Lecture Abstracts

(sorted by date and time)

Monday, September 19

John E. Hopcroft

“Exciting Computer Science Research Directions”
(Monday, September 19, 9:00 a.m.)

We have entered the information age and this has changed the nature of computer science and created many exciting research problems. Two of these are extracting information from large data sources and learning theory. This talk will focus on two problems: first, how to find hidden structure in social networks and second some subareas of research in deep learning.

Sir Michael Atiyah

“The Soluble and the Insoluble”
(Monday, September 19, 9:45 a.m.)

What do we mean by a solution to a problem? This is both a philosophical question, and a practical one, which depends on what one is trying to achieve and the means, time and money available. The explosion in computer technology keeps changing the goal posts. I will reflect on these issues, primarily from the viewpoint of an elderly mathematician.

Brian Schmidt

“State of the Universe”
(Monday, September 19, 11:30 a.m.)

Our Universe was created in 'The Big Bang' and has been expanding ever since. I will describe the vital statistics of the Universe, including its size, weight, shape, age, and composition.
I will also try to make sense of the Universe's past, present, and future – and describe what we know and what we do not yet know about the Cosmos.

Raúl Rojas

“Konrad Zuse‘s Early Computing Machines (1935-1945)”
(Monday, September 19, 12:15 p.m.)

The exhibition "Konrad Zuse‘s Early Computing Machines (1935-1945)" about the machines built by the German inventor Konrad Zuse tells a story. It covers the years 1935 to 1945, which was the most creative phase of his life.
Raúl Rojas gives an overview of the life and work of Konrad Zuse.

Tuesday, September 20

Sir Andrew Wiles

“Equations in arithmetic”
(Tuesday, September 20, 9:00 a.m.)

I will describe some of the interactions between modern number theory and the problem of solving equations in rational numbers or integers'.

Joseph Sifakis

“On the Nature of Computing”
(Tuesday, September 20, 9:45 a.m.)

Computing is a domain of knowledge. Knowledge is truthful information that embedded into the right network of conceptual interrelations can be used to understand a subject or solve a problem. According to this definition, Physics, Biology but also Mathematics, Engineering, Social Sciences and Cooking are all domains of knowledge. This definition encompasses both, scientific knowledge about physical phenomena and engineering knowledge applied to design and build artefacts. For all domains of knowledge, Mathematics and Logic provide the models and their underlying laws; they formalize a priori knowledge which is independent of experience. Computing with Physics and Biology is a basic domain of knowledge. In contrast to the other basic domains, it is rooted in a priori knowledge and deals with the study of information processing – both what can be computed and how to compute it.
To understand and master the world, domains of knowledge share two common methodological principles. They use abstraction hierarchies to cope with scale

problems. To cope with complexity, they use modularity. We point out similarities and differences in the application of these two methodological principles and discuss inherent limitations common to all domains.

In particular, we attempt a comparison between natural systems and computing systems by addressing two issues: 1) Linking physicality and computation; 2) Linking natural and artificial intelligence.

Computing and Physical Sciences share a common objective: the study of dynamic systems. A big difference is that physical systems are inherently synchronous. They cannot be studied without reference to time-space while models of computation ignore physical time and resources. Physical systems are driven by uniform laws while computing systems are driven by laws enforced by their designers. Another important difference lies in the discrete nature of Computing that limits its ability to fully capture physical phenomena. Linking physicality and computation is at the core of the emerging Cyber physical systems discipline. We discuss limitations in bridging the gap between physical and discrete computational phenomena, stemming from the identified differences.

Living organisms intimately combine intertwined physical and computational phenomena that have a deep impact on their development and evolution. They share common characteristics with computing systems such as the use of memory and languages. Compared to conscious thinking, computers are much faster and more precise. This confers computers the ability to successfully compete with humans in solving problems that involve the exploration of large spaces of solutions or the combination of predefined knowledge e.g. AlphaGo's recent winning performance. We consider that Intelligence is the ability to formalize human knowledge and create knowledge by applying rules of reasoning. Under this definition, a first limitation stems from our apparent inability to formalize natural languages and create models for reasoning about the world. A second limitation comes from the fact that computers cannot discover induction hypotheses as a consequence of Gödel's incompleteness theorems.

We conclude by arguing that Computing drastically contributes to the development of knowledge through cross-fertilization with other domains as well as enhanced predictability and designability. In particular, it complements and enriches our understanding of the world with a constructive and computational view different from the declarative and analytic adopted by physical sciences.

Ngô Bảo Châu

"The Riemann zeta function and its generalizations"
(Tuesday, September 20, 11:30 a.m.)

Since the publication of Riemann's memoir on prime numbers less than a given magnitude, the zeta function has never ceased to fascinate mathematicians. We will discuss its many appearances and generalizations in number theory.

Fred Brooks

"What Makes the Illusion Work? Studies in Effective Immersive Virtual Environments"
(Tuesday, September 20, 12:15 p.m.)

