Great Ideas

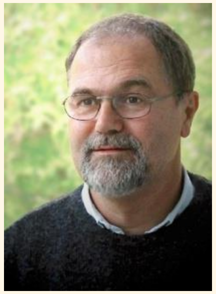
In Theoretical Computer Science

Advanced Course, SS 2013
Saarland University

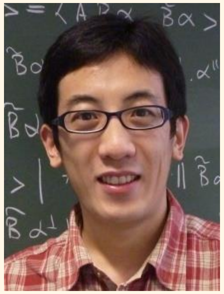
COURSE INFORMATION

- Lectures

  Prof. Kurt Mehlhorn

  Director of Max Planck Institute for Informatics

  Dr. He Sun

  Senior Researcher at Max Planck Institute for Informatics

- Time: Monday 4-6 pm

  First Lecture: April 22

- Location: Room 024, MPII building

  Campus E1.4

Grading

3 Problem Sets.

- You have 2 weeks to work on every problem set individually.
- Problem sets count for 50% for your final grade.
- You need to collect at least 50% points to be eligible for the final exam.

Final Exam (Written).

- Final Exam counts for 50% for your final grade.
- There are questions from every lecture.
- Choose half of the lectures that you like most and answer the corresponding questions.

Reference

- **Lecture Notes**

  Cover everything we discuss in class

- **Related Articles**

  Gain further understanding

- **Blog**

  Join the discussion.
  You will find the area more exciting ☺

BLOG

Great Ideas in Theoretical Computer Science

Advanced Course @ Saarland University

Home Abo

Welcome

Posted on April 15, 2013

Welcome to the course "Great Ideas in Theoretical Computer Science". This course will overview major breakthroughs in theoretical computer science, and highlight their connections to other areas in computer science. In particular, we will discuss great ideas in the past 60 years that (i) provide deep understanding of the world, (ii) give computer scientists intuitions, (iii) have great influence in computer science, and (iv) create excitement.

Starting with the intriguing question about P vs. NP, we will overview our understanding to various aspects of the computation models, like time vs. space, randomized vs. deterministic computation, and finding vs. verifying solutions. We will discuss some of fundamental techniques for designing algorithms, and some fantastic areas in theoretical computer science. This will roughly cover 5 Turing Award winners' work, and 8 Goedel Award papers. Some questions that we will answer during the course: How can we discuss a secret with friends? Is randomness necessary for designing algorithms? What are learning algorithms? Is there any connection between theoretical computer science and algebra, geometry, etc.?

| Leave a comment | Edit

TOPICS

The Millionaire Problem

Streaming Algorithms

Randomness, Hardness, and Approximation

How to talk about a secret?

Cryptography: From Art to Science

Linear Programming

Learning

...and More...

One topic per week

You only need to know basic algorithms before the class

Time vs. Space

Question: What is the 100th digit of pi?

Solution 1: $\pi = 16\tan^{-1}\frac{1}{5} - 4\tan^{-1}\frac{1}{239}$

Solu

Can we speed up algorithms by using smaller space?

Solution 3: Look it up from a dicitonary!

1415926535897932384626433832795028841971693993751058209749445923078164062862089986280348253421170679821480865132823066470938446095505822317253594081284811174502841027019385211055596446229489549303819644288109756659334461284756482337867831652712019091456485669234603486104543266482133936072602491412737245870066063155881748815209209628292540

There is a trade-off between time and space.
This approach works sometimes, but not always.

Is time and space interchangeable?

Do we have a theory about time vs. space like Physics?

P vs. NP

<u>Euler Cycle Problem</u> Given a graph of n vertices, is there a cycle that visits every edge exactly once?

easy to <u>find</u> a solution

<u>Hamiltonian Cycle Problem</u> Given a graph of n vertices, there a cycle that visits every vertex exactly once?

A naïve solution needs n! time.

easy to <u>verify</u> a solution

P vs. NP (Cont)

Easy Problems (1970)

P

Connectivity
Shortest Path
Max Flow
… …

Hard Problems (1970)

NP

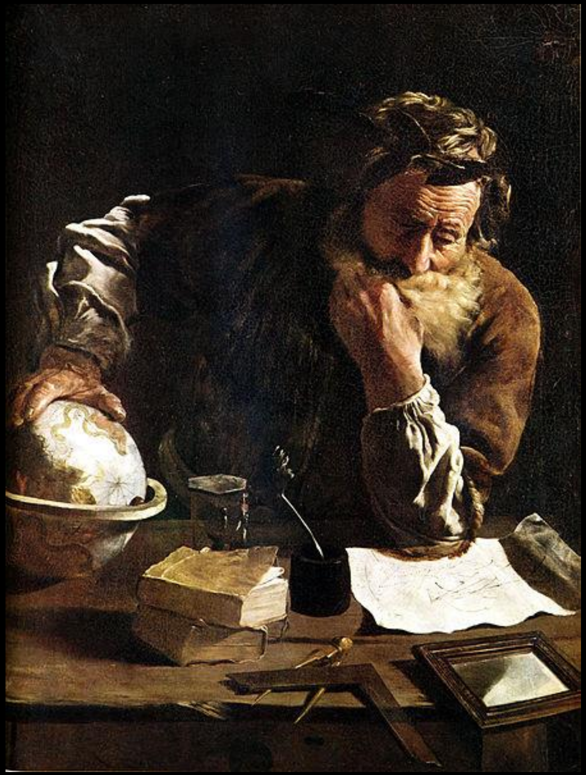
Vertex Cover
Travelling Salesman
Maximum Cut
… …

