



Seminar report

History of Computing: Donald Knuth

Pyry Mäkinen

February 13, 2023

FACULTY OF SCIENCE
UNIVERSITY OF HELSINKI

Contact information

P. O. Box 68 (Pietari Kalmin katu 5)
00014 University of Helsinki, Finland

Email address: info@cs.helsinki.fi

URL: <http://www.cs.helsinki.fi/>

Tiedekunta — Fakultet — Faculty Faculty of Science		Koulutusohjelma — Utbildningsprogram — Study programme	
Tekijä — Författare — Author Pyy Mäkinen			
Työn nimi — Arbetets titel — Title History of Computing: Donald Knuth			
Ohjaajat — Handledare — Supervisors Patrik Floreen			
Työn laji — Arbetets art — Level Seminar report	Aika — Datum — Month and year February 13, 2023	Sivumäärä — Sidoantal — Number of pages 11 pages	
Tiivistelmä — Referat — Abstract Donald Knuth is an influential and widely recognized computer scientist and mathematician born in the United States in 1938. He is best known for his multi-volume work "The Art of Computer Programming" as well as many other important contributions to various fields of computer science, including analysis of algorithms and programming language theory. He also created the popular TeX typesetting system. Richard Karp called him the "founder of the fields of data structures and analysis of algorithms". In this text we'll present a short historical overview of Knuth's rise to becoming one of the most widely recognized computer scientists in the world. ACM Computing Classification System (CCS)			
Avainsanat — Nyckelord — Keywords biography			
Säilytyspaikka — Förvaringsställe — Where deposited Helsinki University Library			
Muita tietoja — övriga uppgifter — Additional information			

Contents

1	Introduction	1
2	Studies	3
3	Works	6
3.1	The Art of Computer Programming	6
3.2	Formal language theory	7
3.2.1	LR parsing	7
3.2.2	Attribute grammars	8
3.3	Surreal Numbers	8
3.4	TeX and METAFONT	8
3.5	Exact string matching	8
4	Current Day	10
	Bibliography	11

1 Introduction

Donald Knuth is an influential and widely recognized computer scientist and mathematician born in the United States in 1938. He is best known for his multi-volume work "The Art of Computer Programming" as well as many other important contributions to various fields of computer science, including analysis of algorithms and programming language theory. He also created the popular TeX typesetting system, which is widely used in academic papers and books, along with METAFONT, a system for mathematically defining the shapes of different symbols in a font. Richard Karp called him the "founder of the fields of data structures and analysis of algorithms" [7]. He's currently a Professor Emeritus at Stanford University.

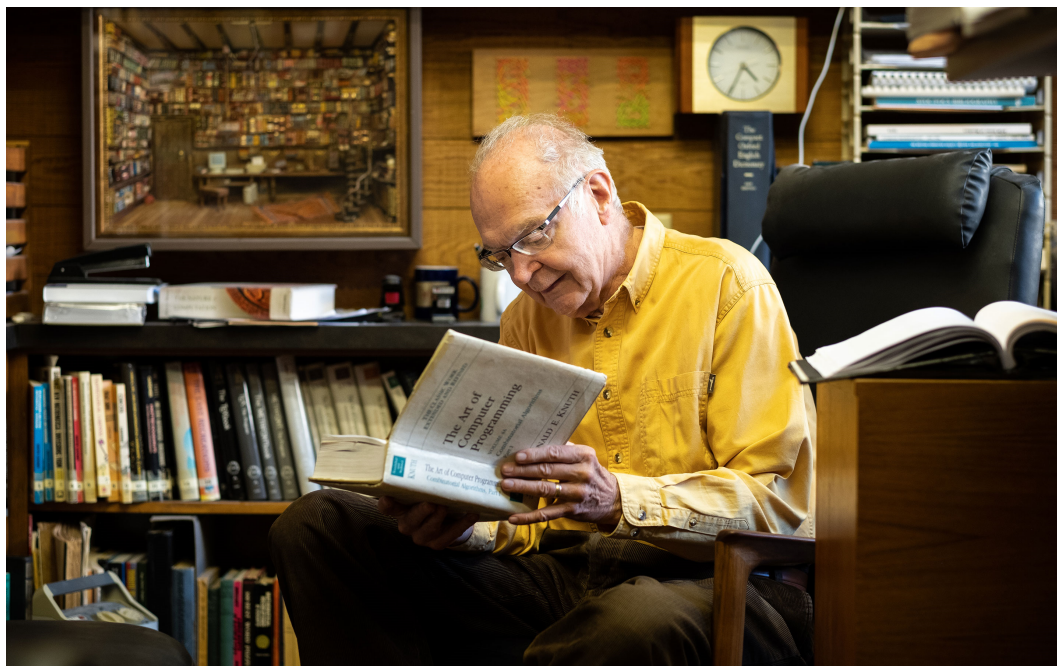
There's a great amount of breadth and variety in Knuth's works. For example, he has written about ancient Babylonian algorithms as well as a fictional novel where he introduced a mathematical construct [10, 11]. Besides mathematics and computer science, Knuth has a great passion for music. In his free time, he is a composer and an organist [1, 9, 10].