A virtual environment (VE) is a technological display to the senses that undertakes to make the user believe (to some degree) and behave as if he is present elsewhere. The vision, proposed by Sutherland in 1965, has driven a half-century of research and development, some at Chapel Hill. A major scientific challenge is how to measure the effectiveness of a VE system, the degree to which it causes the user to behave as if the illusion is real.
We have pursued many measurement methods, arriving finally at physiological measurement of subconscious user responses, a measure that is objective, valid, quantitative, and contemporaneous. Our parameter studies of VE systems show, among other things, that system latency is a major factor, whereas the quality of illumination simulation has little effect. Display to the senses of feel, added to visual and aural, also makes a substantial difference.

Thursday, September 22

Raj Reddy

"Too Much Information and Too Little Time"
(Thursday, September 22, 9:00 a.m.)

This talk is about having to cope with too much information within human time limitations given that we are not changing at exponential rates like semiconductors. Humans make errors, tend to forget, are impatient and look for least effort solutions. Such limitations, sometimes, lead to catastrophic results. At the same time, humans

learn with experience, tolerate error and ambiguity, use vast amounts of knowledge, and communicate using speech and language. Such features are still lacking in most of our systems. Most systems don't get better with experience.
Future opportunities lie in creating tools for coping with 21st century world of "too much information and too little time". In this talk we will present two families of intelligent agents, viz., "cognition amplifiers" and "guardian angels" to help with problem of scarcity of attention. A Cognition Amplifier is a personal autonomic intelligent agent that anticipates what you want to do and helps you to do it with less effort. A Guardian Angel is a personal autonomic intelligent agent that discovers and warns you about unanticipated, possibly catastrophic, events that could impact your safety, security, and wellbeing. Both Cogs and Gats are enduring, autonomic, nonintrusive intelligent agents which are always-on, always working, and always-learning. Future breakthroughs will emerge from those who understand human limitations and can cater to such human needs.

Richard Edwin Stearns

"Strategies for Extensive Form Games"

(Thursday, September 22, 9:45 a.m.)

We want to describe sets of mixed strategies using linear equations such that
1. The number of variables is small compared to the game size.
2. Every mixed strategy has an equivalent strategy in the set.
3. Implementing the strategies is easy.

It is known how to do this for players with perfect recall using equations for behavior strategies or equations for path probabilities. We generalize the perfect recall techniques to cover players without perfect recall. Although the number of variables needed is not always small, it will be small if the recall is close enough to perfect.

Vladimir Voevodsky

"UniMath"

(Thursday, September 22, 11:30 a.m.)

UniMath is a library of formalized mathematics that is based on the Univalent Foundations and written in what we call the UniMath language which is a small subset of the language of the Coq proof assistant.
I wrote the core of the library under the name Foundations in 2010-11 to try out whether the Univalent Foundations can be used with Coq to formalize some real

mathematics. After discovering that it can we, Benedikt Ahrens, Dan Grayson and myself, formed a GitHub organization called UniMath.
Today UniMath has 12 authors, about 7 of them, including all three of the founders, are active in expanding and improving the library.

Leslie Lamport

"The PlusCal Algorithm Language"
(Thursday, September 22, 12:15 p.m.)

An algorithm is not a program, so why describe it with a programming language? PlusCal is a tiny toy-like language that is infinitely more expressive than any programming language because an expression can be any mathematical formula.

Friday, September 23

Barbara Liskov

"The power of abstraction"
(Friday, September 23, 9:00 a.m.)

Abstraction is at the center of much work in Computer Science. It encompasses finding the right interface for a system as well as finding an effective design for a system implementation. Furthermore, abstraction is the basis for program construction, allowing programs to be built in a modular fashion. This talk will discuss how the abstraction mechanisms we use today came to be and how they are supported in programming languages.

Sir Tony Hoare

"A finite geometric representation of computer program behaviour"
(Friday, September 23, 9:45 a.m.)

Scientists often illustrate the behaviour of a dynamic system by a geometric diagram, in which one dimension represents the passage of time, and the other(s) represent distribution of objects in space. We develop a non-metric finite plane geometry as an intuitive representation of the behaviour of a computer program running on a modern distributed network of concurrent processors. Our hope is to prove a collection of algebraic laws that are used for the implementation and optimisation of programs.

Heisuke Hironaka

"Resolution of Singularities in Algebraic Geometry"

(Friday, September 23, 11:30 a.m.)

Algebraic geometry in general has three fundamental types in terms of its base ground: (I) $\mathbb{Q}$ (and its fields extensions), (II) $\mathbb{F}$ (p) with a prime number $p > 0$ (and every finite field), and at last (III) $\mathbb{Z}$ in the case of the arithmetic geometry. In those three cases I will talk about resolution of singularities by means of blowups with permissible centers in smooth ambient spaces. (I) is done in 1964, (II) is proven recently with new concept and technique, while (III) is by combination of (I) and (II). Technically elaborate but conceptually interesting is the case of (II).

Robert Tarjan

"Binary Search Trees"

(Friday, September 23, 12:15 p.m.)

The binary search tree is one of the most fundamental data structures in computer science, with many applications. Binary search trees support binary search in a set of totally ordered items, and ideally reduce search time from linear to logarithmic. A central question is how to keep such a tree balanced in the presence of updates. The first solution was offered by Adelson-Velskii and Landis in 1962. In spite of a huge volume of work during the intervening 64 years, the design space is rich, and basic questions remain open, notably how best to make a search tree adapt to its usage pattern. In this talk I'll explore relatively recent new work and interesting open problems.