P: polynomial-time solvable

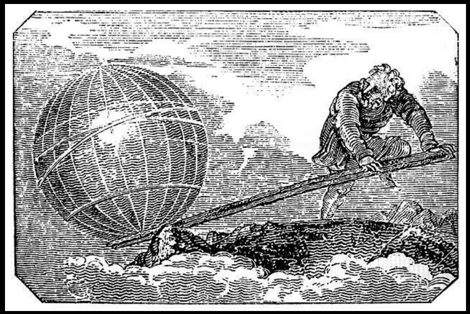
NP: polynomial-time checkable

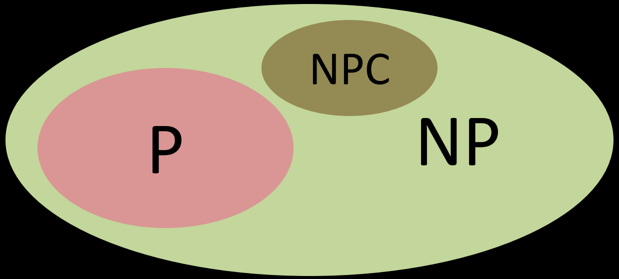
$P \neq NP$?

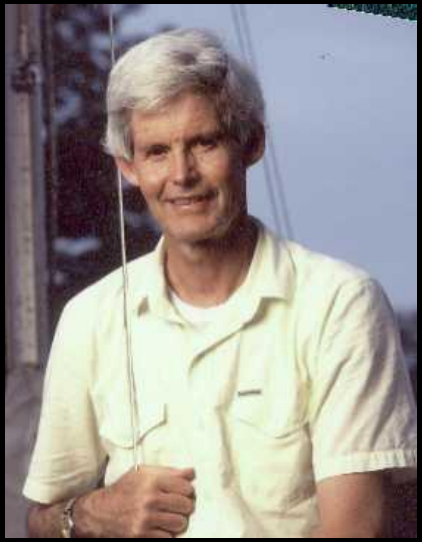
Is there an inherent difference between discovering a solution and verifying a solution?



