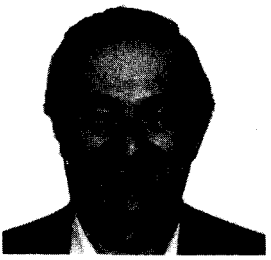
FINITE NATURE

EDWARD FREDKIN
DEPARTMENT OF PHYSICS
BOSTON UNIVERSITY
BOSTON, MA, 02215, USA



ABSTRACT

A fundamental question about time, space and the inhabitants thereof is "Are things smooth or grainy?" Some things are obviously grainy (matter, charge, angular momentum); for other things (space, time, momentum, energy) the answers are not clear. Finite Nature is the assumption that, at some scale, space and time are discrete and that the number of possible states of every finite volume of space-time is finite. In other words Finite Nature assumes that there is no thing that is smooth or continuous and that there are no infinitesimals. If finite nature is true, then there are certain consequences that are independent of the scale.

Introduction

The development of science, from ancient times to the present, has been a series of nearly unbroken steps where one concept after another has moved out of the shadows of doubt and uncertainty and into the light of accepted scientific fact. The atomic hypothesis[1], whether matter is made up of atoms, is only one of many *atomic* hypotheses. So far every such question, discrete versus continuous, about a property of our world either remains undecided or it has been decided as discrete (atoms, electricity, light, angular momentum, etc.). It is hard to imagine the proof that some property will never admit to a finite description, no matter how fine grained. On the other hand, what is interesting is that so many concepts once thought of as continuous are now accepted as discrete. Finite Nature assumes that that historical process will continue to a logical conclusion where, at the bottom, everything will turn out to be atomic or discrete, including space and time.

The prime implication of Finite Nature is that every volume of space-time has a finite amount of information in it. Every small region of space-time (a cell) must be in one of a small number of states. If the future state of that cell is always a function of the space-time neighbors of the that cell, then we are lucky because there is a branch of mathematics, Automata Theory[2], that deals with such systems. Automata Theory is concerned with the behavior of all systems that evolve from one state (out of a finite number of states) to another, on the basis of a rule that takes into account: (1) the current state and (2) an input state. In the case of physics the input state comes from the states of other cells in the space-time neighborhood. There are versions of Automata called Cellular Automata[3] (CA) that are just what we're discussing.

Once we understand the rule of some CA, we name the states. If there are 3 states per cell, we can call them "1, 2 and 3" or "a, b and c"; it doesn't matter. Given any rule, we can find other rules, with other numbers of states per cell, that evolve in a manner isomorphic to the original system. Strangely enough, there is a wonderful chain of logic that proves that a system with just 2 states per cell, the Bank's[4] rule, with a 2-D neighborhood (4 neighbors; north, south, east and west), starting from an appropriate initial condition, can evolve in a way isomorphic to any other CA; no matter how many states per cell, no matter how high the dimensionality of the space, no matter how complex the rule, no matter how the neighborhood is defined! Of course the Bank's rule may take more cells and more time to do the simulation, but it can do it exactly because it's universal.

Automata, Cellular Automata and Finite State Machines are all forms of computers. A computer is universal if it can be programmed to simulate any other target computer. The simulating computer must have enough memory for two things: (1) to represent the state of the target computer and (2) to hold a program that can operate on that state to mimic the behavior of the target computer. Every ordinary computer is universal. The pioneers of automata theory were interested in what could be done by machines with arbitrarily large memories, so a common but unnecessary interpretation of "universal computer" is a computer with an infinite memory. We extend the idea to include computers with finite memory. A Macintosh, can *exactly* simulate the behavior of the fastest super computer if it has only a little bit more disk memory than the super computer. We pay a lot more money for a super computer not for what it can do, but for how *fast* it does it.

Given Finite Nature, what we have at the bottom is a Cellular Automaton of some kind. The first thing to wonder about is "Is it Universal?" The normal way to show that a computer is universal is to demonstrate a program that can simulate a computer known to be universal. The fact that the laws of physics allow us to build ordinary universal computers is proof of what must be a very fundamental law of physics: "The fundamental process of physics is computation universal."

Uncertainty is at the heart of quantum mechanics. Finite Nature requires that we rule out true, locally generated randomness because such numbers would not, in this context, be considered finite. The reason is that there is no way to create, within a computer, a truly random number that is orthogonal to everything in the computer. On the other hand, another kind of randomness appears in CA where the values of many of the bits are so influenced by distant events as to be essentially orthogonal to any local process. The deterministic nature of finite digital processes is different in that it is unknowable determinism. From within the system an observer will never be able to know very much about the true microscopic state of that system. Every part of space is computing its future as fast possible, while information pours in from every direction. The result is the same as caused by the apparent randomness of quantum mechanical processes.

