CS 121: Lecture 26
What we didn't cover

Madhu Sudan

https://madhu.seas.Harvard.edu/courses/Fall2020

Book: https://introtcs.org

How to contact us
- The whole staff (faster response): [CS 121 Piazza]
- Only the course heads (slower): cs121.fall2020.course.heads@gmail.com

Announcements:

- Advanced section: Yael Kalai (Proofs in CS: Probabilistic Checking, Interaction, Zero Knowledge, Delegation)
- Final Exam (any 3hrs between [1:21pm 12/10 – 1:21pm 12/12])

Where we are:

Part I: Circuits:
Finite computation,
quantitative study

Part II: Automata:
Infinite restricted computation,
quantitative study

Part III: Turing Machines:
Infinite computation, qualitative study

Part IV: Efficient Computation:
Infinite computation, quantitative study

Part V: Randomized computation:
Extending studies to non-classical algorithms

Summary of the course

- Turing Machines!
  - Compute everything computable …
  - (weaker models exist (circuits, DFA) but they don’t compute everything)
  - In fact … there exists one TM that computes everything computable!
  - … but some problems not computable ☹
- Can be used to measure complexity …
  - P = poly time = efficient
  - EXP $\neq$ P inefficient
  - NP $\neq$ P? desired to be efficient, believed inefficient.
- STCT Challengers: Randomness and Quantum
  - BPP (can do stuff not known to be in P), status also unknown ….

1. Quantum computing

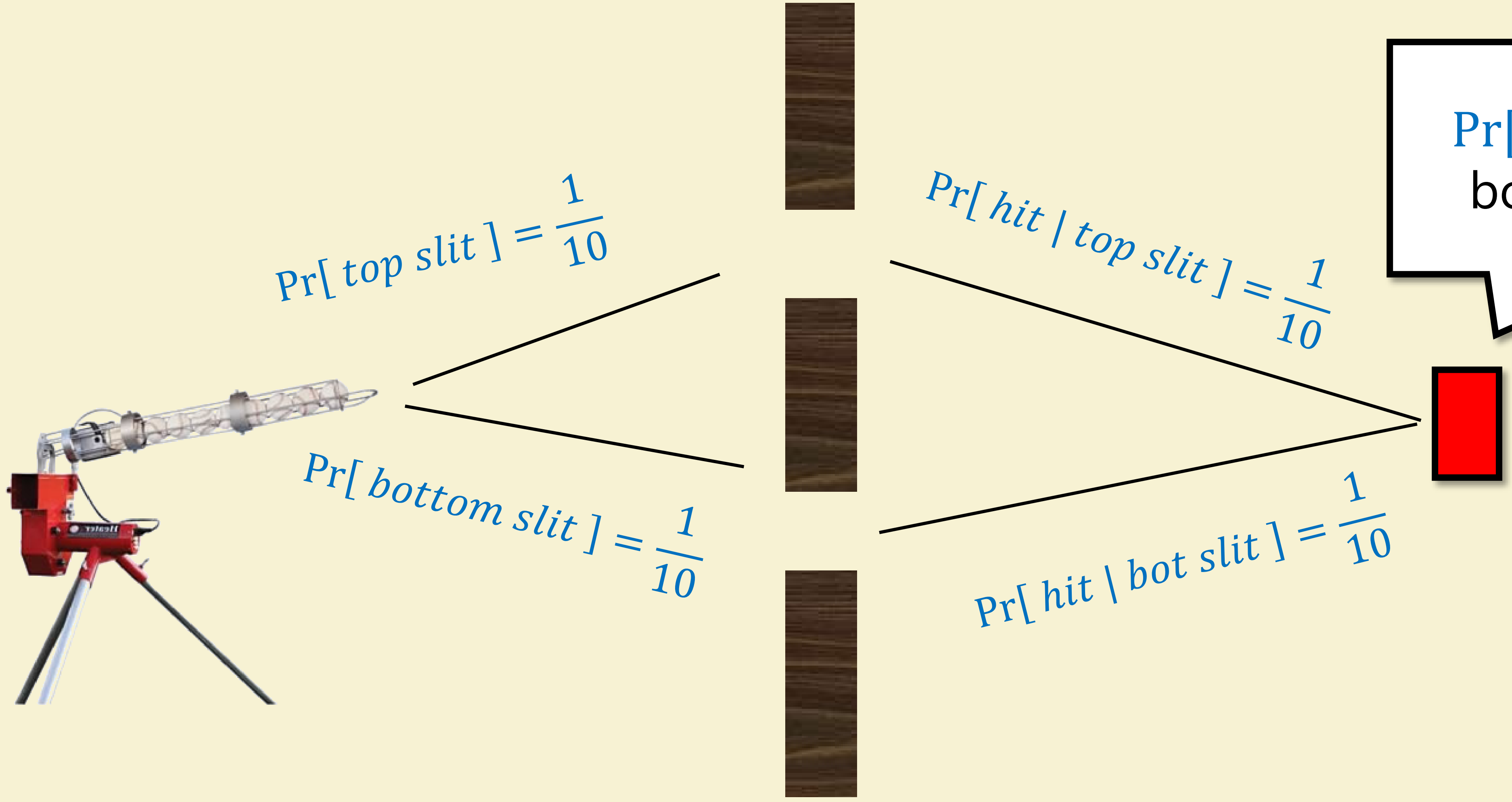
Physics 500BC-1920's: Clockwork universe

