Is Time Computable?

Sami Al-Suwailem[1]

[1]Affiliation not available

March 24, 2023

Abstract

We argue that a global time index may not be computable. Time and simultaneity can be computable only locally.

Is Time Computable?

Sami Al-Suwailem

March 24, 2023

Abstract

We argue that a global time index may not be computable. Time and simultaneity can be computable only locally.

Time and the Halting Problem

The Halting Problem, at heart, is a time problem. It asks the following question: Can we classify computer programs, by looking at their code, into those that execute and stop (or halt), and those that keep running forever? This problem is unsolvable: There is no algorithm or program that can tell upfront if an arbitrary program shall halt or not. So, basically, it is a problem of the time needed for a program to execute and halt.

We may generalize the Halting Problem as follows. Suppose we want to rank all computer programs according to the time they take to execute their code. We do this using a dedicated computer program, call it Global Clock or GC. The GC will rank all programs along a Global Time Index. Does GC exist? Can we compute a global time index? This problem is a generalization of the Halting Problem, and so we expect it to be unsolvable as well.

Theorem: A global time index is not computable.

Proof: Let $\mathcal{E}$ denote the universe of all computer programs. The Global Clock is a program in $\mathcal{E}$ that ranks all members of $\mathcal{E}$ along a global time index. For any program $e \in \mathcal{E}$, let $T(e) = n$ be an index that assigns an integer $n \in [1, +\infty)$, indicating how fast each program is to execute its code relative to other programs. For example, if all programs are initiated at $n = 0$, then the first program to execute and halt will be assigned $n = 1$ along the global time

index; the second fastest program will be assigned $n = 2$, and so on. Programs that execute and halt simultaneously will have the same index number n. Non-halting programs are assigned ∞.

Now suppose that we partition the global time index into two exhaustive and mutually exclusive segments:

- $T_1 = \{n | n \in [1, N]\}$ for an arbitrary N, and
- $T_2 = \{n | n \in [N + 1, +\infty)\}$.

Next, let us classify all the programs in $\mathcal{E}$ based on their ranking in these two segments:

- $E_1 = \{e_n \mid T(e_n) \in T_1\}$, the set of all programs that execute within T_1, and
- $E_2 = \{e_n \mid T(e_n) \in T_2\}$, the set of all programs that execute within T_2.

Now, E_1 and E_2 are computed *simultaneously*: once E_1 is computed, E_2 is automatically computed, and vice versa. They are members of $\mathcal{E}$ because the latter involves all executing programs, including the Global Clock. This means that E_1 and E_2 must belong to the same segment, either T_1 or T_2. But this is impossible because T_1 and T_2, and therefore E_1 and E_2, are mutually exclusive. It follows that a computable global time index $T(e)$, and therefore a Global Clock program, cannot exist. Q.E.D.

The absence of a global time implies the absence of a global "now," and, thus, the absence of absolute simultaneity.

Corollary: A global "now" is not computable.

Proof: Recall that simultaneous programs will have the same index number n on the global time index $T(e)$ if the latter exists. Define the function:

$$S(e_i) = \begin{cases} 1 & \text{if } \exists e_i, e_j, e_i \neq e_j, \text{such that } T(e_i) = T(e_j) \\ 0 & \text{otherwise} \end{cases}$$

Suppose we classify all programs in $\mathcal{E}$ into two mutually exclusive sets: the set of simultaneous programs, i.e., those that take the same amount of time to execute, and the set of non-simultaneous programs:

- $E_S = \{e_n \mid S(e_n) = 1\}$, the set of all simultaneous programs, and

- $E_{NS} = \{e_n \mid S(e_n) = 0\}$, the set of all non-simultaneous ones.

As pointed out earlier, the two sets E_S and E_{NS} are identified simultaneously. This implies that $S(E_S) = S(E_{NS}) = 1$. Thus, both sets must belong to E_S. There is no issue if $E_S \in E_S$. But if $E_{NS} \in E_S$, this contradicts the assumption that the two are mutually exclusive. It follows that there is no function $S(e)$ that can determine global simultaneity. Time and simultaneity, therefore, can be computable only locally. Q.E.D.