Technical Report No. 2009-561
TIME TRAVEL:
A NEW HYPERCOMPUTATIONAL PARADIGM*†

Selim G. Akl
School of Computing, Queen's University
Kingston, Ontario, Canada K7L 3N6

July 22, 2009

Abstract

Assuming that all objections to time travel are set aside, it is shown that a computational system with closed timelike curves is a powerful hypercomputational tool. Specifically, such a system allows us to solve four out of five problems recently advanced as counterexamples to the fundamental principle of universality in computation. The fifth counterexample, however, remains unassailable, indicating that universality in computation cannot be achieved, even with the help of such an extraordinary ally as time travel.

Keywords: evolving computations, hypercomputation, time travel, unconventional computation, universality.

1 Introduction

One of the fundamental principles in computer science is that of *computational universality*, namely, that there exists a Universal Computer $\mathcal{U}$, on which one can simulate any computation possible on any other computer [17, 19, 25, 26, 27, 31, 32]. However, it was recently demonstrated that the principle of computational universality does not hold. Specifically, five instances were presented of a computable function $\mathcal{F}$ that cannot be computed on any purported Universal Computer $\mathcal{U}$ that is capable of a finite (and a priori fixed) number of elementary operations per time unit [3, 4, 5, 6, 8, 9].

This result applies not only to idealized models of computation, such as the Turing Machine, the Random Access Machine, and the like, but also to all known general-purpose computers, including existing conventional computers (both sequential and parallel), as well as contemplated unconventional ones, such as biological and quantum computers. The result is true even if the computer $\mathcal{U}$ can communicate and interact with the outside world to read input and return output (unlike the Turing Machine, but like every realistic general-purpose

*This research was supported by the Natural Sciences and Engineering Research Council of Canada.
†Invited Talk, in: Proceedings of the Hypercomputation Workshop, International Conference on Unconventional Computation, Ponta Delgada, Azores, Portugal, September 2009.

computer). It is also valid if $\mathcal{U}$, when attempting to compute $\mathcal{F}$, is endowed with an infinite memory and is given an unlimited amount of time (like the Turing Machine, but unlike realistic computers). Even accelerating machines that increase their speed at every step (by a fixed acceleration, such as doubling it, or squaring it) at a rate that is set in advance, cannot be universal.

The purpose of this paper is to explore the possibility of empowering $\mathcal{U}$ with the ability to travel in time, and the consequences of this new ability to computational universality. We begin in Section 2 with a computational challenge in order to provide some intuition. Five counterexamples to the existence of a Universal Computer are presented in Section 3. In Section 4, we briefly survey the arguments for and against time travel, and examine the consequences of equipping some purported Universal Computer with a time machine. Our conclusion, namely, that the non-universality result survives the assault of time travel, is presented in Section 6, along with a conjecture generalizing that result.

2 A Computational Challenge

On a limestone wall, n clocks are hanging as shown in Fig. 1 [7]. There are at least two clocks. The clocks are digital, each displaying the time as a quadruple of digits *WX:YZ*; for example, 19:48. All the clocks are working, ticking away synchronously (even as you read this challenge). However, they are not working as you might expect: At every tick, each clock displays a new, but random, quadruple *WX:YZ*, a time perhaps different from the ones displayed by the other $n-1$ clocks. No clock has a memory; therefore, when at the following tick a new time is generated and displayed, the previous quadruple is lost forever. The limestone wall is arbitrarily long, allowing n to be arbitrarily big.

Problem A: For an arbitrary number of clocks n, it is required to compute the average of the n times displayed at a given moment T.

This problem is easily solvable, at least in theory:

1. First, one has to read the n times displayed by the clocks,
2. Then the average function of the n readings can be directly computed.

The problem is also readily solvable in practice by a special-purpose computer. Note that in order to make the problem feasible, it is implicitly assumed that the moment T at which the problem poser asks for the average to be computed occurs *after* the problem solver declares being ready.

Computational Challenge: Design a universal computer that solves Problem A.

A valid solution must satisfy the following conditions:

1. The computer must be universal in the sense that, once defined, its specifications are fixed once and for all.

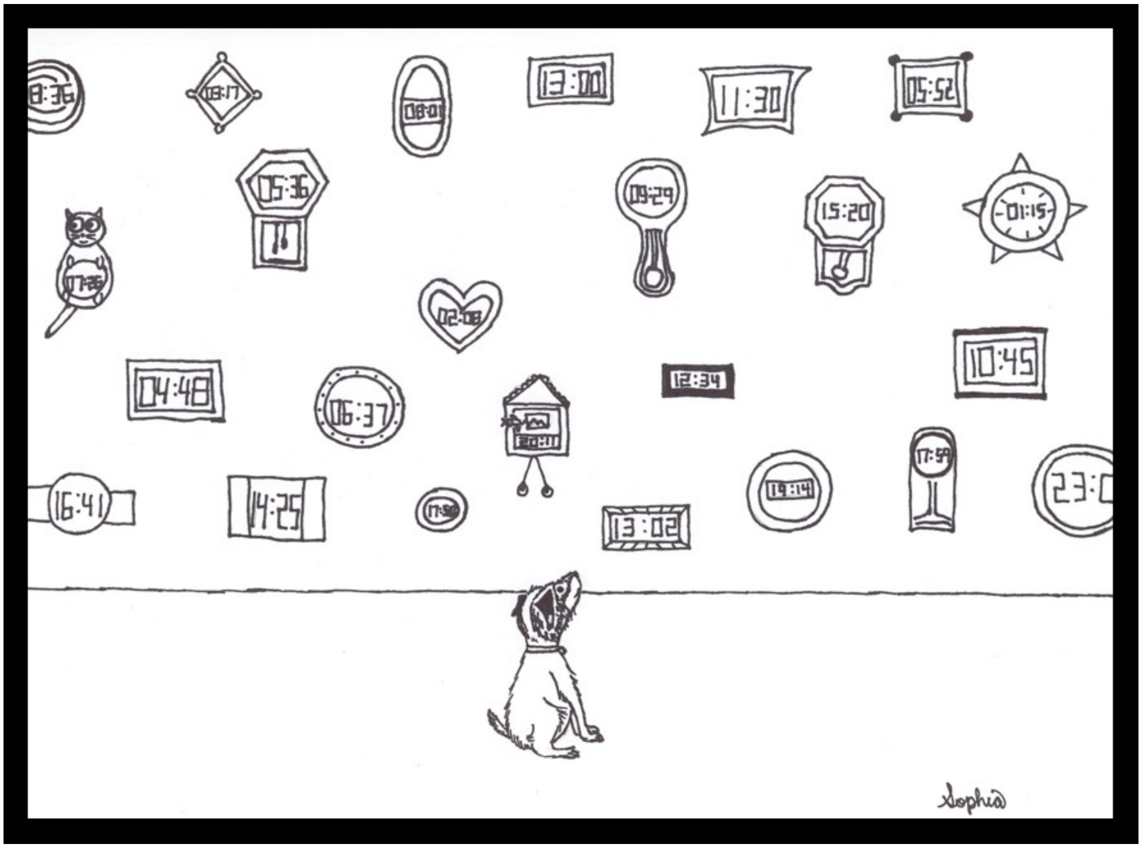

Figure 1: Computing the average of n clocks.

2. The universal computer should be able to solve Problem A for all values of n (a finite but unbounded positive integer) and at a moment T randomly selected by the problem poser.

3. There is no limit on the size of the memory your universal computer can have.

4. There is no limit on the amount of time your universal computer takes to solve Problem A.

5. The only limit is on the number of elementary operations (read, write, add, etc.) that your universal computer can perform per tick: this must be a finite constant.

6. Solutions invoking time travel or divine intervention are beyond the scope of this challenge.

The reader may wish to pause here and attempt to respond to this challenge. Otherwise, it is hoped that the challenge will serve as motivation to move on to the next section.

3 Non-Universality in Computation

A basic assumption in computer science is that every computer, be it theoretical or practical, performs a finite and fixed number of elementary operations per computational step. This number may be a constant (as on a laptop), or it may be a variable (as with accelerating machines), but it is always finite and fixed once and for all (even if it is a function of time). Furthermore, time is divided into discrete time units, and each computational step takes one time unit.

Having made our assumptions clear, we are now ready to state the non-universality result in its most general form. For $n \geq 1$, let

1. Π_n be the set of all computational problems with n scalar inputs (each of finite size), and

2. C_n be a model of computation capable of at most n elementary operations per time unit (each operating on a finite number of finite-size scalars).

Non-Universality Theorem: There exists at least one computational problem $\pi \in \Pi_{n+1}$ solvable on C_{n+1} but not on C_n. ■

We will prove this theorem by presenting (not one, but) five examples of problems $\pi \in \Pi_{n+1}$ that can be solved on C_{n+1} but not on C_n. In the context of these examples, the following formulation of the non-universality result is a little more specific:

Non-universality in computation: Given n spatially and temporally connected physical variables, $X_1, X_2, \ldots, X_n$, where n is a positive integer, there exists a function $F_n(X_1, X_2, \ldots, X_n)$ of these variables, such that no computer can evaluate F_n for any arbitrary n, unless it is capable of unboundedly many elementary operations per time unit. ■

Note that in our five examples to be presented in what follows, F_n is computable by C_n in a straightforward manner. However, the latter cannot compute F_{n+1}. While C_{n+1} can now compute the function F_{n+1}, C_{n+1} is in turn defeated by F_{n+2}. This continues forever.

This point deserves emphasis. While the function $F_{n+1}(X_1, X_2, \ldots, X_{n+1}) = \{F^{(1)}_{n+1}(X_1), F^{(2)}_{n+1}(X_2), \ldots, F^{(n+1)}_{n+1}(X_{n+1})\}$ is easily computed by C_{n+1}, it cannot be computed by C_n. Even if given infinite amounts of time and space, C_n is incapable of simulating the actions of C_{n+1}. Furthermore, C_{n+1} is in turn thwarted by $F_{n+2}(X_1, X_2, \ldots, X_{n+2})$, a function computable by C_{n+2}. The process extends indefinitely. Therefore, no computer is universal if it is capable of exactly $V(i)$ elementary operations during time unit i, where i is a positive integer, and $V(i)$ is finite and fixed once and for all (for it will be faced with a computation requiring $W(i)$ elementary operations during time unit i, where $W(i) > V(i)$ for all i).

Examples of such function F_n occur in:

1. *Computations with time-varying variables:* The variables X_i, over which the function is to be computed, are themselves changing with time. It is therefore appropriate

to write the n variables as $X_1(t), X_2(t), \ldots, X_n(t)$, that is, as functions of the time variable t. Further assume that, while it is known that the X_i change with time, the actual functions that effect these changes are not known (for example, X_i may be a true random variable). The problem calls for computing $F_n^{(i)}(X_i(t))$, for $i = 1, 2, \ldots, n$, at time $t = t_0$, where each $F_n^{(i)}$ is a simple function of one variable that takes one time unit to compute. Specifically, let $F_n^{(i)}(X_i(t))$ simply represent the reading of $X_i(t)$ from an external medium. The fact that $X_i(t)$ changes with the passage of time means that, for $k > 0$, not only is each value $X_i(t_0 + k)$ different from $X_i(t_0)$, but also the latter cannot be obtained from the former. No computer capable of fewer than n read operations per time unit can solve this problem.