What is Happening at the Bottom

We imagine that the cells that underlie physics are very small. Lets assume that the distances between such cells is somewhere between a Fermi, 10^{-13}cm, and Planck's length; $\sqrt{(\hbar G/c^3)} = 1.6 \times 10^{-33}$cm. If it's down near Planck's length, particles such as electrons, would be enormous entities consisting of structures spread out over volumes

containing more than 10^{60} bits. The momentum information would be spread out over volumes greater than 10^{75} bits. Because of superposition only a tiny fraction of the information in that volume would actually represent the momentum of the particle. While it is difficult to give strong arguments as to what the scale should be, many interesting consequences are independent of the scale. We call such informational models of physics "Digital Mechanics". At first glance it seems that Digital Mechanics[5] must be at odds with things we know about physics. However, efforts to reconcile the two make steady progress.

The utility of Digital Mechanics rests on the question of Finite Nature. If Finite Nature is true, then we may be able to derive QM from Digital Mechanics. If Finite Nature is false, then QM cannot be derived from Digital Mechanics. If Digital Mechanics ever becomes a successful model, it might allow for the development of a purely mechanistic, deterministic substrate for Quantum Mechanics.

The advantage of Digital Mechanics is that simple universal digital systems can do anything that can be done by any digital system. This is also the disadvantage. It is hard to think about the properties of the members of a class when each member can do everything. The field of Computer Science has very few examples of useful or meaningful analytic solutions as to what some digital system will or won't do. On the contrary, there is a celebrated proof that, in general, there are no analytical shortcuts that can tell the future state of some general computation any quicker than doing the computation step by step (this is the so called "halting problem" for Turing Machines[6]). There are normally no solutions in closed form. There is not yet any good hierarchy of concepts that express complex behavior in terms of simpler behavior, as is done in physics.

On the other hand, good programmers synthesize programs that are hierarchical. Programmers start with simple subroutines and build on them to create more complex programs and data structures. Its just that most any old computational structure can produce complex behavior that defies any analysis other than running the program to see what happens. A typical Macintosh, with an 80 megabyte hard drive is a universal computer with about $10^{200,000,000}$ states; it is clearly capable of some pretty complex misbehavior.

There are exceptions; for example the use of cellular automata for lattice gas models[7]. In this case very simple digital systems behave, in the limit, like systems easily modeled by differential equations.

The Finite State Machine

In order to better understand Finite Nature, we can look to examples of simple systems with similar properties. A digital computer, exclusive of its input and output facilities (with just a processor and memory), has many of the same properties that we find in Finite Nature. In particular, if we concentrate on a computer hardware technique called *single clock*, it will be easy to see why. *Single clock* means that when the computer is in a state, a single clock pulse initiates the transition to the next state. After the clock pulse happens, various changes propagate through the computer until things settle down again. Just after each clock pulse the machine lives for a short time in the messy real world of electrical transients and noise. At the points in time just before each clock pulse the computer corresponds exactly to the workings of a theoretical automaton or Finite State Machine.

The main point is that a computer is an embodiment of a system where Finite Nature is true (just before each clock pulse)[8]. Every such system has the property that it evolves by progressing from one state to the next state. Computers and all other such finite systems would eventually cycle around a closed trajectory if they ran long enough. Reversible systems are always on such a closed cycle, while an irreversible system can start down a path that will not repeat until finally entering a closed cycle. This is the informational viewpoint, as contrasted from the matter and energy viewpoint.

A computer needs to be able to perform three primitive functions: it must remember, communicate and compute. If we turn tables, systems of particles can model computers. An example is the Billiard Ball Model[9] of computation; an idealized 2-D billiard ball table with balls in motion that can exactly mimic the behavior of any computer. Particles remember by existing, communicate by moving and compute by interacting. A good understanding of what's happening reveals that communicating and remembering are the identical process (related by a space-time transformation) from the perspective of the physics involved.

INFORMATION

What is information? We know that "information" sometimes means "The meaning associated with data..." (Donald Knuth) but in this context we are referring to a scalar quantity that is a measure related to quantities like the $\log_2$(number of states) of some system. Finite Nature means that the world is made up of digital information. We understand "digital" exactly (made of, or making use of digits; symbols representing small integers) but we don't yet have a complete understanding as to how to measure or calculate with the scalar quantity named "information". This is not an easy problem, because

information, like energy or work, comes in different forms and thus it is hard to give a simple definition. Kinetic, potential, heat, electrical, chemical and others are all forms of energy; any of which can be transformed to other forms. Momentum, on the other hand is much simpler. There is really nothing else that momentum can be transformed into.