Give me a place to stand on, and I will move the Earth.
Archimedes 287 BC-212 BC

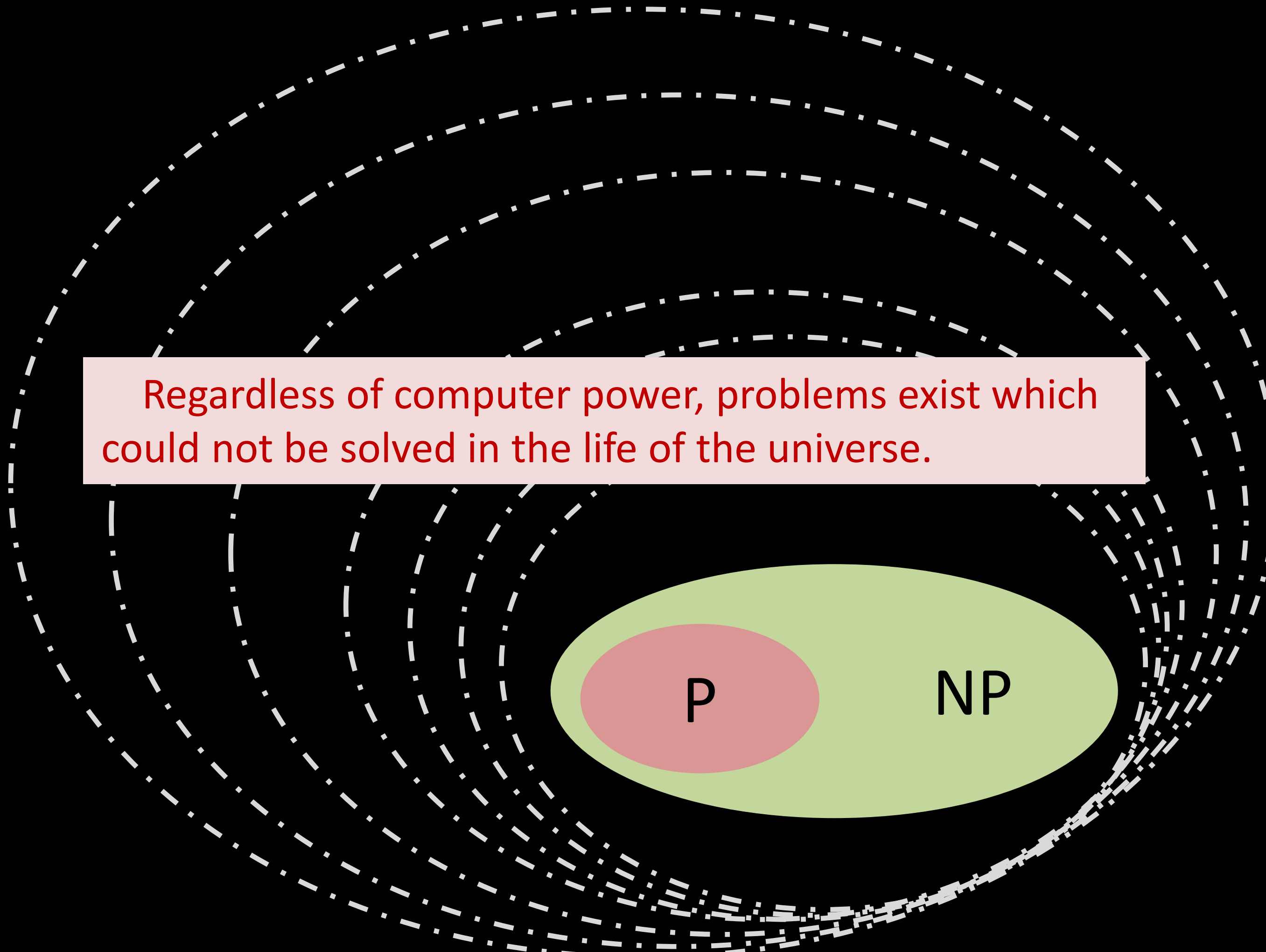






Give me a solution of one hard problem, and I will tell you solutions of thousands of hard problems.
Stephen Cook, 1971

Larry Stockmeyer
(1948-2004)



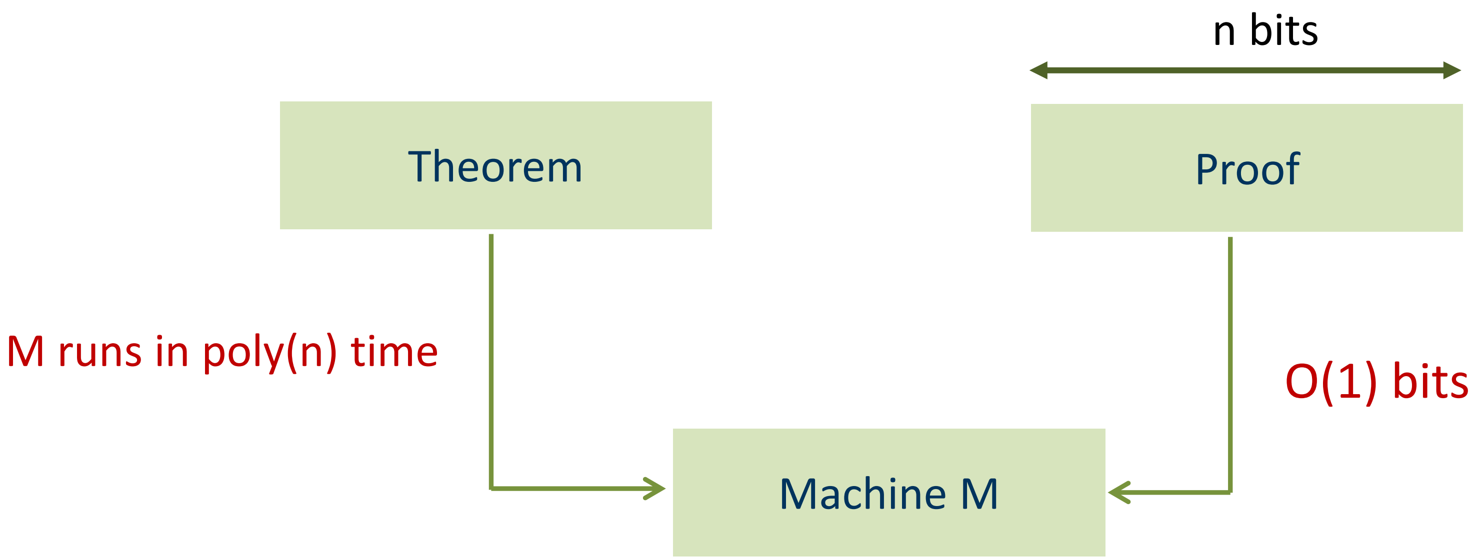
Regardless of computer power, problems exist which could not be solved in the life of the universe.

