Computer science and the recent innovations of the modern society

~ *Ph. D. Professor* **Gheorghe Popescu** *(Academy of Economic Studies)*
~ *Ph. D. Professor* **Veronica Adriana Popescu** *(Academy of Economic Studies)*
~ *Ph. D. Assistant* **Cristina Raluca Popescu** *(University of Bucharest)*

Abstract: *The paper "Computer science and the recent innovations of the modern society" presents the importance of computer science, with the most important historical moments in its evolution, the main theoretical elements of the computation science, computer elements and architecture and the latest innovations in the computer science, such as Artificial Intelligence.*

Introduction

Computer science, or computing science, is the study of the theoretical foundations of information and computation and their implementation and application in computer systems. One well known subject classification system for computer science is the ACM Computing Classification System devised by the Association for Computing Machinery.

Computer science or computing science (sometimes abbreviated CS) is the study of the theoretical foundations of information and computation, and of practical techniques for their implementation and application in computer systems. It is frequently described as the systematic study of algorithmic processes that create, describe, and transform information. According to Peter J. Denning, the fundamental question underlying computer science is, "What can be (efficiently) automated?"

Computer science has many subfields; For example, programming language theory studies approaches to describing

computations, while computer programming applies specific programming languages to solve specific computational problems, and human-computer interaction focuses on the challenges in making computers and computations useful, usable, and universally accessible to people.

The focus of computer science is more on understanding the properties of the programs used to implement software such as games and web-browsers, and using that understanding to create new programs or improve existing ones.

Computer science deals with the theoretical foundations of information and computation, and of practical techniques for their implementation and application.

1. The history of computer science – a brief analysis

The early foundations of what would become computer science predate the invention of the modern digital computer.

➤Machines for calculating fixed numerical tasks, such as the abacus, have existed since antiquity. Wilhelm Schickard built the first mechanical calculator in 1623. Charles Babbage designed a difference engine in Victorian times helped by Ada Lovelace. Around 1900, punch-card machines were introduced. However, all of these machines were constrained to perform a single task, or at best some subset of all possible tasks.

➤During the 1940s, as newer and more powerful computing machines were developed, the term computer came to refer to the machines rather than their human predecessors. As it became clear that computers could be used for more than just mathematical calculations, the field of computer science broadened to study computation in general.

➤Computer science began to be established as a distinct academic discipline in the 1950s and early 1960s. The first computer science degree program in the United States was formed at Purdue University in 1962. Since practical computers became available, many applications of computing have become distinct areas of study in their own right.

➤Although many initially believed it was impossible that computers themselves could actually be a scientific field of study, in the late fifties it gradually became accepted among the greater academic population.

➤It is the now well-known IBM brand that formed part of the computer science revolution during this time. IBM (short for International Business Machines) released the IBM 704 and later the IBM 709 computers, which were widely used during the exploration period of such devices. During the late 1950s, the computer science discipline was very much in its developmental stages, and such issues were commonplace.

➤Time has seen significant improvements in the usability and effectiveness of computer science technology. Modern society has seen a significant shift from computers being used solely by experts or professionals to a more widespread user base.

➤Despite its short history as a formal academic discipline, computer science has made a number of fundamental contributions to science and society. These include:

➤The start of the "digital revolution" which includes the current Information Age and the Internet.

➤A formal definition of computation and computability, and proof that there are computationally unsolvable and intractable problems.

➤The concept of a programming language, a tool for the precise expression of methodological information at various levels of abstraction.

➤In cryptography, breaking the Enigma machine was an important factor contributing to the Allied victory in World War II.