Despite not having a good definition of what information is, we postulate a new law; Conservation of Information. In this context information is the stuff that enables a reversible system to, if time were reversed, proceed backwards along the exact path that it took going forwards. All reversible systems have a conserved quantity; $\log_2(n)$, where n is the number of different states that the system progresses through in one cycle. (All finite state reversible systems evolve along a closed cycle.) This number is constant, independent of where the system is on that cycle and is equal to the quantity of information conserved.

Consider a reversible cellular automaton with a local neighborhood consisting of the seven cells which are east, west, north, south, up, down and past in relation to a particular cell. Finally, we have a transition rule, F, which defines for every possible combination of the states of the neighborhood cells, what state the particular cell should become.

$$C_{x,y,z,t+1}=F(C_{x-1,y,z,t}, C_{x+1,y,z,t}, C_{x,y+1,z,t}, C_{x,y-1,z,t}, C_{x,y,z+1,t}, C_{x,y,z-1,t}, C_{x,y,z,t-1})$$

For the system to be reversible, there must be another function, G, so that:

$$C_{x,y,z,t-1}=G(C_{x-1,y,z,t}, C_{x+1,y,z,t}, C_{x,y+1,z,t}, C_{x,y-1,z,t}, C_{x,y,z+1,t}, C_{x,y,z-1,t}, C_{x,y,z,t+1})$$

If these two criteria are met (the functions F and G exist), then we can say that the system conserves information, i.e., it is a law that information is conserved; no information ever gets lost. It may not be obvious as to what has happened to some information, but all of the information must be there in one form or another.

By observing the behavior of a non-trivial, reversible cellular automata, it becomes apparent that there is something almost magical about the way it evolves. What is hard to appreciate are the consequences of rules that appear to collapse information (a lot of information becoming just a little bit of information), yet nothing gets lost. If each cell has 3 states then 3^7 or 2187 possible states go into the computation of F. The resulting value of F is just one of three possible states. This seems to involve the loss of information; but each of the

neighbors is also involved in the computation of the states of other neighbors.

In ordinary physics, we can easily imagine perfect reversibility based on the fact that no effect ever gets lost because no matter how infinitesimal it becomes, it is still there. This approach makes it clear that the reason that an asteroid might hit the earth 5 years from now might be a direct consequence of the gravitational effect of a rock that rolled down a hill on the moon 2 billion years ago, delicately affecting the orbits of the Earth and all of the asteroids. But with the extreme quantization of the digital process, we have something more akin to the "For the want of a nail, the shoe was lost; for the want of the shoe, the horse was lost..." The results can be similar.

The Consequences of Finite Nature

The following is a list of some of the more obvious consequences of Finite Nature:

1. The *fundamental process* that underlies physics must be computation universal. Since the test of a computation universal process is that one can program up a universal machine within it, the existence of computers proves that the *fundamental process* is universal.
2. Things don't just happen; everything is a simple consequence of the *fundamental process.* Viewed from the informational perspective, the idea that a particle just has a momentum makes no sense. If the momentum is reasonably precise, meaning that it is one of a large number of possible momenta, then it contains a substantial amount of information, and that information must have a means of its representation.
3. The most fundamental stuff of physics is not the *cells* and *bits.* It is the result of the *fundamental process* run in the *cells* with the *bits.*
4. The future is computed as a function of the present and (most likely) the past. One way or another, it is very likely a second order system.
5. Matter has inertia because it takes some kind of process (like from a field) to change the momentum information (the information that governs the motion of the particle).
6. A particle accelerates when the environmental information, representing a field or a photon, interacts to change the information associated with the motion of the particle.
7. Information must have a means of its representation. If we had the right kind of magic microscope, we should

always be able to see the digits that represent whatever information is present. Information is never "just there".

