International Journal of Computing and Optimization
Vol. 5, 2018, no. 1, 5 - 8
HIKARI Ltd, www.m-hikari.com
https://doi.org/10.12988/ijco.2018.833

Why Encubation?

Vladik Kreinovich, Rohan Baingolkar,
Swapnil S. Chauhan, and Ishtjot S. Kamboj

Department of Computer Science
University of Texas at El Paso
500 W. University
El Paso, TX 79968, USA

Abstract

It is known that some algorithms are feasible, and some take too long to be practical/ For example, if the running time of an algorithm is 2^n, where $n = \text{len}(x)$ is the bit size of the input x, then already for $n = 500$, the computation time exceeds the lifetime of the Universe. In computer science, it is usually assumed that an algorithm A is feasible if and only if it is *polynomial-time*, i.e., if its number of computational steps $t_A(x)$ on any input x is bounded by a polynomial $P(n)$ of the input length $n = \text{len}(x)$.

An interesting *encubation* phenomenon is that once we succeed in finding a polynomial-time algorithm for solving a problem, eventually it turns out to be possible to further decrease its computation time until we either reach the cubic time $t_A(x) \approx n^3$ or reach some even faster time n^α for $\alpha < 3$.

In this paper, we provide a possible physics-based explanation for the encubation phenomenon.

Mathematics Subject Classification: 68Q15 83F05 92B20

Keywords: encubation, feasible algorithms, computational complexity, lifetime of the Universe

1 What Is Encubation

Some algorithms are feasible, some are not. It is known that:

- some algorithms are feasible, and
- some take too long to be practical.

For example:

- if the running time of an algorithm is 2^n, where $n = \text{len}(x)$ is the bit size of the input x,
- then already for $n = 500$, the computation time exceeds the lifetime of the Universe.

In computer science, it is usually assumed that an algorithm A is feasible if and only if it is *polynomial-time*. In other words, an algorithm is feasible if its number of computational steps $t_A(x)$ on any input x is bounded by a polynomial $P(n)$ of the input length $n = \text{len}(x)$; see, e.g., [2, 3].

Enculation. An interesting *encubation* phenomenon is that:

- once we succeed in finding a polynomial-time algorithm for solving a problem,
- eventually it turns out to be possible to further decrease its computation time
- until we either reach the cubic time $t_A(x) \approx n^3$ or reach some even faster time n^α for $\alpha < 3$.

How can we explain the encubation phenomenon? How do we explain the fact that the complexity of each formally feasible algorithm is eventually reduced to cubic time?

2 How to Explain Encubation?

Physical reminder. According to modern physics, the Universe has $\approx 10^{90}$ particles; see, e.g., [1, 5].

There are $\approx 10^{42}$ moments of time. The number of moments of time can be obtained if we divide:

- the lifetime of the Universe ($T \approx 20$ billion years)
- by the smallest possible time Δt.

Here, Δt is the time that light passes through the size-wise smallest possible stable particle – a proton.

How many computational steps can we perform? The above amounts means that overall:

- even if each elementary particle is a processor that operates as fast as physically possible,
- the largest possible number of computational steps that we can perform is $10^{90} \cdot 10^{42} = 10^{132}$.

This is the largest possible number of computational steps $t(n)$.

What is largest possible input size. The largest possible *input size* comes if you input 1 bit per unit time. Thus, during the lifetime of the Universe, the largest possible length of the input is $n \approx 10^{42}$ bits.

Main idea behind our explanation. If an algorithm is feasible, then:

- for the largest possible length n of the input
- it should still perform the physically possible number of steps.

This idea indeed explains encubation. For $t(n) \approx n^\alpha$ and $n \approx 10^{42}$ this means that

$$t(n) \approx n^\alpha \le 10^{132}.$$

Thus, we get $\alpha \le \dfrac{132}{42} = \dfrac{22}{7} \approx 3$.

This is exactly what we want to explain.

Comment. Since $\dfrac{22}{7} \approx \pi$, maybe π and not 3 is the actual upper bound?

3 What About Human Computations?

Formulation of the problem. What if instead computability in a computer we consider computability in a human brain?

Let us repeat similar computations for such human computing.

Biological reminder. A human life lasts for ≈ 80 years. Each year has ≈ 30 million second, so overall, we get $\approx 2.4 \cdot 10^9$ seconds.

Brain processing is performed by neurons. Typical neurons involved in thinking and processing data have an operation time about 100 milliseconds, i.e., about 0.1 seconds. Thus, during the lifetime, we have $\approx 2.4 \cdot 10^{10}$ moments of time.

There are about 10^{10} neuron in a brain; for details, see, e.g., [4].

How many computational steps can we perform? Due to the above, overall, if all the neurons are active all the time, we can perform

$$t(n) \approx (2.4 \cdot 10^{10}) \cdot 10^{10} \approx 10^{20}$$

computational steps.

What is largest possible input size. Similarly to the physical case, we can gauge the largest possible size by assuming that enter 1 bit every single moment of time. Thus, the largest input size is $n \approx 10^{10}$.

Let us apply our main idea. Similarly to the physical case, let us check for which α, the number of computational steps $t(n)$ needed to process the largest possible input $n \approx 10^{10}$ does not exceed the largest possible number of computational steps: $t(n) = n^\alpha \leq 10^{20}$.

In this case, we conclude that $\alpha \leq 2$, i.e., that only quadratic-time (and faster) algorithms are feasible in terms of human computations.

Comment. This makes sense: e.g., sorting algorithms that describe how we sort by hand, such as insertion sort, are indeed quadratic-time.

Acknowledgments. This work was supported in part by the US National Science Foundation grant HRD-1242122.

References

[1] R. Feynman, R. Leighton and M. Sands, *The Feynman Lectures on Physics*, Addison Wesley, Boston, Massachusetts, 2005.

[2] V. Kreinovich, A. Lakeyev, J. Rohn and P. Kahl, *Computational Complexity and Feasibility of Data Processing and Interval Computations*, Springer, Kluwer, Dordrecht, 1998. https://doi.org/10.1007/978-1-4757-2793-7

[3] C. H. Papadimitriou, *Computational Complexity*, Pearson, Boston, Massachusetts, 1993.

[4] D. Purves, G. J. Augustine, D. Fitzpatrick, A.-S. LaMania, R. D. Mooney, M. L. Platt, and L. W. White (eds.), *Neuroscience*, Sinauer Associates, Sunderland, Massachusetts, 2012.

[5] K. S. Thorne and R. D. Blandford, *Modern Classical Physics: Optics, Fluids, Plasmas, Elasticity, Relativity, and Statistical Physics*, Princeptn University Press, Princeton, New Jersey, 2017

Received: March 31, 2018; Published: April 25, 2018