We note in passing that the above example is deliberately simple in order to convey the idea. Reading a datum, that is, acquiring it from an external medium, is the most elementary form of information processing. Any computer must be able to perform such operation. This simplest of counterexamples suffices to establish non-universality in computation. Of course, if one wishes, the computation can be made more complex, at will. While our main conclusion remains unchanged, for some, a more complex argument may sound more convincing. Thus, for example, we may add arithmetic by requiring that $F_n^{(i)}(X_i(t))$ call for reading $X_i(t)$ and incrementing it by one, for $i = 1, 2, \ldots, n$, at time $t = t_0$. Here, the computational step, which involves reading an $X_i(t)$, incrementing it by one, and returning the result, takes one time unit. This means that n such steps performed in sequence would require n time units.

In any case, C_n (for example, a computer with n processors operating in parallel [2]) can compute all the $F_n^{(i)}(X_i(t))$ at $t = t_0$ successfully. On the other hand, C_{n-1} fails to compute all the $F_n^{(i)}(X_i(t))$ at $t = t_0$. It would compute $n-1$ of the $F_n^{(i)}(X_i(t))$ at $t = t_0$ correctly. Without loss of generality, assume that it computes $F_n^{(i)}(X_i(t))$ at $t = t_0$ for $i = 1, 2, \ldots, n-1$. Now one time unit would have passed, and when it attempts to compute $F_n^{(n)}(X_n(t_0))$, it would be forced to incorrectly compute $F_n^{(n)}(X_n(t_0 + 1))$.

Is C_n universal? Certainly not. For when the number of variables is $n + 1$, C_n fails to perform the required computation. As stated earlier, each computer in the sequence $C_n, C_{n+1}, C_{n+2}, \ldots$ succeeds at one level only to be foiled at the next.

2. *Computations with time-varying computational complexity:* The computational complexity of the function to be computed is itself changing with time. Suppose we are given n functions $g_0(X_0), g_1(X_1), \ldots, g_{n-1}(X_{n-1})$. All functions are independent, and computing $g_i(X_i)$ at time t requires 3^t elementary operations, $t \geq 0$. It is required to return the values of all n functions at $t = 2$. Not only does a conventional computer fail, but not even an accelerating machine capable of doubling the number of elementary operations it performs at each step, can solve the problem for $n > 1$.

3. *Computations with rank-varying computational complexity:* As before, we are given n independent functions to be computed, with no precedence constraints among them. Supposing that a schedule for computing the functions is established, the computational complexity of a function depends on its position in the schedule. For example, the function to be computed ith in sequence may require 3^i time units, where $i \geq 0$. As

in the case of computations with time-varying computational complexity, a deadline may be set for returning the values of the n functions. If this deadline is $t = 2$, then no conventional computer or accelerating machine can solve the problem for $n > 1$.

4. *Computations with interacting variables:* The variables of the function to be computed are parameters of a physical system that interact uncontrollably when the system is disturbed. Let the variables be X_0, X_1, ..., X_{n-1}. They need to be measured in order to compute $h_0(X_0)$, $h_1(X_1)$, ..., $h_{n-1}(X_{n-1})$, for some functions h_i, $i = 0, 1, \ldots, n-1$. The physical system has the property that measuring one variable disturbs any number of the remaining variables unpredictably and irreversibly, meaning that we cannot tell which variables have changed value, and by how much, nor can we restore the variables to their original values.

5. *Computations with global mathematical constraints:* A function is to be computed over a system whose variables must collectively obey a mathematical condition at all times. The function transforms the system from its original state to a goal state, through n operations. It is required that each intermediate state obey the same mathematical condition imposed on the original and goal states. However, there exist systems over which a transformation can be defined with the property that applying any one operation *in isolation* of the other $n - 1$ results in an intermediate state that does not satisfy the given mathematical condition.

   For illustration, consider a rewriting system. From an initial string ab, in some formal language consisting of the two symbols a and b, it is required to generate the string $(ab)^n$, for an even integer n larger than 1. The rewrite rules to be used are:

$$\begin{aligned} a &\rightarrow ab \\ b &\rightarrow ab. \end{aligned}$$

   Thus, for $n = 4$, the target string is $abababab$. Throughout the computation, no intermediate string should have two adjacent identical characters. Here we note that applying any *one* of the two rules at a time causes the computation to fail (for example, if ab is changed to abb, by the first rewrite rule, or to aab by the second).

Each of the above five examples establishes non-universality in computation. While necessarily finite, the problem size n, in each case, is unbounded. It is this unboundedness that defeats any supposed universal computer when faced with problems involving temporal and spatial variables that require computational ubiquity.

The implication to the theory of computation is akin to that of Gödel's incompleteness theorem to mathematics [21, 34]. In the same way that no finite set of axioms A_i can be complete, no computer C_i is universal that can perform a finite and fixed number of elementary operations per time unit. This is illustrated in Fig. 2: For every set of axioms A_i there exists a statement G_{i+1} not provable in A_i, but provable in A_{i+1}; similarly, for every machine C_i there is a problem P_{i+1} not solvable on C_i, but solvable on C_{i+1}.