- A physical theory has basic objects ("particles") and forces between them.
- Given state of all particles at time t, can compute state at time $t+1$
- If universe has N particles, we can represent state with $O(N)$ numbers and compute one time-step in $O(N)$ or $O(N^2)$ time.

Examples: Newtonian mechanics, Maxwell's equations, Special and general relativity (& TM Configurations!)

Quantum Mechanics is not a "clockwork" theory!

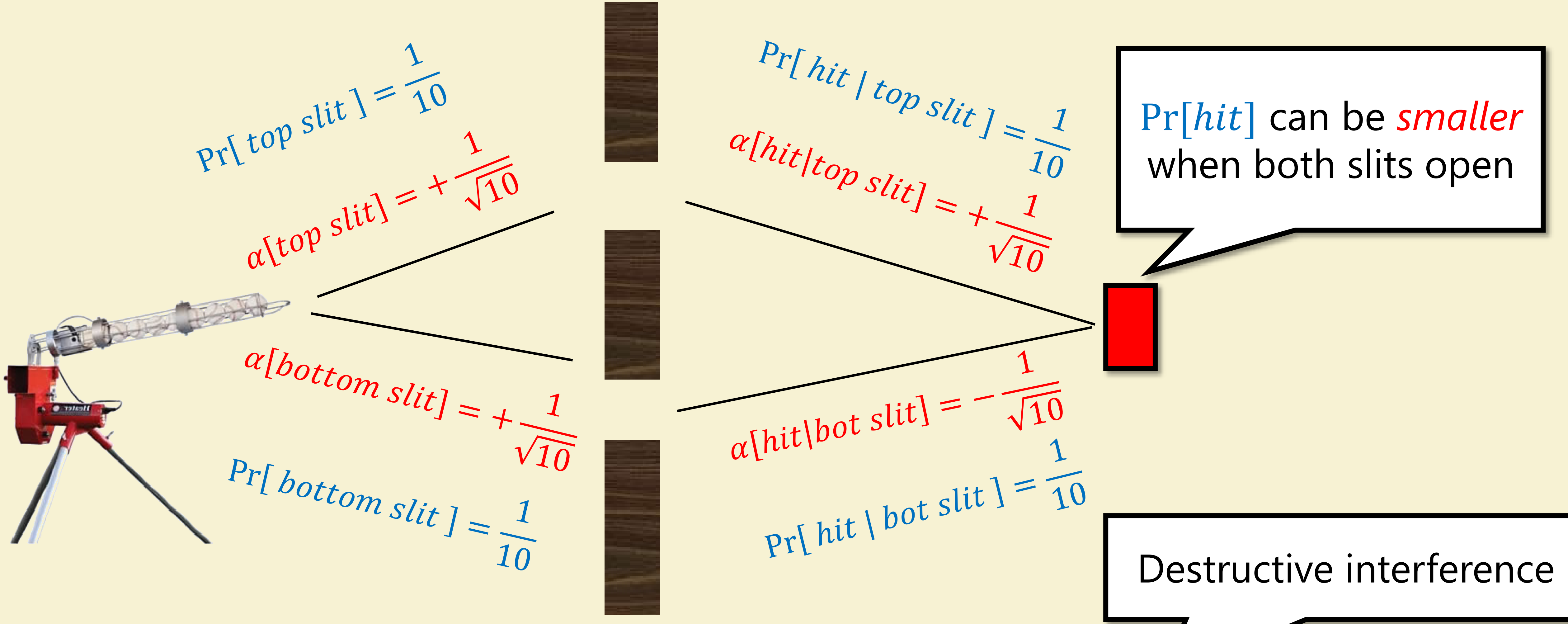
Double slit experiment: classical view



$\Pr[hit]$ only grows if both slits are open

$$\Pr[\, hit] = \Pr[hit|top] \cdot \Pr[top] + \Pr[hit|bot] \cdot \Pr[bot] = \frac{1}{10} \cdot \frac{1}{10} + \frac{1}{10} \cdot \frac{1}{10} = \frac{2}{100}$$

Double slit experiment: quantum view*



$$Pr[hit] = (\alpha[hit|top]\cdot\alpha[top] + \alpha[hit|bot]\cdot\alpha[bot])^2 = \left(\frac{1}{\sqrt{10}}\cdot\frac{1}{\sqrt{10}} - \frac{1}{\sqrt{10}}\cdot\frac{1}{\sqrt{10}}\right)^2 = 0$$

Quantum weirdness II

Measuring if an event happens collapses the amplitude – the event happens with probability α^2 and doesn’t happen with probability $1 - \alpha^2$

An event can depend on particles that are far from each other – measurement can create “spooky correlations at a distance”.

Implications to computing

- Probabilistic computing:
  - Let $f: \{0,1\}^n \to \{0,1\}^m$ be polytime computable
  - Can put computer in the configuration $2^{-n} \cdot \sum_{x \in \{0,1\}^n} f(x)$
  - Allows us to compute "average(f)", "variance(f)" etc.
- Quantum computing:
  - Let $f: \{0,1\}^n \to \{0,1\}^m$ be polytime computable
  - Can put computer in the configuration $2^{-\frac{n}{2}} \cdot \sum_{x \in \{0,1\}^n} (-1)^{x_0} f(x)$
  - Allows us to compute ?

