BIRTH:

October 7, 1939, Seattle Washington, USA

EDUCATION:

B.S., Seattle University (1961, Electrical Engineering); M.S., Stanford University (1962 Electrical Engineering); Ph.D., Stanford University (1964, Electrical Engineering).

EXPERIENCE:

Assistant Professor, Princeton University (1964-1967); Cornell University (Associate Professor, 1967-1971; Professor, 1972-85; Joseph C. Ford Professor, College of Engineering, 1985-1994; Chair, Dept. of Computer Science, 1987-92; Associate Dean for College Affairs, 1992-1993; Joseph Silbert Dean, College of Engineering, 1994-2001; Professor, Department of Computer Science, 2001-2004; IBM Professor of Engineering and Applied Mathematics (2004 and onwards).

HONORS AND AWARDS:

National Science Foundation Graduate Fellow, 1961-1964; Association for Computing Machinery A.M. Turing Award (shared with R.E. Tarjan), 1986; Fellow of the American Academy of Arts and Sciences, 1987; Fellow of the American Association for the Advancement of Science, 1987; Fellow of the Institute of Electrical and Electronics Engineers (IEEE), 1987; Member of the National Academy of Engineering, 1989; Fellow of the Association for Computing Machinery, 1994; Life Fellow of IEEE, 2004; Cornell College of Engineering Michael Tien '72 Excellence in Teaching Award, 2004; IEEE Harry Goode Memorial Award, 2005; Assoc. of Computer Science Undergraduates Faculty of the Year Award, 2006; CRA Distinguished Service Award, 2007; ACM Karl V. Karlstrom Outstanding Educator Award, 2008; Honorary professorship, Beijing Institute of Technology, 2008; Fellow of Society for Industrial and Applied Mathematics, 2009; Member of the National Academy of Sciences, 2009; Einstein professor Chinese Academy of Sciences, 2010; IEEE von Neumann Medal, 2010, 2011; Ralph S. Watts 72 Excellence in Teaching Award, 2011; Recognized by the Societe Mathematique de Tunisie (SMT) for "notable services and outstanding contributions in the application of mathematical theories in theoretical computer science", March 2010; Honorary professorship, Yunnan University, 2010;

JOHN E HOPCROFT

United States – 1986

CITATION

With Robert E Tarjan, for fundamental achievements in the design and analysis of algorithms and data structures.

SHORT ANNOTATED BIBLIOGRAPHY | ACM DL AUTHOR PROFILE | ACM TURING AWARD LECTURE | RESEARCH SUBJECTS | ADDITIONAL MATERIALS

John Hopcroft was born into a working class family on October 7, 1939 in Seattle W ashington. His father was a British veteran of the First W orld War who moved to Canada because he was unable to find employment in Britain. He eventually worked his way to the west coast and finally to Seattle, where he met and married John's mother and worked as a janito r.

John grew up in Seattle and attended the local schools. Neither of his parents were high school graduates, but they instilled in him a love of learning and worked hard to ensure that he had the chance to obtain more education than either of them had. At an early age John was fascinated by technology – trains at first, and later mathematics and logic, particularly those aspects relating to electrical engineering.

He claims that because of the lack of family experience with higher education, it never occurred to him to look at other than the local Seattle University . He graduated with a BS in Electrical Engineering in 1961. He was a fine student and was quickly admitted to graduate studies at Stanford University , where he received a MS in 1962 and a PhD in 1964, both in Electrical Engineering.

John's fist academic job was as an Assistant Professor of Electrical Engineering at Princeton. He became a computer scientist almost by accident, when his department head asked him to teach a class in computing science. Since he had no experience in the subject, he had to ask what subjects should be included. The department head recommended a few papers, and John managed to create and teach this first course to six bright students who in turn motivated him. He taught that class for several years, but he says that first year was the most enjoyable because he and his few students were feeling their way into unfamiliar territory together .