It should be observed that the only constraint that we have placed on the computer (or model of computation) that claims to be universal is that the number of elementary

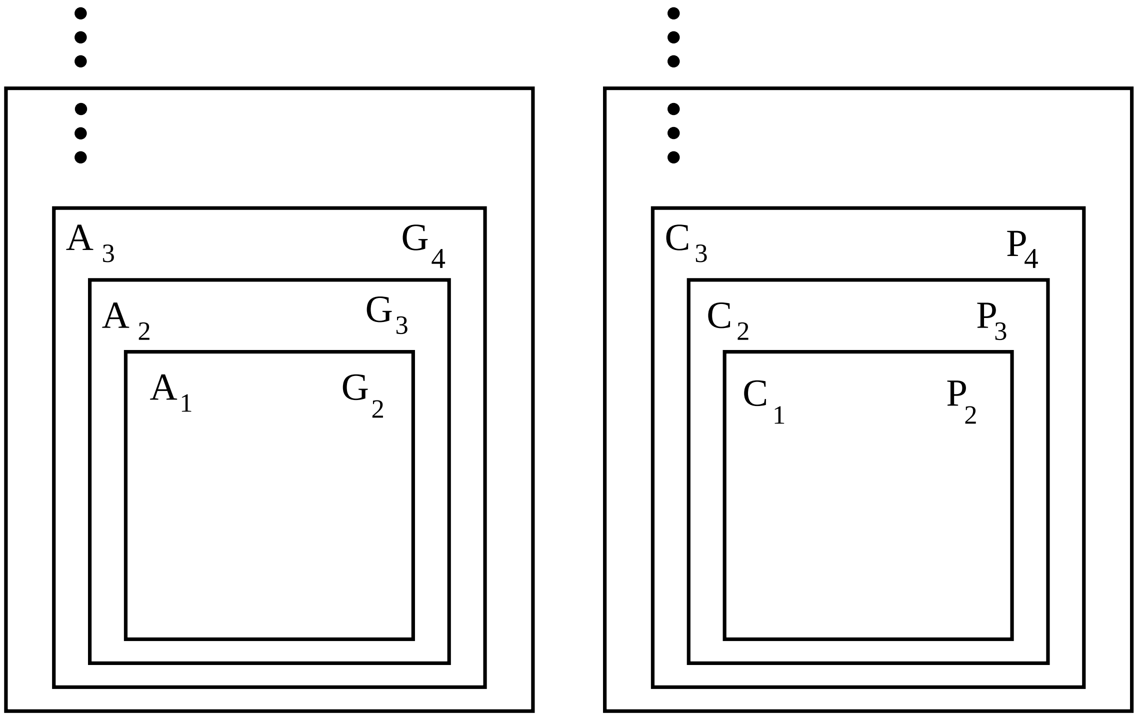

Figure 2: Incompleteness in mathematics an non-universality in computation.

operations of which it is capable at each time unit be finite and fixed once and for all. In this regard, it is important to note that:

1. The requirement that the number of elementary operations per time unit, or step, be *finite* is necessary for any "reasonable" model of computation (see, for example, [38]).

2. The requirement that this number be *fixed* once and for all is necessary for any model that purports to be "universal" (see, for example, [19]).

Without these two requirements, the theory of computation in general, and the theory of algorithms, in particular, would be totally irrelevant. To see this, note that in order to study the computational complexity of a problem, or the running time of an algorithm, a computational model is assumed. The operations that this model can perform are defined, and their number per time unit (however large) is bounded. This allows the complexity of the problem, or the running time of an algorithm for its solution, to be expressed as a function of the size of the problem (to within a constant multiplicative factor, which depends, primarily on the number of operations performed per time unit). The importance of the aforementioned requirements to the theory and practice of computing should therefore be clear. This is not to say that a computing system that evolves with the problems it faces (for example, by increasing the number of processors at its disposal, as needed) would not be useful. However, a system with unbounded power, a priori, would trivialize any attempt at a serious analysis.

The consequences of the non-universality result to theoretical and practical computing are significant. Thus the conjectured "Church-Turing Thesis" [38] is false. It is no longer true that, given enough time and space, any single general-purpose computer, defined a priori, can perform all computations that are possible on all other computers; not the Turing Machine, not a laptop, not the most powerful of supercomputers. In view of the computational problems mentioned above, the only possible universal computer would be one capable of an unbounded number of elementary operations per time unit.

4 Time Travel to the Rescue

When stating the conditions for solving the computational challenge of Section 2, solutions invoking time travel were explicitly forbidden. There, time travel was lumped with divine intervention as belonging to the realm of the fantastic, rather than serious computer science. We now revisit this assumption, and ask: What if ... time travel were indeed possible? Thus, for example, in the case of the 'time-varying variables' paradigm that disproved universality, it was required to read n variables at time t_0, namely, $X_0(t_0), X_1(t_0), X_2(t_0), \ldots, X_{n-1}(t_0)$. This proves impossible for a universal computer since n is unbounded. But, with time travel, could we go back in time and pick up the readings that we missed?

4.1 Is there a time arrow?

To be sure, there are several objections to the possibility of traveling back in time. Some of the standard arguments against time travel are given in what follows.

1. **Thermodynamic arrow of time:** The entropy of a closed system increases with the passage of time and this process cannot be reversed.

2. **Cosmological arrow of time:** The Universe is expanding at an accelerating rate and the direction of its expansion defines an arrow of time.

3. **Electromagnetic arrow of time:** The unidirectional propagation of electromagnetic waves precludes the possibility of going back in time in the opposite direction.

4. **Quantum arrow of time:** The decoherence of a quantum system, subject to an observation or simply through exposure to the environment, is irreversible.