Some quantum computing history

- 1981: Feynman talks about difficulty of simulating quantum physics with classical computers, speculates maybe a different computer would work.
- 1985: David Deutsch starts studying quantum computers in their own right.
- 1993: Bernstein and Vazirani give formal definitions, first formal evidence of exponential speedup.
- Spring 1994: Simons gives "period finding" algorithm for functions $f: \mathbb{Z}_2^n \to \mathbb{Z}_2^n$.
- Fall 1994: Shor gives "period finding" algorithm for functions $f: \mathbb{Z}_N \to \mathbb{Z}_N$ and show it implies a polynomial-time factoring algorithm. Field explodes.

More details:

- Model: Quantum Turing Machine or <u>(uniform) Quantum circuits</u>
  - Uniform circuit = constructed in time poly in its size.
- Complexity Measure: Quantum time/Quantum size (equal up to poly factors, due to uniformity)
- Complexity Class: BQP – Boolean functions computable in Poly time.

Thms: $P \subseteq BQP$, $BPP \subseteq BQP$, $BQP \subseteq EXP$

Moral of the story

- Importance of STCT!
  - Axioms of quantum physics being tested by STCT!
  - Tools used to establish tests: from CS 121/221/321….

2. Cryptography

Cryptography

- Very subtle – long history of people getting it wrong
- Can’t be taught in one class, not even one term
- Our focus is connection between cryptography, computational complexity, and randomness.

History of Crypto: 3000BC-1976

Design crypto system

Believe impossible to break

Use for life or death applications

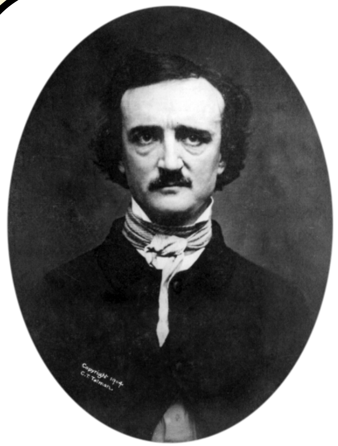
System is broken

Die

Fix crypto system

History of Crypto: 3000BC-1976

Design crypto system



"Human ingenuity cannot concoct a cipher which human ingenuity cannot resolve."