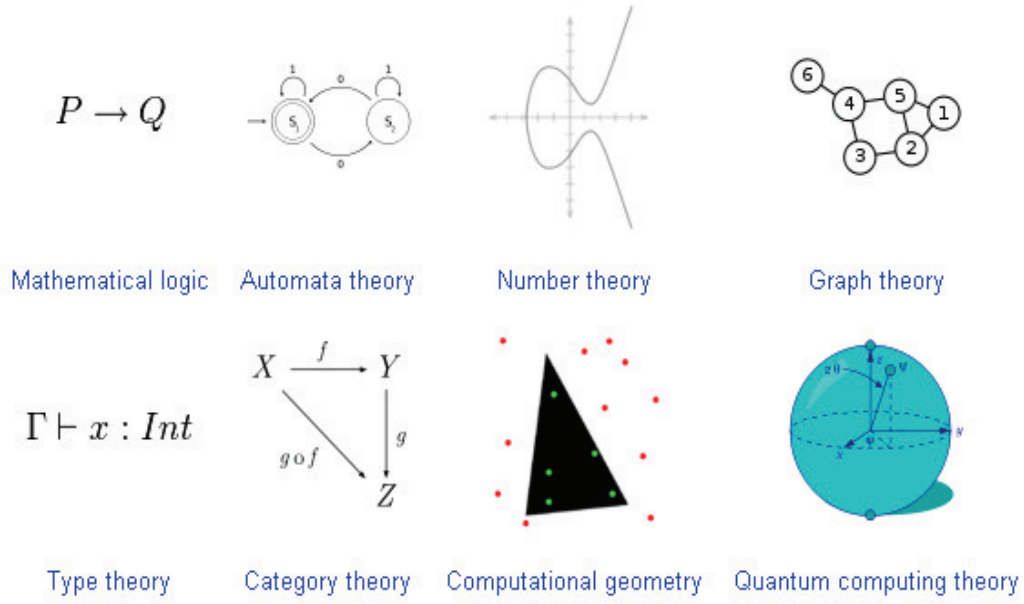
➤Scientific computing enabled advanced study of the mind, and mapping of the human genome became possible with the Human Genome Project.

➤Algorithmic trading has increased the efficiency and liquidity of financial markets by using artificial intelligence, machine learning, and other statistical and numerical techniques on a large scale

2. Theoretical computer science

The broader field of theoretical computer science encompasses both the classical theory of computation and a wide range of other topics that focus on the more abstract, logical, and mathematical aspects of computing.

Figure 1: *"Theoretical computer science and its major elements"*



Source: www.wikipedia.org

3. Theory of computation

The study of the theory of computation is focused on answering fundamental questions about what can be computed and what amount of resources are required to perform those computations.

In an effort to answer the first question, computability theory examines which computational problems are solvable on various theoretical models of computation.

The second question is addressed by computational complexity theory, which studies the time and space costs associated with different approaches to solving a computational problem.

The famous "P=NP?" problem, one of the Millennium Prize Problems, is an open

problem in the theory of computation.

Computability theory, also called recursion theory, is a branch of mathematical logic that originated in the 1930s with the study of computable functions and Turing degrees. The field has grown to include the study of generalized computability and definability. In these areas, recursion theory overlaps with proof theory and effective descriptive set theory.

• Computational complexity theory is a branch of the theory of computation in computer science that focuses on classifying computational problems according to their inherent difficulty. In this context, a computational problem is understood to be a task that is in principle amenable to being solved by a computer. Informally, a computational problem consists of problem instances and solutions to these problem instances. In particular, computational complexity theory determines the practical limits on what computers can and cannot do.

• Closely related fields in theoretical computer science are analysis of algorithms and computability theory. A key distinction between computational complexity theory and analysis of algorithms is that the latter is devoted to analyzing the amount of resources needed by a particular algorithm to solve a problem, whereas the former asks a more general question about all possible algorithms that could be used to solve the same problem. More precisely, it tries to classify problems that can or cannot be solved with appropriately restricted resources. In turn, imposing restrictions on the available resources is what distinguishes computational complexity from computability theory: the latter theory asks what kind of problems can be solved in principle algorithmically.