4.2 Closed timelike curves

All of the standard objections to time travel notwithstanding, it appears that theoretical physics may in fact allow time travel. Einstein's Field Equations of General Relativity are given by

$$G_{\mu v} = 8\pi T_{\mu v}$$

where $T_{\mu v}$ is the stress-energy tensor, and

$$G_{\mu v} = R_{\mu v} - \frac{1}{2} g_{\mu v} R$$

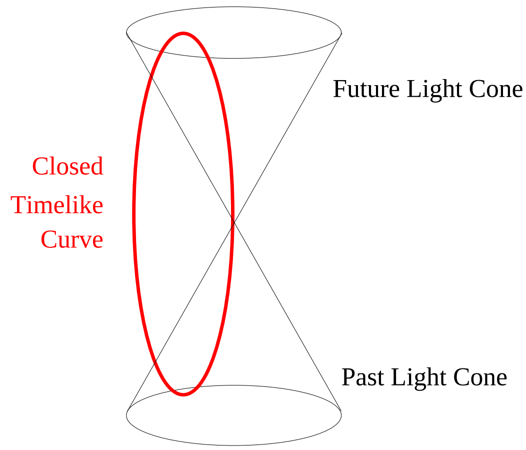

Figure 3: A closed timelike curve.

is the Einstein tensor, $R_{\mu v}$ the Ricci curvature tensor, $g_{\mu v}$ the metric tensor, and R the scalar curvature [20, 30]. These equations relate the curvature of spacetime to the matter and energy content of the universe. Thus, the geometry of spacetime is curved by mass, and the motion of masses is determined by the geometry of spacetime. Some solutions to these equations lead to so-called *closed timelike curves* or CTCs, defined as follows:

- A *light cone* represents all possible future positions of an object from its current position.
- If space were curved, a given trajectory would form a CTC, as shown in Fig. 3.

A CTC would allow for time travel: Fig. 4, illustrates a series of light cones that loops back on itself in a CTC; by moving along the CTC, one would return to a previous moment in time.

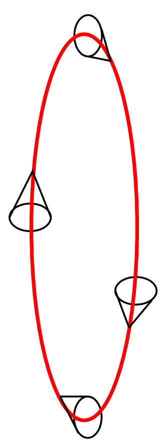

Figure 4: A closed timelike curve would permit time travel.

4.3 How can a time machine be built in theory?

Einstein's Theory of General Relativity states that space is curved in the vicinity of massive bodies, which explains the phenomenon of gravity. Most, though not all, attempts at proposing a way to build a time machine involve a very large and heavy object that is rotating at very high speed, thus causing space to curve, thereby creating a CTC. Some of these proposals and others are outlined in chronological order in Table 1 (for surveys and further references, please see [11, 16, 28, 35]).

Author	**Time Machine**
Van Stockum [42]	An infinitely long rotating cylinder
Gödel [22]	A rotating universe
Kerr-Newman [29, 36]	Rotating black holes
Tipler [39]	A sufficiently long but finite rotating cylinder
Thorne [40, 41]	Two spheres connected by a wormhole and using negative energy
Gott [23, 24]	Traveling around colliding cosmic strings
Alcubierre [12, 37]	Warp drive machines traveling faster than the speed of light

Table 1: Various proposals for time machines.

4.4 Difficulties with time travel

Assuming the theoretical issues surrounding time travel as discussed so far are settled, there remain serious engineering obstacles, even if we assume a civilization significantly more advanced than ours, with a technology vastly superior to what we can muster. These obstacles are created by the requirements in the existing proposals for building time machines, including:

1. Infinitely long cylinders [42]
2. Prohibitively long cylinders [39]
3. Rotating black holes [29, 36]
4. Wormholes, negative matter, and negative energy [40, 41]
5. Cosmic strings [23, 24]
6. Superluminal warp drives [12, 37].

Another difficulty is the fact that these time machines do not allow travel to a date in the past that is earlier than the time the machine itself was built. This means that, if we wish to travel to our own past, we must hope that a more advanced civilization has already created such machines.

4.5 More difficulties with time travel

Another type of challenge facing time travel are the so-called paradoxes that it appears to create. For example:

1. **Grandfather paradox:** This paradox involves changing the past through actions that make the present illogical, as when one travels to the past, and kills his grandfather before his father is born, thereby making his own existence impossible.

2. **Information paradox:** Here, knowledge comes from the future to make the same knowledge available in the future, as when a writer, successful of late, goes back in time and gives the novel that caused her fame to her struggling younger self.

3. **Bilker's paradox:** In this paradox, the present is changed in order to prevent the future from happening, as when someone travels to the future, discovers that she will be a mediocre violinist, returns and picks up the flute instead.

4. **Sexual paradox:** There are many variants to this paradox, including traveling to the past and accomplishing the biologically impossible feat of becoming one's own father.

Time travel enthusiasts have proposed various ways to resolve these paradoxes, including:

1. **Events occur only once.** Thus, if you did not do something at a certain moment in the past, then you cannot go back and do it through time travel, because that moment occurred only once with you in it. In other words, there is no second time [35].

2. **Novikov's self-consistency principle.** Here, an invisible law of nature prevents actions leading to paradox. This results in interesting debates about free will [28].

3. **Parallel universes.** The idea here is that for every choice a new world is created [19].

4.6 Objections specific to unconventional problems

Let us assume some 'universal computer' is allowed unlimited time travel to the past to recover what was missed. Still, some non-standard arguments against the usefulness of time travel persist. For example [10]:

1. **Duration of the trip:** It takes time to get there and back; you may end up at a different time both ways!