One of those first students was Jef f Ullman, who would become a major collaborator in his work. Jef f left Princeton to work at Bell Laboratories and teach an evening course at Columbia University . This was the era when there were few computer science texts, particularly for the theoretical aspects of the subject. Hopcroft and Ullman took John' s course notes and exp anded them into one of the earliest and most influential books on the subject [1]. This volume and its many translations educated an entire generation of computer scientists. Its successor [3] is still in use today.

John's first computer science research ef forts were in an area that became known as formal language theory . Linguists had developed formal grammars for the study of human languages, and similar tools could be used for the developing high level programming languages. The earliest compilers had not been based on a theoretical foundation like that; they were simply a collection of ad-hoc routines that analyzed the statements in a high level language and created the equivalent set of instructions in computer machine code. This process sufficed for the earliest compliers but quickly became unworkable as languages increased in complexity . John made fundamental advances in formal grammars before turning his attention to the study of algorithms.

In the early days of computers, algorithms tended to be compared based on total running time, which depends on the speed of the computer , the programm ing efficiency, the constant overhead of the algorithm, and the growth rate of the running time as the input gets larger . John recognized that as computers became faster and worked on larger problems, the most important part to understand was the last: the growth rate of the running time. His focus on this "asymptotic complexity" set the direction for the new field of analysis of algorithms.

Much of this work dealt with the algorithms used to manipulate graphs, which are made from a set of points (vertices) connected by lines (edges). While this may appear to be a very academic and abstract subject, many practical problems can be described using graphs. Thus any improvements in the general algorithms used to manipulate graphs can provide practical improvements to real life problems. Such problems span the range from linguistics (for grammar and syntactic analysis), to chemistry (for modelling of atoms and molecules to compute their properties), to network analysis (for determining the flow of traf fic in the World Wide Web or in highways).

Hopcroft's work on graph algorithms was not just theoretical analysis, it also included synthesis. He explored efficient structures for storing data in a compu ter, and created ef ficient algorithms for solving the problems they could represent. This work, new and exciting at the time, is now part of the standard computer science curriculum.

His work on formal languages and the analysis of algorithms has made John Hopcroft one of the handful of pioneering computer scientists who put the discipline on a firm theoretical foundation. His co-authored texts on formal languages and their relation to automata [1] and on the design and analysis of algorithms [2] became the standards for a generation of computer scientists.

Honorary professorship, Chongqing University, 2011; Honorary professorship, Jiao Tong University, 2011; Honorary degrees from: Seattle University; University of Sydney; Saint Petersburg State University of Information Technologies, Mechanics & Optics; HKUST; Beijing Institute of Technology; and National College of Ireland.

His current work, which again uses graph models, explores ways to track social networks and build the search engines of the future.

John is not only a respected researcher , but also an inspiring teacher. Each year the top 20 graduates at Cornell are asked to name the professor that had the most influence on their education. John has been chosen twice, a record of which he is justly proud.

He believes that students should be required to take fewer specialist courses. They should be allowed acquire a broad education and have more free time to learn things on their own. He admits that his success in changing university curricula requirements has been limited, but he' s still trying.

Hopcroft has been an influential and inspiring PhD advisor to at least 34 students since 1967. His advisees learned how to do research from him, but they also absorbed his sense of discrimination, his standard of excellence, and his commitment to community service. Many of them now serve in influential positions in academia and industry. Among his PhD stude nts are the president of an Israeli university , a MacArthur Fellow, a vice president of a computing research firm, and many chairs of academic departments.

Throughout his career, Hopcroft has helped t he field by serving on national and international committees. The respect for his views and his work are evident from the many honors he has received.


ABOUT THE A.M. TURING AWARD NOMINA TIONS


VIDEO: THE ORIGINS OF THE A WARD


2015 LAUREA TES: WHITFIELD DIFFIE AND MAR TIN HELLMAN


THE A.M. TURING AWARD LECTURES