Time travel and computation

Sebastian Stengele

2.8.2022

Introduction

Time travel – or more precisely closed timelike curves (CTCs) – have been studied in physics in a multitude of different ways. For example, solutions of Einstein's field equations exhibiting CTCs (such as the famous Gödel metric) have been found, and consistency conditions for fields living on such spacetimes have been formulated.

Interestingly, there is a somewhat strange connection between general relativity and the theory of complexity: Certain spacetime geometries, so-called Malament-Hogarth spacetimes, allow undecidable problems to be solved in finite time (as seen by an observer), however with quite substantial drawbacks, as the observer will need to jump into a black hole.

Coming back to time travel, this raises the question whether CTCs can be used in a similar way to speed up computations or even to solve undecidable problems. One easy way to instantaneously solve a large class of problems is to directly send the solution back in time, that is, solving the problem as one would normally do – using as much resources as one would normally need – and then sending the outcome back to the point in time when the computation started. However, while this can solve problems in constant time, it does not change the resources needed to solve it, they are just spent after we already know the solution[1].

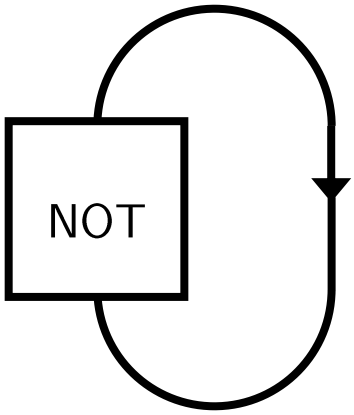
A refined question is thus whether we can speed up computations while actually using bounded resources. This question is thoroughly treated in [1] where it is shown that both classical and quantum polynomial-time algorithms

[1] And if we already know the solution, why should we spend the resources? Or we could just use our time machine computer again to get the solution instead of actually spending the resources. This might lead to all kind of interesting paradoxes.

supplemented with CTCs are equal to PSPACE, the class of languages accepted by a Turing machine using a polynomial amount of memory.

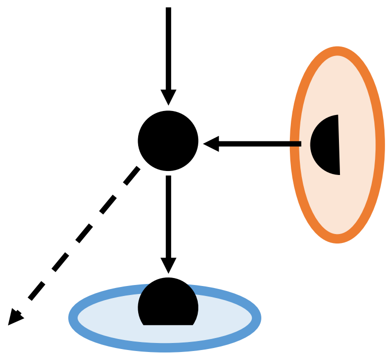
Paradoxes

The main idea connecting CTCs and computation is to make use of consistency conditions. Consider the setup on the right: The output of a NOT-Gate is looped back in time and feed into its input. This is a simplified version of the grandfather paradox. Assume the input would be true, then the output is false, which loops back and becomes the input, which should however be true. Hence no truth value can be assigned to the input. Similarly, if the gate would be the identity instead, both true and false are consistent assignments, so the system is underdetermined. In general, only systems or gates with a fixed point have a consistent assignment when placed in a CTC, and any fixed point is such an assignment.



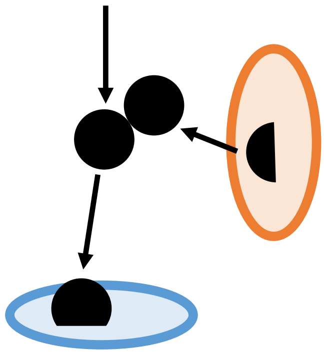
Billiard

In the case of computation, one can always construct a system with at least one fixed point, however for general physical theories this seems to pose a large restriction on the possible physics. Consider for example, the following setup: A ball is thrown into a wormhole, it exits the other side earlier, just in the right moment to collide with its earlier self and knock it off its path such that it will not enter the wormhole and hence not collide with itself.



This is not a consistent solution. However, it does not imply that the system has no fixed points. For example, imagine that the ball exits the wormhole with a slight angle, such that it hits its younger self also at an angle, deflecting just a bit, such that it enters the wormhole at a slight angle. This solution is consistent, and it has been shown that a large class of such problems always have at least one consistent solution [2].

So, a large number of physical scenarios which seem to be inconsistent at first turn out to have a consistent solution. This apparent inconsistency arises from naively evolving the system along the proper time of the system travelling through the wormhole. In fact, one would need to take into account the collisions for all possible ways the ball could exit the wormhole a and then check whether any of those lead to a consistent solution.