Edgar Allan Poe, 1841

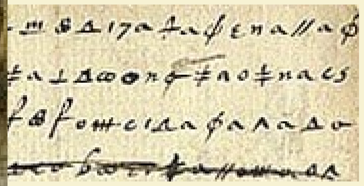
System is broken

Example 1: Mary's cipher

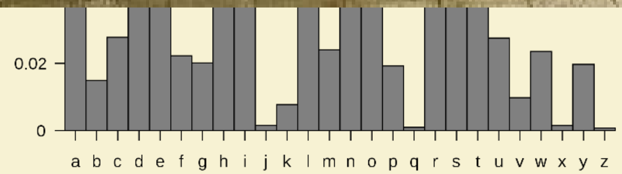
Mary qu

cousin q

Commun



Sir Franc



Example 2: Enigma

A typewriter that based on wires and rotor setting would emit different letter for every keypress.

current state

letter typed

new state

letter output

About 10^{113} possibilities to set the wirings and rotors.

Lightspeed supercomputer will take $\gg 10^{17}$ years to check them all

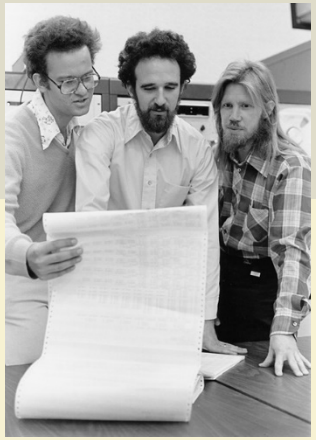
(universe is only 10^{10} years old)

Believed impossible to break by Germans.

Broken (following Polish advances) via heroic efforts by British at Bletchley park

- Cut German U-Boat success in sinking ships by ~90%
- Sank about 60% of German U-Boats in Mediterranean
- Crucial to success of Normandy D-day landing.

Modern Cryptography (1976-)



"We stand today on the brink of a revolution in cryptography"

Whit Diffie and Martin Hellman, 1976

	Data	**Cycles**	**Person-years**	**Result**
Mary's cipher	10^4 *bytes*	*N/A*	1	✗ *Broken*
Enigma	10^7 *bytes*	10^{13}	10^5	✗ *Broken*
1976				
Diffie-Hellman/RSA	10^{22} *bytes*	10^{25}	10^8	✓ *Unbroken!*

DH/RSA are simpler than Enigma, and allow public encryption key

Security through obscurity ➡ *Security through simplicity*

Modern cryptography

- Key insight: $NP \neq P$ on steroids
- There exist "one-way" functions $f: \{0,1\}^n \rightarrow \{0,1\}^n$ such that:
  - f is easy to compute (in poly(n) time)
  - f is hard to invert (Given $f(x)$, hard to find any x' such that $f(x') = f(x)$)
  - (Exercise: Prove that if NP=P, then such functions don't exist!)
- 3 Phases of modern crypto:
  - Phase 1: Diffie/Hellman, Rivest/Shamir/Adleman: Realized above, and used sheer ingenuity to build some crypto primitives (encryption, signature, key distribution)
  - Phase 2: Blum/Goldwasser/Micali/Yao: Used principles of CS (!!!reductions!!!) to "automate" development of cryptography. Start with o.w.f., + build almost everything else from them! Proven secure unless owf breaks.
  - Phase 3: … Barak/Gentry/Sahai/Waters … : Ingenuity+Principles: Homomorphic Encryption, Obfuscation

Computational Secrecy

There are not in nature two real, absolute beings, indiscernible from each other", Gottfried Wilhelm Leibniz