• Cryptography (or cryptology; from Greek κρυπτός, kryptos, "hidden, secret"; and γράφω, gráphō, "I write", or -λογία, -logia, respectively) is the practice and study of hiding information. Modern cryptography intersects the disciplines of mathematics, computer science, and engineering. Applications of cryptography include ATM cards, computer passwords, and electronic commerce. Until modern times cryptography referred almost exclusively to encryption, which is the process of converting ordinary information (plaintext) into unintelligible gibberish (ciphertext). Decryption is the reverse, in other words, moves from the unintelligible ciphertext back to plaintext. A cipher (or cypher) is a pair of algorithms that create the encryption and the reversing decryption. The detailed operation of a cipher is controlled both by the algorithm and in each instance by a key. This is a secret parameter (ideally known only to the communicants) for a specific message exchange context. Keys are important, as ciphers without variable keys can be trivially broken with only the knowledge of the cipher used and are therefore less than useful for most purposes. Historically, ciphers were often used directly for encryption or decryption without additional procedures such as authentication or integrity checks.

4. Algorithms and data structures

a) Analysis of algorithms

To analyze an algorithm is to determine the amount of resources (such as time and storage) necessary to execute it. Most algorithms are designed to work with inputs of arbitrary length. Usually the efficiency or running time of an algorithm is stated as a function relating the input length to the

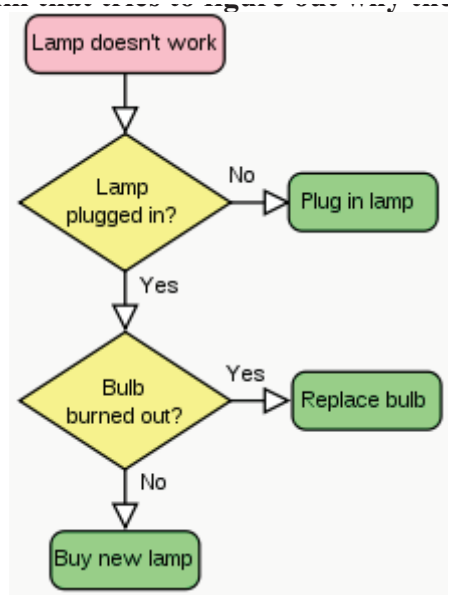
number of steps (time complexity) or storage locations (space complexity).

Algorithm analysis is an important part of a broader computational complexity theory, which provides theoretical estimates for the resources needed by any algorithm which solves a given computational problem. These estimates provide an insight into reasonable directions of search for efficient algorithms.

b) Algorithms

Bellow is an algorithm that tries to figure out why the lamp doesn't turn on and tries to fix it using the steps.

Figure 1: *"Theoretical computer science and its major elements"*



Source: www.wikipedia.org

c) Data structure

In computer science, a data structure is a particular way of storing and organizing data in a computer so that it can be used efficiently. Different kinds of data structures are suited to different kinds of applications, and some are highly specialized to specific tasks. Data structures are used in almost every program or software system. Specific data structures are essential ingredients of many efficient algorithms, and make possible the management of huge amounts of data, such as large databases and internet indexing services. Some formal design methods and programming languages emphasize data structures, rather than algorithms, as the key organizing factor in software design.

Data structures are generally based on the ability of a computer to fetch and store data at any place in its memory, specified by an address — a bit string that can be itself stored in memory and manipulated by the program.

5. Computer elements and architecture

5.1.Computerelements–major elements

a) Digital electronics are systems that represent signals as discrete levels, rather than as a continuous range. In most cases the

number of states is two, and these states are represented by two voltage levels: one near to zero volts and one at a higher level depending on the supply voltage in use. These two levels are often represented as "Low" and "High".

b) Micro architecture: In computer engineering, micro architecture (sometimes abbreviated to μarch or uarch) is the way a given instruction set architecture (ISA) is implemented on a processor. A given ISA may be implemented with different micro architectures. Implementations might vary due to different goals of a given design or due to shifts in technology. Computer architecture is the combination of micro architecture and instruction set design.

c) Multiprocessing is the use of two or more central processing units (CPUs) within a single computer system. The term also refers to the ability of a system to support more than one processor and/or the ability to allocate tasks between them. There are many variations on this basic theme, and the definition of multiprocessing can vary with context, mostly as a function of how CPUs are defined (multiple cores on one die, multiple chips in one package, multiple packages in one system unit, etc.). Multiprocessing sometimes refers to the execution of multiple concurrent software processes in a system as opposed to a single process at any one instant. However, the terms multitasking or multiprogramming are more appropriate to describe this concept, which is implemented mostly in software, whereas multiprocessing is more appropriate to describe the use of multiple hardware CPUs. A system can be both multiprocessing and multiprogramming, only one of the two, or neither of the two.