Oh no NOT again

In the same spirit one can try to find a consistent assignment for the example of the NOT gate. If we consider the gate only on $\{0,1\}$ it is easy to see that there is no consistent solution[2]. However, if one allows probability distributions over $\{0,1\}$ a consistent solution is easily found: Representing distributions by the probability p of getting 0, we can write the NOT gate as $\mathsf{NOT}\ \mathrm{p}\ = 1 - \mathrm{p}$. This has a fixed point at $p = \frac{1}{2}$. In words, if both 0 and 1 are equally probable, then switching them doesn't change the probability distribution. In general, for any function $f\colon \{0,1\}^n \to \{0,1\}^n$, there are fixed points over the distributions. Namely, a uniform distribution over the elements of a cycle in the graph of the function[3] does the job. Such a cycle will always exist for finite graphs of a function.

Replacing classical gates with quantum operations, the same holds [3]. Any quantum operation has a fixed point, which in general is not a pure state but a density matrix.

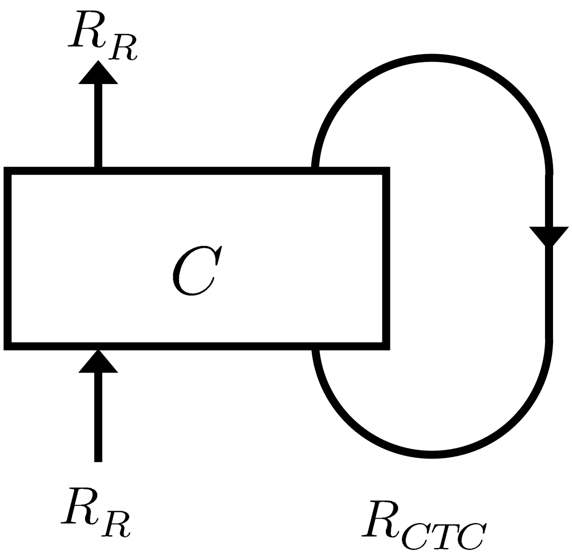
[2] This is just the grandfathers paradox.

[3] I.e. the graph where the values are nodes and there is an edge from x to y iff $f(x) = y$.

Classical computation

Motivated by the previous examples, Aaronson and Watrous use the fact that CTCs – if they exists – will have a consistent solution as a computational resource. More precisely, the fact that CTCs will "find" a fixed point for any operation is crucial, as this is generally a computationally expensive task.

In the first step, they define the class $\mathsf{P}_{\mathsf{CTC}}$ of problems solvable by classical computers using CTCs. They define a classical computer with CTCs as a polynomial-Time Turing machine which for any input $x \in \{0,1\}^*$ outputs a Boolean circuit $C = C_x$ containing only AND, OR and NOT gates. Furthermore, the circuit should act on two registers, a causality-respecting register R_R and a CTC register R_{CTC}. The latter will be placed in a CTC, such that a consistent solution will be a fixed point of C given the values for the causality-respecting register.



For simplicity, assume that the causality-respecting register is always initialized to 0. Then let D' be a distribution over R_{CTC} which is a fixed point of $C(0, -)$. As argued above, such a distribution always exists. Let $D = \langle 0, D' \rangle$ be the distribution over $R_R \times R_{CTC}$ which is always 0 on R_R and D' on R_{CTC}.

A CTC algorithm accepts an input x if, for every such distribution D, the distribution $C_x(D)$ has only support on values where the last bit of the causality-respecting register is 1. Similarly, if every distribution has only support where the last bit is 0, we say that the algorithm rejects input x.

Now, to show that $\mathsf{P_{CTC}} \subseteq \mathsf{PSPACE}$, note that any cycle in the graph of $C(0,-)$[4] will lead to a 1 in the last bit of R_R if x is accepted and to a 0 if x is rejected. Hence an algorithm looking for a cycle and then checking R_R can solve any problem in $\mathsf{P_{CTC}}$. Finding such a cycle is easy in polynomial space. For example, by repeatedly applying C for any input until a cycle is found[5].

