Computer science

Computer science, or **computing science**, is the study of the theoretical foundations of information and computation and their implementation and application in computer systems. Computer science has many sub-fields; some emphasize the computation of specific results (such as computer graphics), while others relate to properties of computational problems (such as computational complexity theory). Still others focus on the challenges in implementing computations. For example, programming language theory studies approaches to describing computations, while computer programming applies specific programming languages to solve specific computational problems with solutions. A further subfield, human-computer interaction, focuses on the challenges in making computers and computations useful, usable and universally accessible to people.

History

The history of computer science predates the invention of the modern digital computer by many centuries. 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 (between 1837 and 1901) helped by Ada Lovelace . Around 1900 the IBM corporation sold punch-card machines . 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 1960s, with the creation of the first computer science departments and degree programs. Since practical computers became available, many applications of computing have become distinct areas of study in their own right.

Major achievements

German military used the Enigma machine during World War II for communication they thought to be secret. The large-scale decryption of Enigma traffic at Bletchley Park was an important factor that contributed to Allied victory in WWII.

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

Applications within computer science

- 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

Applications outside of computing

- Sparked the Digital Revolution which led to the current Information Age
- 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 the human genome was possible with Human Genome Project.

Relationship with other fields

Despite its name, much of computer science does not involve the study of computers themselves. Because of this several alternative names have been proposed. Danish scientist Peter Naur suggested the term datalogy, to reflect the fact that the scientific discipline revolves around data and data treatment, while not necessarily involving computers. The first scientific institution applying the datalogy term was DIKU, the Department of Datalogy at the University of Copenhagen, founded in 1969, with Peter Naur being the first professor in datalogy. The term is used mainly in the Scandinavian countries. Also, in the early days of computing, a number of terms for the practitioners of the field of computing were suggested in the *Communications of the ACM* -- *turingineer*, *turologist*, *flow-charts-man*, *applied meta-mathematician*, and *applied epistemologist* . Three months later in the same journal, *comptologist* was suggested, followed next year by *hypologist* . Recently the term *computics* has been suggested .

In fact, the renowned computer scientist Edsger Dijkstra is often quoted as saying, *"Computer science is no more about computers than astronomy is about telescopes."* The design and deployment of computers and computer systems is generally considered the province of disciplines other than computer science. For example, the study of computer hardware is usually considered part of computer engineering, while the study of commercial computer systems and their deployment is often called information technology or information systems. Computer science is sometimes criticized as being insufficiently scientific, a view espoused in the statement *"Science is to computer science as hydrodynamics is to plumbing"* credited to Stan Kelly-Bootle and others. However, there has been much cross-fertilization of ideas between the various computer-related disciplines. Computer science research has also often crossed into other disciplines, such as artificial intelligence, cognitive science, physics (see quantum computing), and linguistics.

Computer science is considered by some to have a much closer relationship with mathematics than many scientific disciplines . Early computer science was strongly influenced by the work of mathematicians such as Kurt Gödel and Alan Turing, and there continues to be a useful interchange of ideas between the two fields in areas such as mathematical logic, category theory, domain theory, and algebra.

The relationship between computer science and software engineering is a contentious issue, which is further muddied by disputes over what the term "software engineering" means, and how computer science is defined. David Parnas, taking a cue from the relationship between other engineering and science disciplines, has claimed that the principal focus of computer science is studying the properties of computation in general, while the principal focus of software engineering is the design of specific computations to achieve practical goals, making the two separate but complementary disciplines.

Fields of computer science

Computer science searches for concepts and formal proofs to explain and describe computational systems of interest. As with all sciences, these theories can then be utilised to synthesize practical engineering applications, which in turn may suggest new systems to be studied and analysed. While the ACM Computing Classification System can be used to split computer science up into different topics of fields a more descriptive break down follows:

Mathematical foundations

Mathematical logic
: Boolean logic and other ways of modeling logical queries; the uses and limitations of formal proof methods.

Number theory
: Theory of proofs and heuristics for finding proofs in the simple domain of integers. Used in cryptography as well as a test domain in artificial intelligence.

Graph theory
: Foundations for data structures and searching algorithms.

Type Theory
: Formal analysis of the types of data, and the use of these types to understand properties of programs — especially program safety.

Category Theory
: Category theory provides a means of capturing all of math and computation in a single synthesis.

Computational geometry
: The study of algorithms to solve problems stated in terms of geometry

Theory of computation

Main article: Theory of computation

Automata theory

Different logical structures for solving problems.

Computability theory

What is calculable with the current models of computers. Proofs developed by Alan Turing and others provide insight into the possibilities of what can be computed and what can not.