Probabilistically Checkable Proofs

Probabilistically Checkable Proofs (Cont)

Question: Do we need to read a math proof completely to check it?

PCP: Probabilistically Checkable Proofs



- Theorem correct → there is a proof that M accepts w. prob. 1
- Theorem incorrect → M rejects every claimed proof w. prob 1/2

Probabilistically Checkable Proofs (Cont)

Students write n bits in total.

Professors flip O(log n) coins, and read O(1) fraction of your solution.

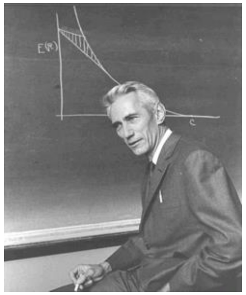
1.0, 1.3, 2.0, 2.3,…

Good for professors

Good for students

Randomness

<u>What's Randomness ?</u>



almost random = high entropy

Claude Shannon (1919-2001)

almost random =
[Length of shortest program to produce x ≈ |x|]

A.N. Kolmogorov (1903-1987)

Randomness (Cont)

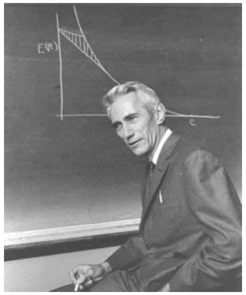
1415926535897932384626433832795028841971693993751058209749445923078164062
28620899862803482534211706798214808651328230664709384460955058223172535940
81284811174502841027019385211055596446229489549303819644288109756659334
461284756482337867831652712019091456485669234603486104543266482133936072
602491412737245870066063155881748815209209628292540

$$\pi = \left(\frac{2\sqrt{2}}{9801} \sum_{k=0}^{\infty} \frac{(4k)!(1103 + 26390k)}{(k!)^4 396^{4k}} \right)^{-1}$$

$$\pi = 16 \tan^{-1} \frac{1}{5} - 4 \tan^{-1} \frac{1}{239}$$

Randomness (Cont)

<u>What's Randomness ?</u>



almost random = high entropy

Claude Shannon (1919-2001)

almost random =
[Length of shortest program to produce x ≈ |x|]

A.N. Kolmogorov (1903-1987)

almost random =
no algorithm can distinguish it apart from truly random strings.

Andrew Yao (1946-)

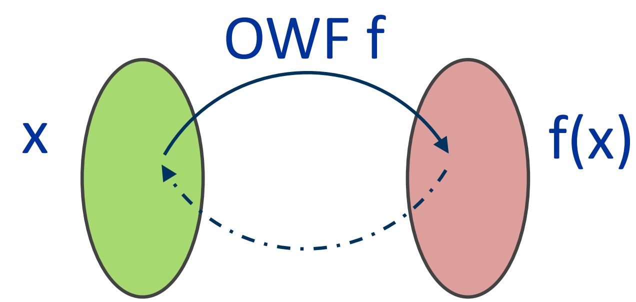
They are almost equivalent.

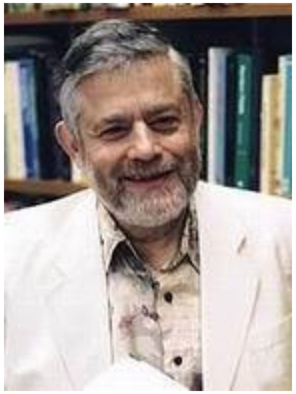
Randomness (Cont)

One-Way Functions

More randomness inside ≈ Harder to describe

010100001101010100111
010101011010001010110
101000101001010111101



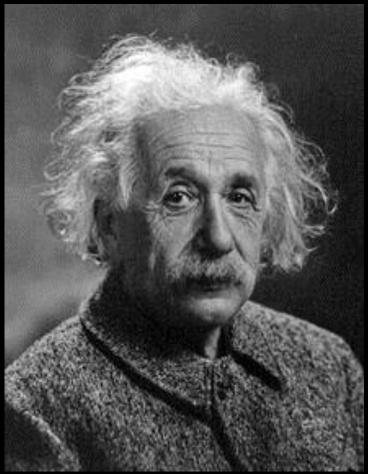


Manuel Blum
(1938 -)

existence of algo. to output such strings
≈ existence of OWF

f is easy to compute
easy to encrypt messages
f^{-1} is hard to compute
hard to decrypt messages, unless you have secrete info.

fundamental tools in modern cryptography



God does not play dice!

Copenhagen, 1928

Corresponding Statement in Computer Science?

Cryptography

Yao's Millionaires' Problem
Two people want to compare who is richer without revealing their actual wealth.

After comparison, they do not know how much money the other has.

You know the result, but you do not know the certificate.