5.2. Computational science (or scientific computing)

Computational science (or scientific computing) is the field of study concerned with constructing mathematical models and numerical solution techniques and using computers to analyze and solve scientific problems. In practical use, it is typically the application of computer simulation and other forms of computation to problems in various scientific disciplines.

6. Artificial Intelligence

As our world becomes smaller, scientific communities are becoming increasingly international. National scientific societies are evolving to serve their international constituencies, and in doing so, have come to reconsider their roles, their purposes, their images, their identities, their "branding," and, consequently, their names.

This branch of computer science aims to create synthetic systems which solve computational problems, reason and/or communicate like animals and humans do. This theoretical and applied subfield requires a very rigorous and integrated expertise in multiple subject areas such as applied mathematics, logic, semiotics, electrical engineering, philosophy of mind neurophysiology, and social intelligence which can be used to advance the field of intelligence research or be applied to other subject areas which require computational understanding and modeling such as in finance or the physical sciences. It all started with the grandfather of computer science and artificial intelligence, Alan Turing, who proposed the Turing Test for the purpose of answering the ultimate question... "Can computers think?".

By far the greatest danger of Artificial Intelligence is that people conclude too early that they understand it. Of course this problem is not limited to the field of AI. Jacques Monod wrote that a curious aspect of the theory of evolution is that everybody thinks he understands it. The field of Artificial Intelligence (AI) has a reputation for making huge promises and then failing to deliver on them. Most observers conclude that AI is hard; as indeed it is. It is very easy for people to think they know far more about Artificial Intelligence than they actually do. It is very difficult to write about global risks of Artificial Intelligence.

REFERENCES:

1. *7th European Conference on Case Based Reasoning. First Joint Workshop on Computational Creativity*. 30th August - 2nd September, Madrid, Spain. CERSA, 2004.
2. *Advances in Case-Based Reasoning. Proceedings of the 7th European Conference on Case Based Reasoning. Lecture Notes in Artificial Intelligence*, 30th August - 2nd September, Madrid, Spain. Springer, 2004.
3. **Aguado de Cea**, **Guadalupe**, **Asunción Gómez-Pérez**, **Inmaculada Álvarez de Mon y Rego**, and **Antonio Pareja-Lora.** "*OntoTag's Linguistic Ontologies: Improving Semantic Web Annotations for a Better Language Understanding in Machines*." ITCC 2 (2004): 124-128.
4. **Bostrom, Nick**. *"Are You Living in a Computer Simulation?"* Philosophical Quarterly (2003), Vol. 53, No. 211, pp. 243-255. http://www.simulation-argument.com/simulation.html 2009
5. *'Brain' In A Dish Acts As Autopilot, Living Computer*. ScienceDaily 22 Oct. 2004:
6. http://www.sciencedaily.com/releases/2004/10/041022104658.htm 2004-10-25
7. **Crochat, P.** and **Franklin, D.** (2000) *Back-Propagation Neural Network Tutorial*. http://ieee.uow.edu.au/~daniel/software/libneural/
8. **Deacon, T.** 1997. *The symbolic species: The co-evolution of language and the brain*. New York: Norton.
9. **Hibbard, B.** 2004. *Reinforcement learning as a Context for Integrating AI Research.* Presented at the 2004 AAAI Fall Symposium on Achieving Human-Level Intelligence through Integrated Systems and Research.
10. *** www.wikipedia.org
11. *** www.aaai.org
12. *** www.elsevier.com/locate/artint