2. **Parallel worlds:** You may end up in a different world both times!

3. **Philosophical question:** Can information be recovered from the past?

4.7 What if time travel were possible?

Putting all objections aside, would time travel help restore universality in computation? Would a general-purpose computer equipped with time travel, solve the five counterexamples to universality, namely,

1. Time-varying variables
2. Time-varying computational complexity
3. Rank-varying computational complexity
4. Interacting variables
5. Mathematical constraints

for all values of n?

4.8 Basic assumptions

For the sake of coherence and plausibility, to the extent possible, we make two assumptions that follow from the discussion in the previous sections:

Assumption 1. The past cannot be changed through time travel.

Assumption 2. Time travel to the past operates on independent entities, possibly in parallel worlds.

Because of the second assumption, the first assumption is not violated: Obtaining information (for example, reading a variable), or performing an action (for example, computing a function), do not represent alterations of the past.

4.9 Can time travel overcome the passage of time?

As Figure 5 indicates, it would be indeed possible through time travel to solve the time-varying variables computation with a computer capable of only one elementary operation per time unit. We simply read $X_0(t_0)$ at time t_0, then go back in time and recover each of $X_1(t_0)$, $X_2(t_0)$, and so on. In order to understand Fig. 5, note that we assume for simplicity that $t_0 = 0$. Reading X_0 at time t_0 takes one time unit, so we are presently at time $t_1 = 1$. In order to read X_1 at time t_0, we need to travel one time unit back in time. Another time unit is required to perform the read operation itself. Since two time units have now elapsed, we must now traverse the same period in time to return, and thus we find ourselves at $t_2 = 5$. This continues until we have read X_{n-1}. Here, the value of Assumption 2 in the previous section becomes apparent: It is convenient to interpret each trip back in time as taking us to a different parallel world, thus avoiding time travel paradoxes (such as, for example, the computer meeting several younger copies of itself, each reading a different X_i at t_0).

Similar solutions can be applied to the next three computations that defied universality, namely,

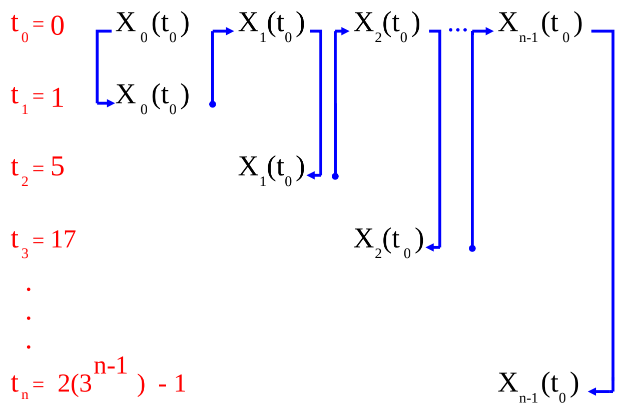

Figure 5: Reading time-varying variables by traveling to the past.

1. Time-varying computational complexity, where we go back in time and compute each of the functions in one time unit at time $t = 0$,

2. Rank-varying computational complexity, where we go back in time and compute each of the functions in one time unit at time $t = 0$ as if it had rank 0, and

3. Interacting variables, where we go back in time and measure each of the variables, as if it were the first to be measured.

It is interesting to note that in all four computations executed through time travel, the only price to be paid is, ironically, time itself: It takes $2 \times 3^{n-1} - 1$ time units for C_1 to accomplish what C_n would do in only one time unit. In fact, in the cases of time-varying computational complexity and rank-varying computational complexity, care must be taken to return to a time before the deadline when reporting the results, which itself may not be possible if we are to travel back to the same world where the computation originated.

The reader may object to the solution, through time travel, of the computation involving interacting variables. After measuring one variable, is it not the case that the remaining $n - 1$ variables are affected unpredictably? How can we go back and obtain reliable values for them? This is indeed a subtle point. The fact of the matter here is that the system of variables *as a whole* remains unaffected in unvisited parallel worlds. Each trip to a parallel world, previously unvisited, finds each of the n variables in its pristine original state. The ith variable can thus be measured accurately in the ith trip as though it were the first to be measured.

However, time travel is of no help in allowing C_1 to solve problems involving mathematical constraints, for $n > 1$. In the example of the rewriting system presented in Section 3, after one application of one of the two rules, the condition is already violated and the computation has already failed. No amount of time travel can change the past. This is to be contrasted with the discussion in the previous paragraph. Unlike the case with the interacting variables computation, here we are concerned with maintaining a mathematical condition on the *whole system, across all parallel worlds*. It helps to think of the present computation in one of two ways: Either there exists a single instance of the system, or there exist n separate copies of the system.

In both cases, the requirement is for the system to be transformed, while obeying a mathematical condition throughout the transformation. In the first case, one operation causes the mathematical condition to be violated by the single instance residing in the past. Every subsequent visit to the system, taking place in a different parallel world, will find the same system already altered beyond repair. In the second case, every visit to the system will find it in its original state (unchanged by previous visits). Each visit will operate on a different copy, in a different world, resulting in n failed computations.

4.10 Would time travel to the future help?

A computer equipped with a CTC may be able to travel to the future and send itself (back in time, to the present) solutions to problems that are otherwise computationally intractable. This idea was proposed by several authors [1, 13, 14, 15, 18, 33]. It is not clear, however, how time travel to the future would solve problems with mathematical constraints, that time travel to the past has failed to solve (and, presumably, one would be more interested to learn how the stock market will do in the future!).