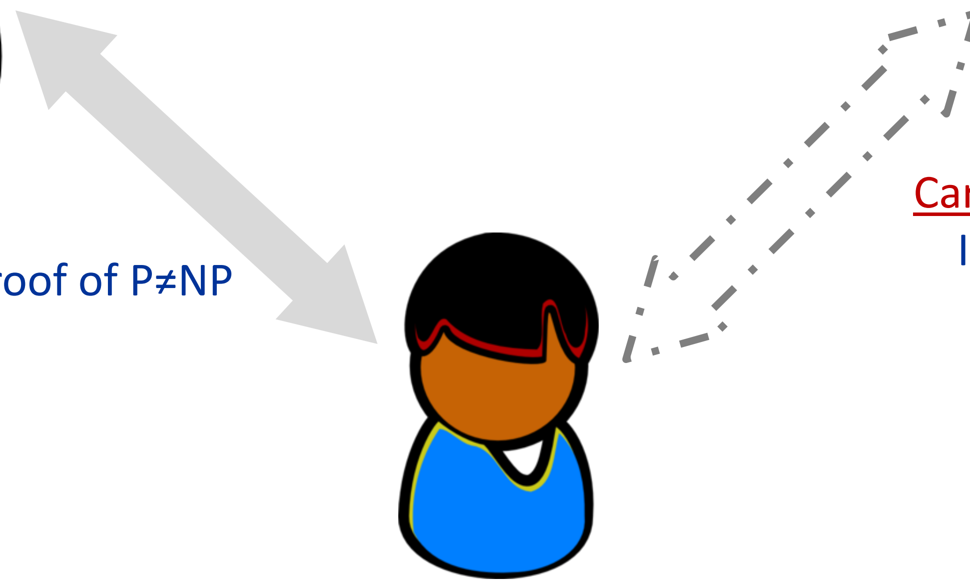
Cryptography (Cont)

Zero Knowledge Proofs

Question: Does a proof inherently carry with it some knowledge or not?

Alice:
I have the proof of P≠NP

Carlie:
I cannot get anything from Bob.