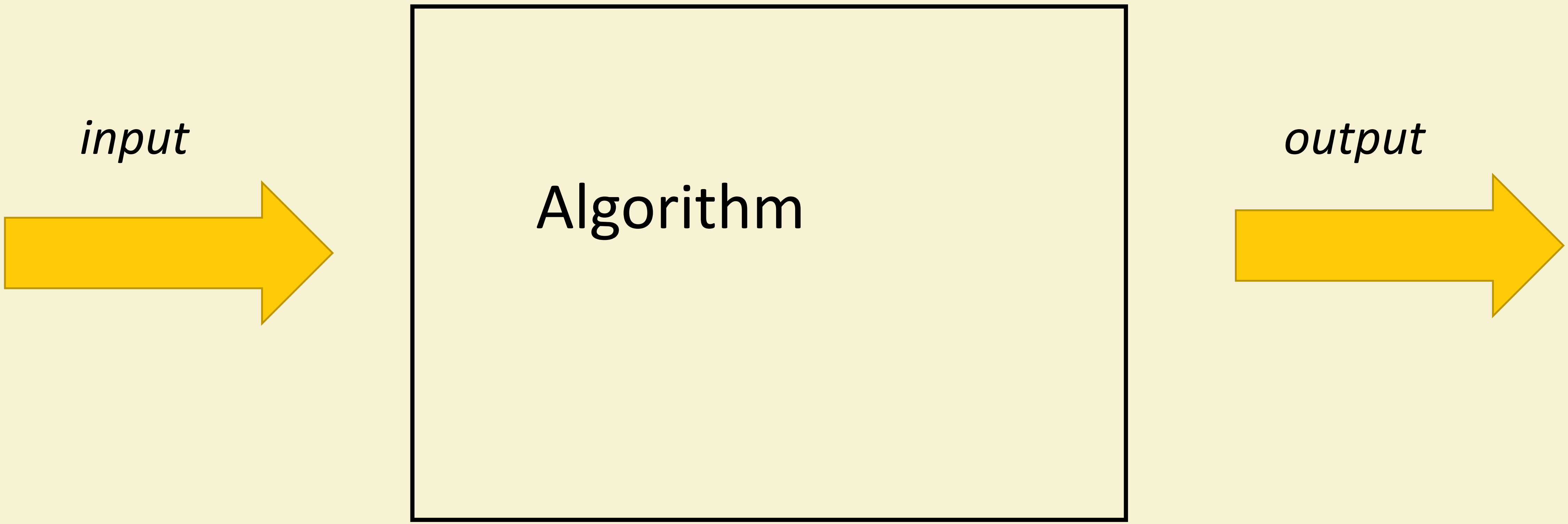
"Identity of Indiscernibles Principle"

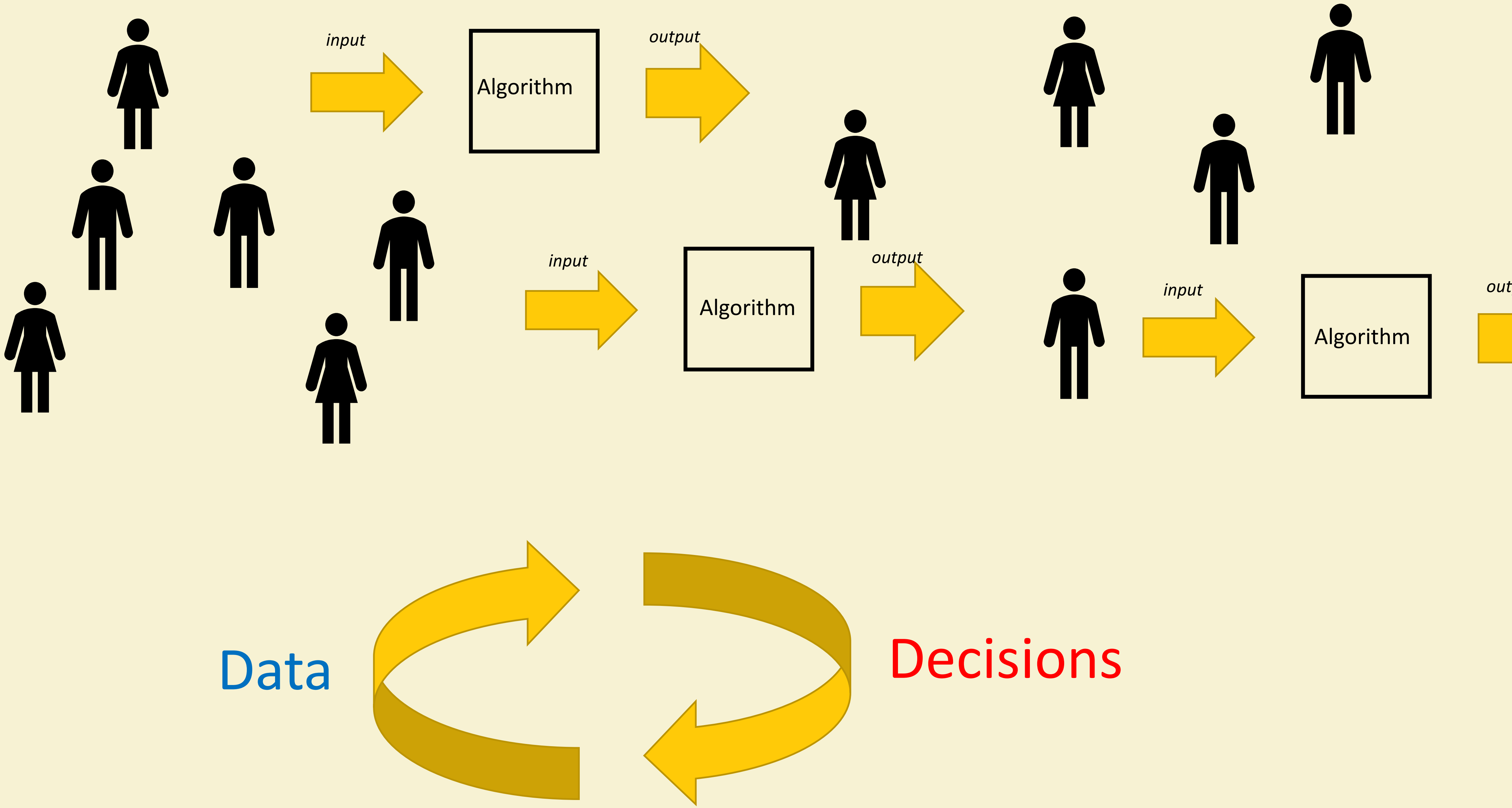
a.k.a "If two distributions cannot be distinguished by polynomial-time algorithms, they may as well be the same"