5 About Time

Time is one of the most elusive and mysterious concepts in human experience. Throughout the ages, it has preoccupied philosophers, scientists, poets, and saints. Here, we reflect briefly on the role of time in computing.

In conventional computing, time is:

1. Internal to the computer (time is *running time*, it is unrelated to the outside world, and is valid only when performing a computation);
2. Static (time depends on the algorithm while nothing depends on time);
3. Unbounded (at least in theory);
4. Entirely under the control of the user (a better algorithm improves time and often there is time-space tradeoff);
5. Used solely in performance analysis.

In unconventional computing, by contrast, time takes a new significance. Physical time in the environment external to the computer plays a crucial role in the definition of the computational problem, its requirements, and ultimately in the success or failure of the computation. This is illustrated in this paper. As an actor in a play, time portrayed three characters in the pursuit of computational universality. First, time was the foe, whether as a central or secondary figure; it is the inexorable *passage of time* that defeated universality. Second, *time travel* was a friend, an ally of sorts, called in to rescue computational universality as a fundamental principle in computer science. Third, as time travel was pitted against the passage of time, *time itself* became the price to pay, the currency spent in the vain attempt to salvage universality.

6 Conclusion

Time travel is a powerful hypercomputational paradigm. Assuming all objections are ignored (or addressed), time travel allows us to solve problems of size n that are otherwise only solvable for all n on a universal computer capable of an unbounded number of elementary operations per time unit. These problems include the following ones presented in this paper and involving:

1. Time-varying variables
2. Time-varying computational complexity
3. Rank-varying computational complexity
4. Interacting variables

However, the non-universality result remains valid! Time travel, despite being an extraordinary assumption in itself, still fails to solve the counterexample to universality which involves variables that must obey a mathematical constraint throughout a computation. In these types of computation, time plays a secondary but crucial role: The relevant mathematical condition is maintained if and only if actions are simultaneous. The requirement of computational ubiquity appears in this case to render time travel useless. The data needed to solve the problem are all available at the outset, and the transformations needed to be applied to them are all known. Neither time travel to the past, nor time travel to the future (regardless of how far one goes in either direction), will allow for the necessary actions to be performed on the data *at the same time*, as required for a successful computation.

Non-universality poses a serious challenge to some fundamental beliefs in computer science. As we saw, time travel, assuming it is possible, appears to mitigate this inconvenience. But not always! An interesting open question therefore is this: Can those computational problems amenable to solution by time travel (and those that are not) be fully identified? A somewhat related question is whether there exist computational problems that require for their solution the following sequence of time travels: travel to the past, return to the present, travel to the future, and finally a return to the present (with the sequence possibly repeated several times, until a solution is obtained). Finally in this vein, can the time required by our solution illustrated in Fig. 5 be reduced from $2 \times 3^{n-1} - 1$ to $4n - 2$, by continuously traveling to the past, avoiding a return to the present, until after the final trip to the past is completed (for example, when X_{n-1} has been read)?

We end this study by returning to the subject that motivated it in the first place, namely, non-universality in computation. What follows is a conjecture that generalizes our non-universality result. Let H be a branch of human thought, such as mathematics, science, philosophy, linguistics, and so on. A system U is said to be universal over H if it encompasses all the constituents of H, through explanation, generation, or solution, as the case may be. If U is closed (that is, finite and fixed), then it cannot be universal over H. Evidence for the validity of the conjecture already exists in mathematical logic, physics, and linguistics; this paper presented a proof for computer science.

Acknowledgments

In March 2009, I had the pleasure of meeting Professor István Németi of the Rényi Institute of Mathematics, Budapest, Hungary. The occasion was the Conference on the Science and Philosophy of Unconventional Computation, held at the University of Cambridge. István encouraged me to pursue the research described here and generously provided some of the background on the physics of time travel. I am grateful for his enthusiasm and dedicate this paper to him. I also wish to thank members of the Parallel and Unconventional Computation Group in the School of Computing at Queen's University, and in particular Naya Nagy, Marius Nagy, and Charlotte L. Haley, for interesting discussions and comments.

References

[1] Aaronson, S. and Watrous, J., Closed timelike curves make quantum and classical computing equivalent, *Proceedings of the Royal Society A*, Vol. 465, No. 2102, 2009, pp. 631–647.

[2] Akl, s.G., *Parallel Computation: Models and Methods*, Prentice-Hall, Upper Saddle River, New Jersey, 1997.

[3] Akl, S.G., The myth of universal computation, in: *Parallel Numerics*, Trobec, R., Zinterhof, P., Vajteršic, M., and Uhl, A., Eds., Part 2, Systems and Simulation, University of Salzburg, Salzburg, Austria and Jožef Stefan Institute, Ljubljana, Slovenia, 2005, pp. 211–236.

[4] Akl, S.G., Three counterexamples to dispel the myth of the universal computer, *Parallel Processing Letters*, Vol. 16, No. 3, 2006, pp. 381–403.

[5] Akl, S.G., Conventional or unconventional: Is any computer universal?, in: *From Utopian to Genuine Unconventional Computers*, Adamatzky, A. and Teuscher, C., Eds., Luniver Press, Frome, United Kingdom, 2006, pp. 101–136.

[6] Akl, S.G., Even accelerating machines are not universal, *International Journal of Unconventional Computing*, Vol. 3, No. 2, 2007, pp. 105–121.

[7] Akl, S.G., A computational challenge, manuscript, 2008, `http://research.cs.queensu.ca/home/akl/CHALLENGE/A_Computational_Challenge.htm`

[8] Akl, S.G., Unconventional computational problems with consequences to universality, *International Journal of Unconventional Computing*, Vol. 4, No. 1, 2008, pp. 89–98.

