

Randall, A. (2006, Feb 14). *Q&A: A lost interview with ENIAC co-inventor J. Presper Eckert.* Retrieved May 5, 2009, from ComputerWorld: http://www.computerworld.com/s/article/print/108568/Q_A_A_lost_interview_with_ENIAC_co_inventor_J._Presper_Eckert.

Shannon, C. (1948). A mathematical theory of communication. *Bell System Technical Journal, 27*, 379–423 .

Toom, A. L. (1980). Stable and attractive trajectories in multicomponent system. In R. L. Dobrushin, *Multicomponent Systems* (pp. 549-575). New York: Dekker.

Turing, A. M. (1937). On Computable Numbers, with an Application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society , 42*, 230-265.

von Neumann, J. (1956). Probabilistic logics and the synthesis of reliable organisms from unreliable components. In C. S. McCarthy, *Automata Studies* (pp. 43-98). Princeton, NJ: Princeton University Press.

Wolfram, S. (2002). *A New Kind of Science.* Wolfram Media, INC.

DOI: 10.1145/1895419.1920826