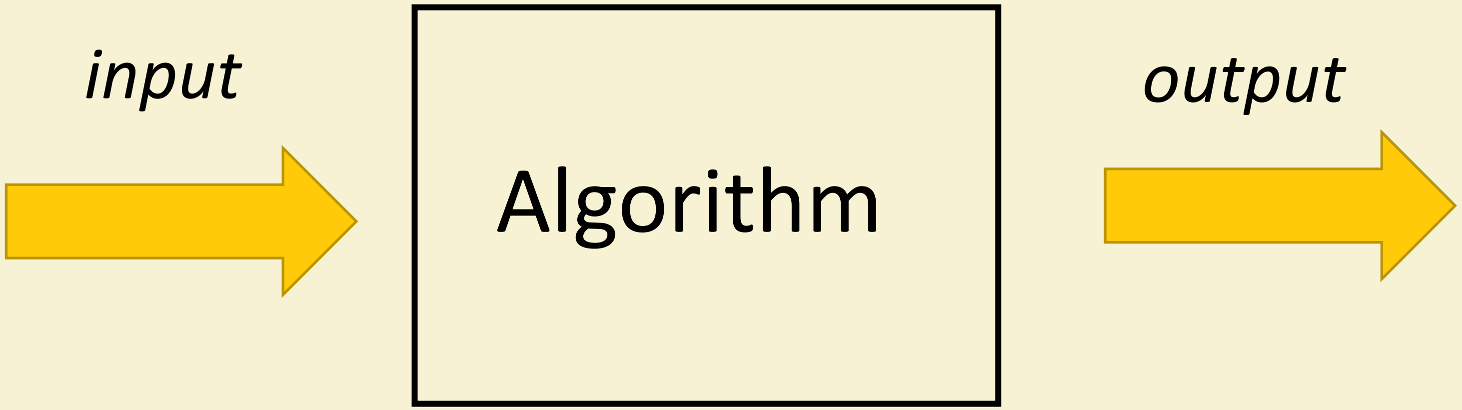
Cryptography vs. security

What we *didn't* see

(and where to see it.)


input
Algorithm
output


input
Algorithm
output
input
Algorithm
output
input
Algorithm
Data
Decisions



Who supplies the input? And what do we do with the output?