Computational complexity theory

Fundamental bounds (especially time and storage space) on classes of computations.

Quantum computing theory

Representation and manipulation of data using the quantum properties of particles and quantum mechanism.

Algorithms and data structures

Analysis of algorithms

Time and space complexity of algorithms.

Algorithms

Formal logical processes used for computation, and the efficiency of these processes.

Data structures

The organization of and rules for the manipulation of data.

Programming languages and compilers

Compilers and interpreters

Ways of translating computer programs, usually from higher level languages to lower level ones.

Programming languages

Formal language paradigms for expressing algorithms, and the properties of these languages (e.g. what problems they are suited to solve).

Concurrent, parallel, and distributed systems

Concurrency

The theory and practice of simultaneous computation; data safety in any multitasking or multithreaded environment.

Distributed computing

Computing using multiple computing devices over a network to accomplish a common objective or task and thereby reducing the latency involved in single processor contributions for any task.

Parallel computing

Computing using multiple concurrent threads of execution.

Software engineering

Algorithm design

Using ideas from algorithm theory to creatively design solutions to real tasks

Computer programming

The practice of using a programming language to implement algorithms

Formal methods

Mathematical approaches for describing and reasoning about software designs.

Reverse engineering

The application of the scientific method to the understanding of arbitrary existing software

Software engineering

The principles and practice of designing, developing, and testing programs, as well as proper engineering practices.

System architecture

Computer architecture

The design, organization, optimization and verification of a computer system, mostly about CPUs and Memory subsystem (and the bus connecting them).

Computer organization

The implementation of computer architectures, in terms of descriptions of their specific electrical circuitry

Operating systems

Systems for managing computer programs and providing the basis of a useable system.

Communications

Computer audio

Algorithms and data structures for the creation, manipulation, storage, and transmission of digital audio recordings. Also important in voice recognition applications.

Networking

Algorithms and protocols for reliably communicating data across different shared or dedicated media, often including error correction.

Cryptography

Applies results from complexity, probability and number theory to invent and break codes.

Databases

Data mining

Data mining is the extracting of the relevant data from all the sources of data

Relational databases

Study of algorithms for searching and processing information in documents and databases; closely related to information retrieval.

Artificial intelligence

Artificial intelligence
: The implementation and study of systems that exhibit an autonomous intelligence or behaviour of their own.

Artificial Life
: The study of digital organisms to learn about biological systems and evolution.

Automated reasoning
: Solving engines, such as used in Prolog, which produce steps to a result given a query on a fact and rule database.

Computer vision
: Algorithms for identifying three dimensional objects from one or more two dimensional pictures.

Machine learning
: Automated creation of a set of rules and axioms based on input.

Natural language processing/Computational linguistics
: Automated understanding and generation of human language

Robotics
: Algorithms for controlling the behavior of robots.

Visual rendering (or Computer graphics)

Computer graphics
: Algorithms both for generating visual images synthetically, and for integrating or altering visual and spatial information sampled from the real world.

Image processing
: Determining information from an image through computation.

Human-Computer Interaction

Human computer interaction
: The study of making computers and computations useful, usable and universally accessible to people, including the study and design of computer interfaces through which people use computers.

Scientific computing

Bioinformatics
: The use of computer science to maintain, analyse, and store biological data, and to assist in solving biological problems such as Protein folding, function prediction and Phylogeny.

Cognitive Science
: Computational modelling of real minds

Computational chemistry
: Computational modelling of theoretical chemistry in order to determine chemical structures and properties

Computational neuroscience
: Computational modelling of real brains

Computational physics
: Numerical simulations of large non-analytic systems

Numerical algorithms
: Algorithms for the numerical solution of mathematical problems such as root-finding, integration, the solution of ordinary differential equations and the approximation/evaluation of special functions.

Symbolic mathematics
: Manipulation and solution of expressions in symbolic form, also known as Computer algebra.

Computer science education

Some universities teach computer science as a theoretical study of computation and algorithmic reasoning. These programs often feature the theory of computation, analysis of algorithms, formal methods, concurrency theory, databases, computer graphics and systems analysis, among others. They typically also teach computer programming, but treat it as a vessel for the support of other fields of computer science rather than a central focus of high-level study.

Other colleges and universities, as well as secondary schools and vocational programs that teach computer science, emphasize the practice of advanced computer programming rather than the theory of algorithms and computation in their computer science curricula. Such curricula tend to focus on those skills that are important to workers entering the software industry. The practical aspects of computer programming are often referred to as software engineering. However, there is a lot of disagreement over what the term "software engineering" actually means, and whether it is the same thing as programming.