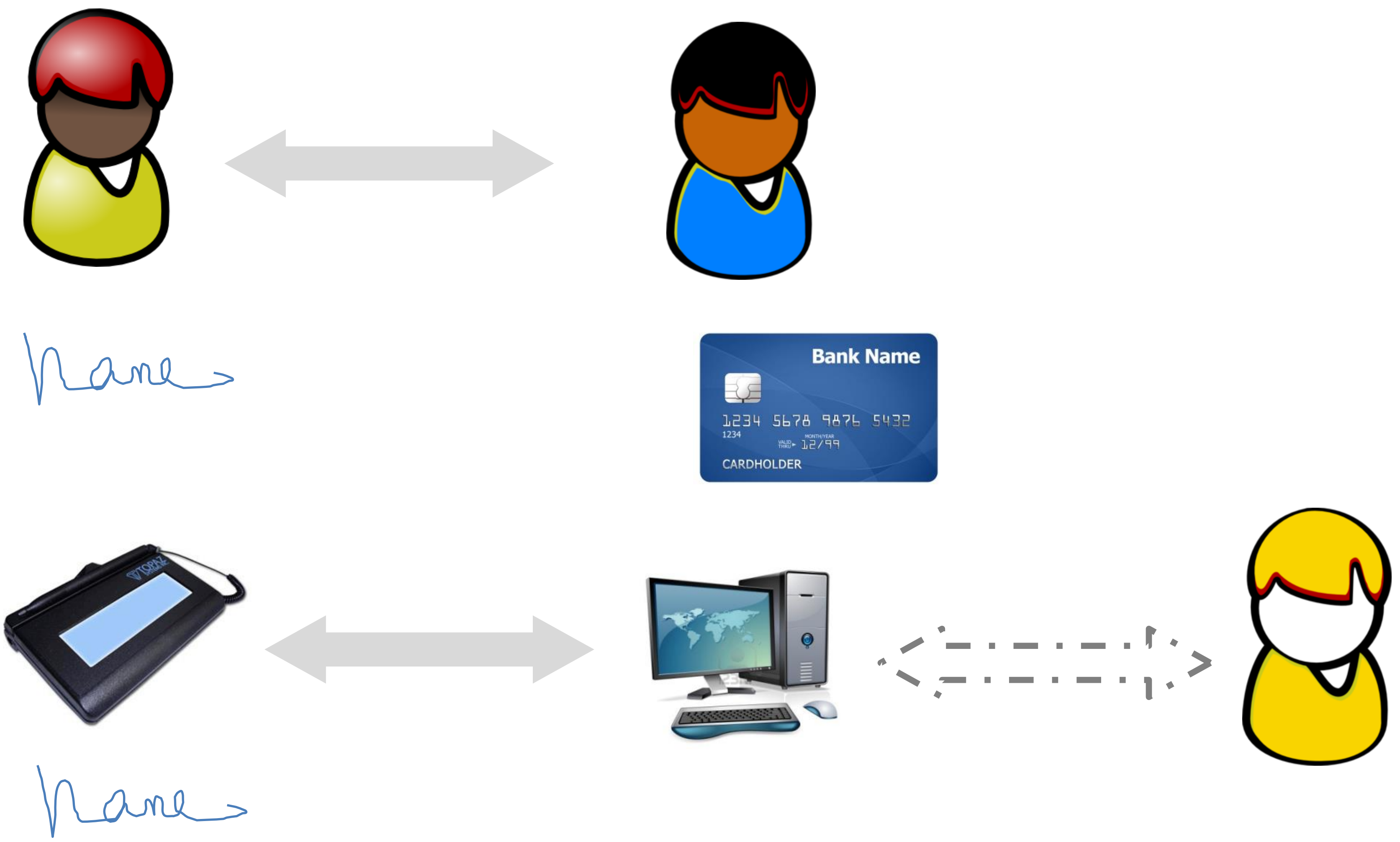


Bob:
I am convinced that Alice proved P≠NP.
But I know nothing about the proof.

Shafi Goldwasser (1958 -)

Cryptography (Cont)

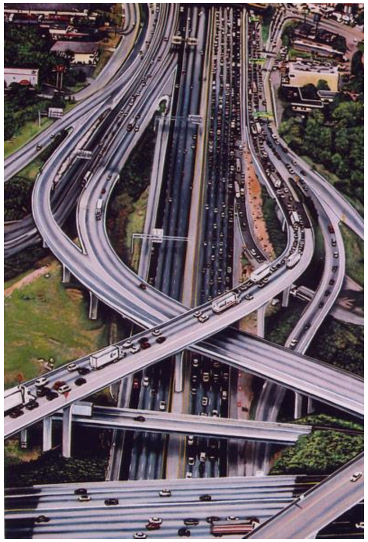
Zero Knowledge Proofs



Expander Graphs

Super Concentrators

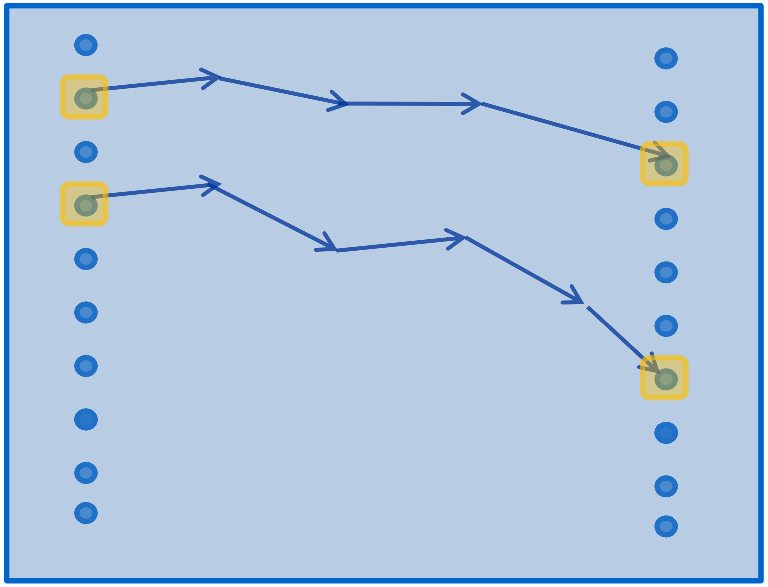
Given n cities in the north and n cities in the south, construct a highway network, such that for any k cities in the north and k cities in the south, there are k disjoint paths.



"disjoint" => efficiency of transportation, no delay

# of edges ⇔ construction cost

Complete bipartite graph is an example, but too expensive.



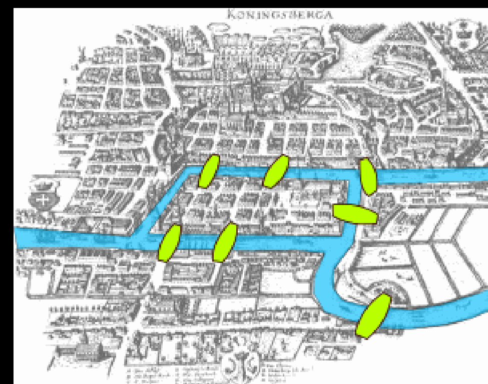
Construct a network (directed graph) with n input nodes and n output nodes, such that for any K input nodes, and any K output nodes, there are K disjoint paths connecting them.

For any n, there is a super concentrator with 28n edges.

Finally, the <u>super concentrators</u> constructed by Valiant in the context of computational complexity established the fundamental role of <u>expander graphs</u> in computation.