[9] Akl, S.G., Evolving computational systems, Chapter One in: *Handbook of Parallel Computing: Models, Algorithms, and Applications*, Rajasekaran, S. and Reif, J.H., Eds., Taylor and Francis, CRC Press, Boca Raton, Florida, 2008, pp. 1–22.

[10] Akl, S.G., Space and time in unconventional computation: Science and philosophy, Invited talk, Conference on the Science and Philosophy of Unconventional Computation, Cambridge, United Kingdom, March 2009.

[11] Andréka, H., Németi, I., and Wüthrich, C., A twist in the geometry of rotating black holes: seeking the cause of acausality, *General Relativity and Gravitation*, Vol. 40, 2008, 1809–1823.

[12] Alcubierre, M., The warp drive: Hyper-fast travel within general relativity, *Classical and Quantum Gravity*, Vol. 11, 1994, pp. 73–77.

[13] Bacon, D., Quantum computational complexity in the presence of closed timelike curves, *Physical Review A*, Vol. 70, 2004, pp. 032309.1–032309.7.

[14] Brun, T.A., Computers with closed timelike curves can solve hard problems efficiently, *Foundations of Physics Letters*, Vol. 16, No. 3, 2003, pp. 245–253.

[15] Brun, T.A., Harrington, J., and Wilde, M.M., Localized closed timelike curves can perfectly distinguish quantum states, *Physical Review Letters*, Vol. 102, 2009, pp. 210402.1–210402.4

[16] Davies, P., *How to Build a Time Machine*, Penguin, Toronto, 2003.

[17] Denning, P.J., Dennis, J.B., and Qualitz, J.E., *Machines, Languages, and Computation*, Prentice-Hall, Englewood Cliffs, New Jersey, 1978.

[18] Deutsch, D., Quantum mechanics near closed timelike lines, *Physical Review D*, Vol. 44, 1991, 3197–3218.

[19] Deutsch, D., *The Fabric of Reality*, Penguin Books, London, England, 1997.

[20] Einstein, A., Die Grundlage der allgemeinen Relativitätheorie, *Annalen der Physik*, No. 17, 1916, pp. 284–339.

[21] Gödel, K., Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I, *Monatshefte für Methematik und Physik*, Vol. 38, 1931, pp. 173–198.

[22] Gödel, K., An example of a new type of cosmological solutions of Einstein's field equations of gravitation, *Review of Modern Physics*, Vol. 21, 1949, pp. 447–450.

[23] Gott, J.R., Closed timelike curves produced by pairs of moving cosmic strings: Exact solutions, *Physical Review Letters*, Vol.66, 1991, pp. 1126–1129.

[24] Gott, J.R., *Time Travel in Einstein's Universe*, Houghton Mifflin, Boston, 2001.

[25] Harel, D., *Algorithmics: The Spirit of Computing*, Addison-Wesley, Reading, Massachusetts, 1992.

[26] Hillis, D., *The Pattern on the Stone*, Basic Books, New York, 1998.

[27] Hopcroft, J.E. and Ullman, J.D., *Formal Languages and their Relations to Automata*, Addison-Wesley, Reading, Massachusetts, 1969.

[28] Kaku, M., *Parallel Worlds*, Doubleday, New York, 2005.

[29] Kerr, R.P., Gravitational field of a spinning mass as an example of algebraically special metrics, *Physical Review Letters*, Vol. 11, 1963, pp. 237–238.

[30] Kox, A.J., Klein, M.J., and Schulmann, R., Eds., *The Collected Papers of Albert Einstein*, Vol. 6, Princeton University Press, Princeton, New Jersey, 1997.

[31] Lewis, H.R. and Papadimitriou, C.H., *Elements of the Theory of Computation*, Prentice Hall, Englewood Cliffs, New Jersey, 1981.

[32] Minsky, M.L., *Computation: Finite and Infinite Machines*, Prentice-Hall, Englewood Cliffs, New Jersey, 1967.

[33] Moravec, H., Time travel and computing, manuscript, 1991, `http://www.frc.ri.cmu.edu/~hpm/project.archive/general.articles/1991/TempComp.html`

[34] Nagel, E. and Newman, J.R., *Gödel's Proof*, New York University Press, New York, 2001.

[35] Nahin, P.J., *Time Machines*, Springer-Verlag, New York, 1999.

[36] Newman, E.T., Couch, E., Chinnapared, K, Exton, A. Prakash, A., and Torrence, R., Metric of rotating, charged mass, *Journal of Mathematical Physics*, Vol. 6, No. 6, 1965, pp. 918–919.

[37] Parsons, P., A warped view of time travel, *Science*, Vol. 274, No. 5285, 1996, pp. 202–203.

[38] Sipser, M., *Introduction to the Theory of Computation*, PWS Publishing Company, Boston, Massachusetts, 1997.

[39] Tipler, F.J., Rotating cylinders and the possibility of global causality violation, *Physical Review D*, Vol. 9, 1974, pp. 2203–2206.

[40] Thorne, K.S., Do the laws of Physics permit closed timelike curves?, *Annals of the New York Academy of Science*, Vol. 631, 1991, pp. 182–193.

[41] Thorne, K.S., *Black Holes and Time Warps: Einstein's Outrageous Legacy*, W.W. Norton, New York, 1994.

[42] van Stockum, W.J., The gravitational field of a distribution of particles rotating about an axis of symmetry, *Proceedings of the Royal Society of Edinburgh*, Vol. 57, 1937, pp. 135–154.