Knuth's most influential works include his 1965 work on algebraic systems "Finite Semifields and Projective Planes", his 1965 and 1968 papers on formal language theory "On the Translation of Languages from Left to Right" and "Semantics of context-free languages", as well as "The Art of Computer Programming" book series [3].

Knuth has been awarded many awards and honors including the Turing award, which is often called the Nobel of computer science, in 1974 [5] and the National Medal of Science in 1979 [11]. Knuth is humble about his accomplishments and forgoes a trophy cabinet in favor of using the bowl which commemorates his Turing Award to hold fruit [10].

Knuth is sometimes called the greatest computer programmer of all time. Anecdote from Alan Kay [10]:

When I was at Stanford with the AI project [in the late 1960s] one of the things we used to do every Thanksgiving is have a programming contest with people on research projects in the Bay area. The prize I think was a turkey. McCarthy used to make up the problems. The one year that Knuth entered this, he won both the fastest time getting the program running and he also won the fastest execution of the algorithm. He did it on the worst system with remote batch called the Wilbur system. And he basically beat the shit out of everyone. And they asked him, "How could you possibly do this?" And he answered, "When I learned how to program, you were lucky if you got five minutes with the machine a day. If



Knuth in 2020 [2]

you wanted to get the program going, it just had to be written right. So people just learned to program like it was carving in stone. You sort of have to sidle up to it. That's how I learned to program."

2 Studies

Donald Ervin Knuth was born on Jan 10 1938 in Milwaukee, Wisconsin, U.S. His father was a teacher and a church organist [10]. He grew up playing multiple musical instruments including saxophone, tuba, and organ. He has a passion for music and thought that he would be a music major. He says that it appears that a disproportionate amount of people in mathematics are also musicians though he doesn't have a theory as to why exactly that is [1].

I was torn between physics and music, but I enrolled in college as a physics major. I had done a lot of piano playing and some orchestrations, so I didn't know whether I should major in music or physics. My choice of majors was due essentially to the different scholarships I got. The college I chose was better in physics than in music. If I had gone to Valparaiso instead of Case, I would have majored in music.

Knuth was always algorithmically minded. There are several examples of this from his childhood. While an editor for the school newspaper he invented crossword puzzles for it [10]. Once in eighth grade, he won a prize for his school by finding a list of approximately 4500 words by recombining the string "Ziegler's Giant Bar". The judges only had about 2500 on their master list.

He says that when Computer Science began as a discipline in universities it brought together similarly minded people who didn't have a home before [1]. He also differentiates between mathematicians and computer scientists, thinking of them as different groups of people with different ways of thinking. He also differentiates software engineers from computer scientists.

Computer science, I am convinced, exists today in universities because it corresponds to a mode of thinking, a peculiar mind-set that is the computer scientist's way of looking at knowledge. One out of fifty people, say, has this peculiarity.

He also got very interested in mathematics in high school. He was fascinated by graphs and would draw hundreds of them to see the behavior of different functions. However, he was told that he wasn't likely to be able to carry on through mathematics in college despite having scored very well in high school. He was told that one out of three would fail in freshman year of college mathematics [1].

Therefore his main considerations were then music and physics. There was a lot of excitement in the field of physics post-World War II [10]. Upon receiving a scholarship for physics in 1956, he started studying at Case Institute of Technology, though he says that he might have majored in music instead if it was not for the school being better known for physics [1].