2010 ACM Turing Award Citation

Leonhard Euler
(1707-1783)



Seven Bridges of Königsberg, 1736

- Coloring
- matching
- Hamiltonian Cycles
- Spanning Trees

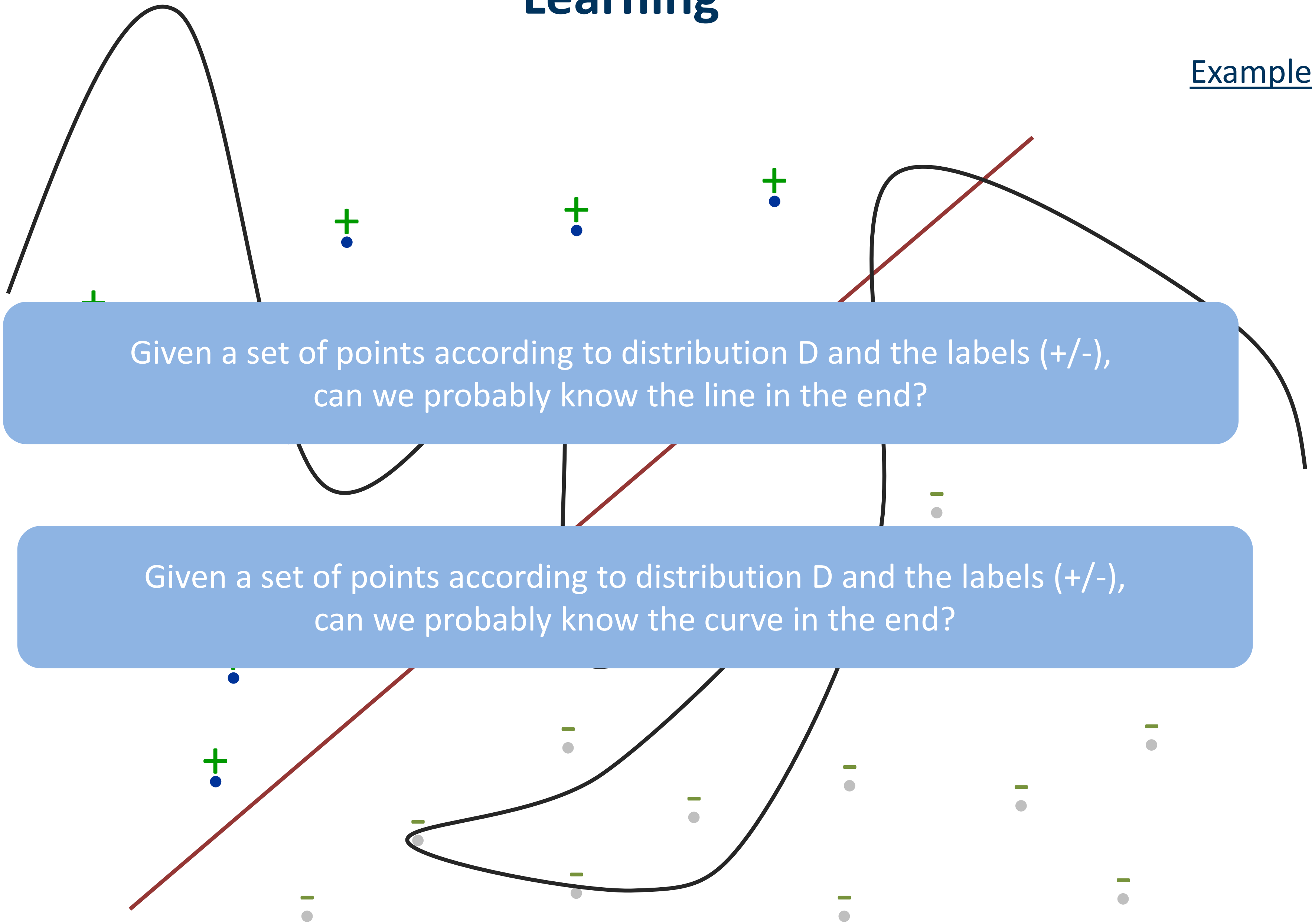
- Algebra
- Geometry
- Topology
- Group Theory

Since 1700s

Since 1970s

Learning

Example



Given a set of points according to distribution D and the labels (+/-), can we probably know the line in the end?

Given a set of points according to distribution D and the labels (+/-), can we probably know the curve in the end?

Learning

- Which properties make these two problems so different?
- What can computers learn?
- Do computers learn in the same way as human beings?
- ……

We will tell you how everything starts.

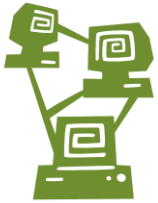
Streaming Algorithms

Motivations

- There are 100 billions web pages.

- There are 3 Billion Telephone Calls in US each day,
  30 Billion emails daily, 1 Billion SMS, IMs.

- Scientific data: NASA's observation satellites generate billions of readings each per day.