For the other direction, $\mathsf{PSPACE} \subseteq \mathsf{P_{CTC}}$, choose a deterministic $p(n)$-space Turing machine T for some polynomial p. Without loss of generality we can assume that T always halt[6]. Now, consider the polynomial-size circuit S which acts on the configurations m of T (that is, tape content, state and head position) and maps each state to its successor (Note that S cannot have a fixed point, otherwise T would loop on some input). Using that, construct

$$C\ m,b,r\ = \begin{cases} \langle m_0, 1{,}1\rangle & m \text{ is an accepting configuration} \\ \langle m_0, 0{,}0\rangle & m \text{ is a rejecting configuration} \\ \langle S\ m\ , b, b\rangle & \text{otherwise} \end{cases}$$

where r is in R_R and m,b are in R_{CTC} and where m_0 is the initial configuration of T. Now assume that T will accept the input, then the only possible cycle in the graph of C is the one where $\langle m_0, 1{,}1\rangle$ gets mapped along the computational path to an accepting configuration $\langle m_a, 1{,}1\rangle$ and then back to $\langle m_0, 1{,}1\rangle$. If m_0 is rejected, there is a similar cycle with $b = r = 0$. Hence, if x is a yes-instance, the corresponding circuit will yield $r = 1$ and if x is a no-instance it will yield $r = 0$. This shows that indeed $\mathsf{PSPACE} = \mathsf{P_{CTC}}$.

[4] Which correspond exactly to the consistent distributions.

[5] One needs to take care that the algorithm halts. However, as the length longest cycle is exponential in the input length, the algorithm can check in polynomial-space whether that length has been reached and in this case continue with the next input (to C).

[6] Otherwise we can construct a Turing machine T' which counts the number of steps and halts when the number of steps is larger than the number of possible configurations of T (including the tape). As the number of configurations is exponential in the input size, such a counter only uses polynomial space.

Quantum computation

In their paper Aaronson and Watrous then continue to discuss the quantum case, where they show that the quantum analog of $\mathsf{P}_{\mathsf{CTC}}$, $\mathsf{BQP}_{\mathsf{CTC}}$, is also equal to PSPACE.

$\mathsf{BQP}_{\mathsf{CTC}}$ is defined similarly to $\mathsf{P}_{\mathsf{CTC}}$, the difference being that classical gates, registers and circuits are replaced by quantum gates, registers and circuits, and that an algorithm accepts or rejects, when the respective measurement of R_R yields 1 or 0 a probability of at least 2/3, respectively.

While $\mathsf{PSPACE} \subseteq \mathsf{BQP}_{\mathsf{CTC}}$ follows from the fact that $\mathsf{P}_{\mathsf{CTC}} \subseteq \mathsf{BQP}_{\mathsf{CTC}}$, showing that $\mathsf{BQP}_{\mathsf{CTC}} \subseteq \mathsf{PSPACE}$ is more involved and amounts to proving that fixed points of quantum channels are computable in PSPACE.

Conclusion

The work of Aaronson and Watrous highlights an interesting connection between the theory of computation and the structure of spacetime. It also shows that there is no quantum advantage if one has access to CTCs and polynomial time computations. This makes access to CTCs a more powerful resource than quantum computation.

Some open question Aaronson and Watrous mention are the study of bounded CTCs, where the information one can send back in time is limited (e.g. to a single bit) and using CTCs in other settings such as communication complexity or with different kind of automata.

While time travel itself is still only understood in special cases and it's not clear whether there is any chance of it being realizable, some consequences of time travel can still be studied. While this might not lead to any application anytime soon, it still gives some insights in computational complexity theory and the role of time therein. Furthermore, studying the possible effects of CTCs might also help understand why they have not been observed so far (and maybe never will).

References

[1] S. Aaronson and J. Watrous, "Closed timelike curves make quantum and classical computing equivalent," *Proc. R. Soc. A,* vol. 465, no. 2102, 2009.

[2] F. Echeverria, G. Klinkhammer and K. S. Thorne, "Billiard balls in wormhole spacetimes with closed timelike curves: Classical theory," *Phys. Rev. D,* vol. 44, no. 4, 1991.

[3] D. Deutsch, "Quantum mechanics near closed timelike lines," *Phys. Rev. D,* vol. 44, no. 10, 1991.