In his sophomore year at Case, Knuth took a course in abstract algebra from professor Green [1]. In that course, the professor had given a very difficult problem and said that anyone who solved it would get an automatic 'A'. Knuth found the course very interesting and one day, after having missed a bus, looked into the problem to kill some time. He was able to solve it and the professor lived up to his bargain.

Meanwhile, in physics and chemistry labs, Knuth was having a terrible time [1]. He didn't do well in the labs, to begin with, but then there were welding labs where he couldn't wear his glasses which made things even worse. His other grades were also suffering due to his obsession with Green's abstract algebra course.

Knuth's freshman calculus teacher Paul Guenther was also a big influence: "I had never met a mathematician before. He had a good sense of humor, but no matter what you said to him, he was unimpressed." [10]

Eventually, Knuth switched over to major in mathematics. He says that despite his academic success he always had an inferiority complex [1]. He felt so worried about passing that he obsessively studied his college mathematics book - "Calculus and Analytic Geometry" - working through every problem in the book. He thinks that his inferiority complex helped him in working hard to learn math.

While attending Case Institute of Technology, Knuth had a summer job drawing graphs. There he had access to an IBM 650 computer, which he spent many late nights hacking on [10, 1]. His first computer program, written on punch cards, factored numbers into primes. His second program performed base conversions.

Knuth's third program was a game of tic-tac-toe and an AI that would learn to play it. The learning strategy involved holding all possible positions in memory and updating a value for each position's level of success, much like the concept of Q-tables in reinforcement learning. If the AI won a game, all positions were marked as slightly better, and if it lost, all positions were marked as bad. He also wrote a symbolic AI that played tic-tac-toe perfectly and then pitted the two AIs against each other. After 90 games, the learning AI learned how to draw against the symbolic one.

As Knuth kept working on computer programs he felt motivated by the feeling of being on the cutting edge and being able to write programs better than some professionals, although he admits that the standards were very low at the time. Part of that motivation came from

a desire to prove himself [1].

During his stay at Case, he also developed a formula for rating basketball players, which was supposedly instrumental in the school team's success, although Knuth no longer believes in the formula. The formula was based on statistics over multiple variables, such as field goals, steals, and fumbles. The team's coach noted that the numbers did correlate with his intuition on player performance, but Knuth did not pursue a career in the field of sports statistics.

Upon receiving his bachelor's degree in mathematics in 1960, the faculty was so impressed by Knuth's work that they, in an unprecedented move, awarded him a master's degree simultaneously [9]. He started his Ph.D. at the California Institute of Technology (CalTech) and completed it 3 years later in 1963 [1]. He wrote his thesis, titled "Finite Semifields and Projective Planes", on combinatorial geometry.

3 Works

Knuth says that there's a common thread in the problems he solves - he takes on problems which he thinks he might have a unique background for. He thinks that teaching something to a computer is sort of like the ultimate test of what you know [10].

In general, whatever you're trying to learn, if you can imagine trying to explain it to a computer, then you learn you don't know about the subject. It helps you ask the right questions. It's the ultimate test of what you know.

3.1 The Art of Computer Programming

When Knuth was a graduate student at CalTech he also consulted with the Burroughs Corporation (later Unisys) [10]. His work included hardware and software design, especially for the Algol 60 programming language. Through his work, he was introduced to another legend in the field, the dutch Edsger Dijkstra, who had worked on the first Algol 60 compiler.

Knuth started working on The Art of Computer Programming in January 1962 when he was a second-year graduate student [1]. Since Knuth had worked on compilers and there wasn't much literature on the subject, publisher Addison-Wesley suggested he write a book about it. He took them them up on the offer.

In fall 1962 he taught a course at CalTech and used the class notes as material for his book. In 1963 he received his PhD and became an assistant professor of mathematics at CalTech. Afterward, he started working on the chapter on sorting which he felt was relevant to compilers, especially given recent work on the LARC Scientific Compiler that he had read about. Initially he didn't know all that much about sorting.

During his research for the sorting section, he found that the existing literature was poor in quality, especially when it came to algorithm performance. Often published articles were just wrong. The literature was already large, but unreliable. Knuth says that: "publications were so bad, in fact, that people didn't even bother reading them".