8. What cannot be programmed cannot be physics. This is a very important idea. If a process cannot be programmed on a particular universal computer, despite having enough memory and time, then it cannot be programmed on any computer. If we can't program it on an ordinary computer, Finite Nature implies it can't be a part of physics because physics runs on a kind of computer.
9. An informational process cannot represent motion without a reference metric; finite nature demands the existence of a metric. This is a difficult idea. The implication is that the computational substrate of quantum mechanics must have access to some kind of metric in order to create inertial motion. Whether or not higher level processes in physics can access that metric is another question. It is hard to see why such access need be forbidden, since every particle must access the metric in order to move. Of course it is also clear that experiments done so far fail to detect any such metric. The argument for asserting the metric is that inertial motion requires an informational process that can take information that represents the velocity and use it to change the position of the particle. There must be information available that allows the process to know the velocity relative to something (the metric) that persists throughout the space that the particle will travel through. Without access to a constant, fixed space-time metric or its informational equivalent, there is no way to create an informational process that can involve particles moving independently in rectilinear motion
10. Energy, time, spin and other properties of the world cannot be programmed (made to work in an automata) without the existence of a metric.
11. Digital Mechanics is deterministic with unknowable determinism.
12. In general, physics is computing the future as fast as it can.
13. Some special cases (like the motion of the planets) can be computed faster than physics, allowing for predictions.
14. Most microscopic things, like quantum mechanics, cannot be computed on an ordinary computer faster than real time. From our perspective, the information bath that all events are immersed in will always be beyond our computing (in general). Thus, we call events dependent on that information "random".

15. The speed up theorem limits the detail with which we can know the future.
16. Physics is a consequence of the *fundamental bits* being engaged in the *fundamental informational process.*
17. The *Engine* is the computer that runs the *fundamental informational process* that runs physics.
18. The *Engine* and the *fundamental informational process* are computation universal.
19. Any universal *Engine* will do (given enough *memory*).
20. Physics is totally independent of the characteristics of the *engine*, so long as *it* is universal.
21. Everything mentioned in items 1 through 20 above that is in italics, is not a part of physics and is not to be found in this universe. Rather, it is stuff that exists somewhere, where, perhaps, the rules may be different. If we imagine using a computer to simulate air flow in a wind tunnel, we can think of that simulated airflow as a little universe following simple laws of physics. The computer is not made up of the air flowing in the simulated wind tunnel; the computer is not in the simulated wind tunnel, it is somewhere else.

Tasmania, or Physics out of Biology

On a recent trip to Tasmania, I visited a tree farm. The thousands of trees they planted every week were all identical clones. From a single cell of what was hoped to be a super tree, additional cells are grown in a tissue culture, and after a series of steps they have a forest of clones. It is one thing to contemplate the wonder that a little seed can carry the complete design of a tree, but it is another thing to contemplate that almost any part of the tree has the whole design of the tree in it! Imagine you catch someone trying to scrape of a few cells from his seat in a Boeing 747; with the idea of starting them off in a tissue culture and eventually growing 747 clones. What's wrong with the idea? It can work for trees and frogs! A floppy disk might contain the entire design of an integrated circuit memory chip; but why can't we make more memory chips by immersing the floppy in a nourishing solution instead of sending the floppy to a chip foundry?

The answers have to do with two laws about information; (1) "All information must have a digital means of its representation." We must be able to find in a single cell of the tree every bit of the information about the design of the tree. The second law involves process; (2) "An informational process transforms the digital representation of the state of a system into its future state." That a seed contains the design for a tree is not sufficient to transform the seed into the tree. The information in the seed must be acted upon and transformed like data

and programs are acted upon and transformed in the memory of a computer. The information in the seed is processed and transformed, along with other environmental information and material, in order to grow into the tree. The floppy may have all of the information about the memory chip, but it needs the environment of the chip foundry to be transformed into memory chips, in the same way that the seed needs fertile earth, sunshine, rain, air and time in order to be transformed into a tree.

Every physical system is subject to the same two laws!

---

[1] Dalton, John, "New System of Chemical Philosophy" (part I, 1808; part II, 1810)

[2] Minsky, Marvin, "Computation, Finite and Infinite Machines" (Prentice Hall, Englewood Cliffs, NJ, 1967)

[3] von Neuman, John, "Theory of Self-Reproducing Automata" (edited and completed by Arthur Burks), University of Illinois Press (1966)

[4] Banks, Edwin, "Information Processing and Transmission is Cellular Automata," Doctoral Dissertation and Technical Report MAC TR-81, MIT Project MAC (1971).

[5] Fredkin, Edward, "Digital Mechanics", Physica D, (1990) 254-270 North-Holland

[6] Turing, Alan, "On Computable Numbers with an Application to the Entscheidungsproblem", Proceedings London Math. Soc., series 2, 43 (1936), 544-546

[7] Frisch, Uriel, Brosl Hasslacher and Yves Pomeau, "Lattice-Gas Automata for the Navier-Stokes Equation" Phys. Review Letters 56 (1986), 1505-1508

[8] For a description of computation based on ideas from physics, see: Fredkin, Edward, "A Physicist's Model of Computation" Proceedings of the XXVIth Recontre de Moriond (1991) 283-297

[9] Fredkin, Edward, and Toffoli, Tommaso, "Conservative Logic," International Journal of Theoretical Physics 21 (1982), 219-253