Ubiquity Symposium

What is Computation?

Computation and Information

by Ruzena Bajcsy,
Electrical Engineering and Computer Science Department
at the University of California-Berkeley

Editor's Introduction

In this sixth article in the ACM Ubiquity symposium,"What is Computation?" Ruzena Bajcsy of the University of California-Berkeley explains that computation can be seen as a transformation or function of information.

—Editor

Ubiquity Symposium

What is Computation?

Computation and Information

by Ruzena Bajcsy,
Electrical Engineering and Computer Science Department
at the University of California-Berkeley

Computation is a transformation/function applied to information.

In order to answer the question, “What is computation?”, I will rephrase it to “What is Information?”

Information is not only a reflection of reality, such as physical measurements and/or observations, but it can also be synthetic concepts invented by humans as they occur for example, in mathematics.

There are many examples of both types of information and their corresponding computation. While the history of differential and integral calculus goes back to the ancient times, the development of it came from the need to explain celestial mechanics.

On the other hand, Riemannian geometry came about as purely mathematical exercise, almost in protest to Euclidean geometry, which only later found applications such as relativity theory and how to compare different complex shapes.

L. Euler in 1736 was the first who introduced concept of graph theory motivated by his observation of the seven bridges of Knigsberg. Today we use graphs for many artificial intelligence problems, but we also study their complexity properties.

It should be rather clear to the reader the type of computation will be determined by the representation of the information. Hence computation with analytical functions will be very different than finding isomorphism or homomorphism.

The above computations are implicit as oppose to explicit.

Implicit computation is requires to solve a concurrent computation of the function, or of an equation. Explicit computation is iterative and/or recursive.

A good example of such computation is computing *N*! (*n* factorial) or Fibonacci numbers.

A convenient mental model of recursion defines the recursive object (whether that object is an equation, an algorithm, an image, or a rule) in terms of "previously defined" objects of the same class.

Finally, we need to clarify the concept of algorithm as a computational method/mechanisms...

The history of algorithms goes to 9th century Persian mathematician Abu Abdullah Muhammad ibn Musa Al-Khwarizmi. The word stands for all definite procedures that solves problem or perform some tasks. Turing 1936 formalized the concept by introducing Turing machine, which is the foundation of all sequential computing.

There is a class of problems that cannot be computed implicitly, such as decision making, but only algorithmically.

So what is computation?

It depends on the representation of the information upon which the computation has to be performed but also on the given task. Algorithmic computation is most general since it includes implicit, explicit, recursive computation, and decision making.

About the Author

Ruzena Bajcsy is Director of CITRIS and Professor of Electrical Engineering and Computer Sciences at the University of California-Berkeley. Her research interests include tele-immersive environments, computer vision, AI, robotics, and sensor networks.

DOI: 10.1145/1895419.1899473