In the 60s other computer scientists were content with relatively inexact expressions of algorithm time and space complexity, such as simply stating that an algorithm takes time proportional to the square of its input. Knuth gave more exact complexity relations such as proving that an algorithm takes exactly 3.65 times the square of the input [10]. Due to

his work on complexity, he has been called the father of the analysis of algorithms [7] and credited with popularizing the asymptotic notation.

The book grew in scope rapidly following Knuth finding so many interesting things on sorting. Eventually, Knuth asked Addison-Wesley if the scope of the book could be expanded and they gave him the green light. It took him 3 and a half years to write the first chapter of ACP. Upon submitting it to Addison-Wesley he learned that the entire 2000 pages of manuscript he had written for the 12 chapters of ACP would be far too much material for even a very long book. They settled into dividing the 12 chapters into 7 separate volumes.

A lot of new research has come out during the last 60 years of writing the ACP series and the scope has expanded even further [1]. Out of the seven planned volumes, to date, Knuth has published three as well as two out of three parts of the fourth volume, combinatorial algorithms. The parts of the fourth volume have been published as separate books, 4A and 4B.

TODO in x he received the turing award. His work in TACP was specifically cited

3.2 Formal language theory

3.2.1 LR parsing

Knuth invented LR(k) parsing in 1965 in relation to the original topic of the Art of Programming, compilers [10]. A parser's job is to match a source string onto grammatical rules of a special kind of deterministic context-free grammar. An LR parser is a bottom-up parser meaning that it starts from the smallest partitions of the source string, known as terminals, and climbs up to construct the parse tree.

A regular context-free grammar may be ambiguous (meaning that the same string matches multiple grammatical patterns). The value of LR grammars is that, when parsed by an LR parser, the grammar can be seen as unambiguous. To achieve this, lookahead and deterministic derivation strategies are employed. An LR(k) parser looks ahead k terminal symbols to resolve ambiguities. The letter L comes from terminals being consumed from left to right. A derivation strategy determines in which order non-terminals of a grammatical rule are derived. In an LR(k) parser the rightmost rule is derived first, which is where the R in LR comes from (rightmost derivation).

3.2.2 Attribute grammars

Following his work on parsing, Knuth worked on the problem of augmenting non-terminals with additional instructions that are included in the parsed form, the Abstract Syntax Tree (AST), that is used as an intermediary form in compilers [10]. Knuth worked on attribute grammars along with Peter Wegner.

3.3 Surreal Numbers

Knuth wrote the fictional novel "Surreal Numbers: How Two Ex-Students Turned on to Pure Mathematics and Found Total Happiness" in one week while on a sabbatical in Oslo, Norway in 1974 [1]. The construction of surreal numbers had been discovered by John Horton Conway who explained it to Knuth over lunch in 1972. The revolutionary new approach motivated Knuth's book. It was unprecedented to introduce a major mathematical discovery in a work of fiction. The story has the two characters discover Conway's number system through many false starts and frustrations - a process meant to mirror the real process of mathematical discovery.

3.4 TeX and METAFONT

In spring 1977 Knuth started working on typography - a diversion that would take nine years [10]. The work was partially motivated by displeasure with existing systems, but also by his own fascination with typesetting [1]. TeX positions letters and other symbols on the page, while METAFONT defines the shapes of the letters. METAFONT makes use of classical mathematics to design alphabets. Knuth thinks that mathematical exactness is relevant to beauty in typography. He has published several papers on the subject, for example "The Letter S" where he discusses his process of coming up with the equation that creates the most pleasing shape for the title letter.

3.5 Exact string matching

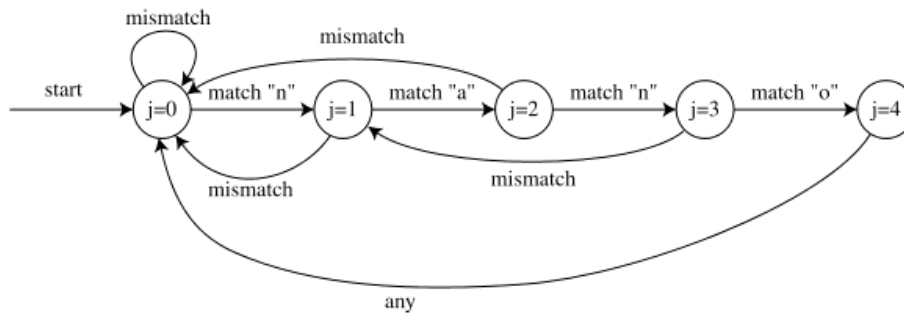
The exact string matching problem tries to answer the following problem: given a string of characters, report all occurrences of a given substring in it [6]. KMP solves the problem without any additional data structures or precomputation.