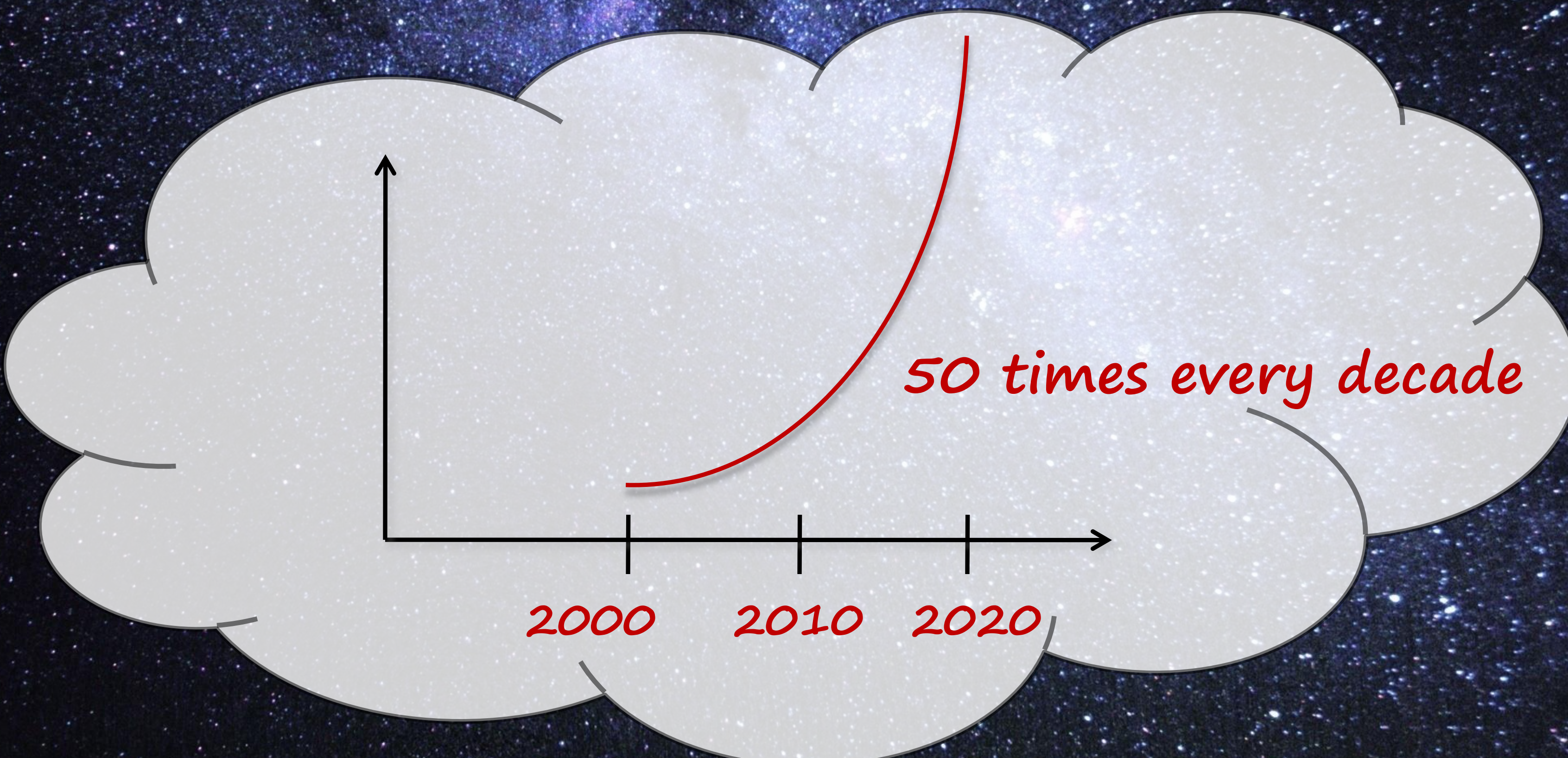


- IP Network Traffic: up to 1 Billion packets per hour per router. Each ISP has many (hundreds) routers!

- Whole genome sequences for many species now available: each megabytes to gigabytes in size.

The digital universe

1.2 ZB = 1.2 * 2^{70} bits/2011



Streaming Algorithms

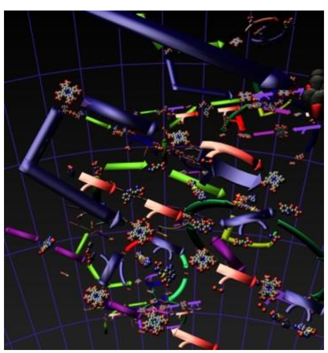
Examples

What we face nowadays

- massive data sets
- Storing the whole data is impossible
- Good approximation suffices

amazon

Amazon can evaluate the popularity of one product by # of different IPs looking at the webpage.



Scientists can detect certain diseases by counting # of certain patterns in a biological network.

Great Ideas

- clearly motivated
- help us understand the world
- have philosophical meaning
- open new areas
- create excitement