The obvious way to do it would be to iterate over all characters in the text and at every index

attempt to match the pattern. However, this is very wasteful since we are re-reading characters relative to the length of the pattern for every index. The KMP algorithm eliminates the re-reading such that every character of the text has to only be read once.

Morris developed the algorithm while working on Berkeley Unix, which is now known as BSD. He, together with Vaughan Pratt, published a technical report in 1970, where the algorithm was introduced. It was independently discovered by Knuth, who was inspired by automata theory. They ended up writing a joint paper on the algorithm in 1977.

Knuth was inspired specifically by the stack automaton, which had a recent interesting proof associated with it. A stack automaton is less powerful than a Turing machine, but the proof gives a performance guarantee of $O(n \log n)$ for any language that can be recognized by the automaton. Knuth was fascinated by the result, but couldn't easily think of a problem that could be represented by a stack automaton. As such he developed the KMP algorithm by starting from the solution (stack automaton) and looking for a problem that it could solve.



Finite automaton for matching the pattern "nano" [4]

4 Current Day

Today Donald Knuth is one of the most widely recognized names in computer science. He's a professor emeritus at Stanford, having retired from teaching in 1992 in order to finish the Art of Programming which he is still working on [8]. The fourth volume, on combinatorial algorithms, is planned to consist of three parts. The first part was released in 2011 and the second part came out in October 8th 2022.

Knuth still occasionally holds lectures at Stanford. In 2014 he held a Kailath Lecture at Stanford about meta-history of computer science [9]. He says that in recent years the field of history of computer science appears to have strayed away from technical detail and more into mass appeal "business history". Stanford's website has videos of Knuth's most recent public lectures. Most recently in December 2022 he held a "christmas tree" lecture on Twintrees, Baxter Permutations, and Floorplans.

Bibliography

- [1] D. Albers and G. Alexanderson. *Mathematical People: Profiles and Interviews*. Ak Peters Series. Taylor & Francis, 2008. ISBN: 9781568813400.
- [2] V. Cromwell. In: (2020). URL: <https://www.quantamagazine.org/computer-scientist-donald-knuth-cant-stop-telling-stories-20200416/>.
- [3] D. Knuth / Semantic Scholar. URL: <https://www.semanticscholar.org/author/D.-Knuth/1717349>.
- [4] *Design and Analysis of Algorithms Lecture notes: Knuth-Morris-Pratt string matching*. Feb. 1996. URL: <https://www.ics.uci.edu/~eppstein/161/960227.html>.
- [5] Donald E. Knuth - A.M. Turing Award Laureate. URL: https://amturing.acm.org/award_winners/knuth_1013846.cfm.
- [6] L. Fridman and D. E. Knuth. *219 - Donald Knuth: Programming, Algorithms, Hard Problems the Game of Life*. 2021. URL: <https://youtu.be/EE1R8FYUJm0?t=5056>.
- [7] R. M. Karp. “Combinatorics, Complexity, and Randomness”. In: *Commun. ACM* 29.2 (Feb. 1986), pp. 98–109. ISSN: 0001-0782. DOI: [10.1145/5657.5658](https://doi.org/10.1145/5657.5658). URL: <https://doi.org/10.1145/5657.5658>.
- [8] D. E. Knuth. *Knuth: Frequently Asked Questions*. URL: <https://www-cs-faculty.stanford.edu/~knuth/faq.html>.
- [9] D. E. Knuth. “Let’s Not Dumb Down the History of Computer Science”. In: *Communications of the ACM* 64.2 (Feb. 2021).
- [10] D. Shasha and L. C. *Out of their Minds: The Lives and Discoveries of 15 Great Computer Scientists*.
- [11] J. Woehr. “An Interview with Donald Knuth”. In: (Apr. 1996).