Incentives, mechanism design (CS 13x, 23x)

Privacy , *Fairness* (CS126, CS208)

Cryptography/security(CS 127/227, CS 263, MIT 6.857, MIT 6.875)

Average case complexity, learning, generalization (CS 181, 183, 228)

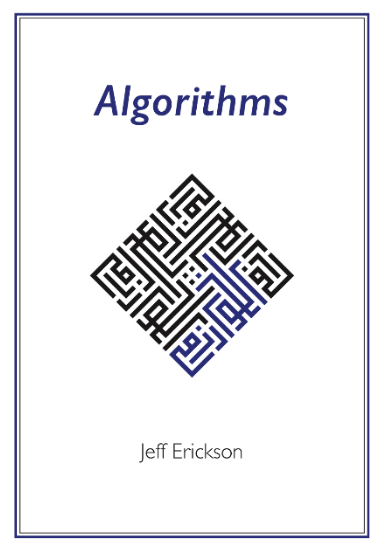
Surprising algorithms and data structures

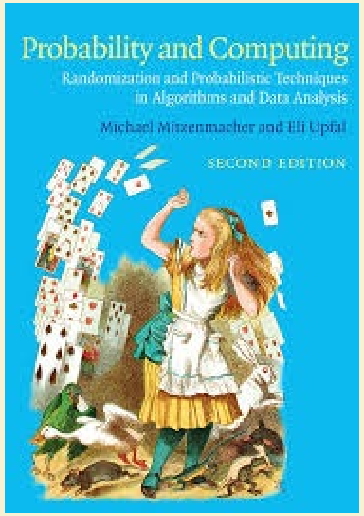
http://algorithms.wtf/

Multiply n bit numbers in $\ll n^2$ time.

Multiply $n \times n$ matrices in $\ll n^3$ time.

Solve linear programming in $\mathrm{poly}(n)$ time.





http://www.designofapproxalgs.com/

Answer query "$i \in S$?" in $O(1)$ time. *(dictionaries, hash tables)*

Answer query "$dist(u, v) < k$?" in $\ll n$ time. *(distance oracles, nearest neighbors)*

CS 124, CS 222/223, MIT 6.854

More on computational complexity

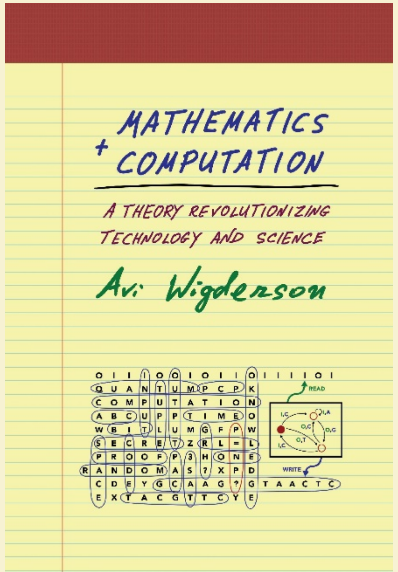
- Hardness of approximation and probabilistically checkable proofs.
- Lower bounds for concrete computational models.
  (For general Boolean circuits, can't rule out $6n$ gate circuit for 3SAT!)
- Derandomization from weaker assumptions.

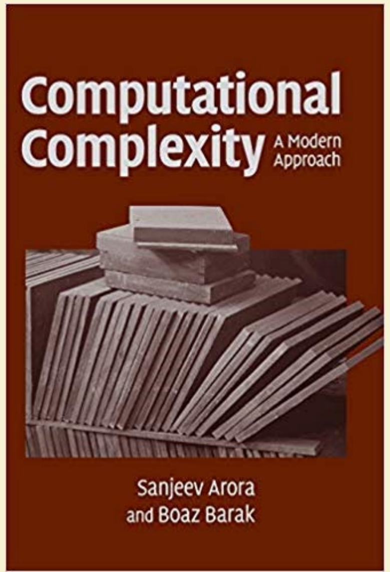
CS 221 , MIT 6.841

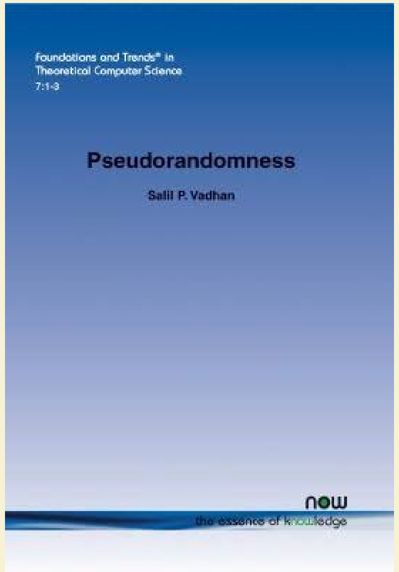
https://www.math.ias.edu/avi/book

https://theory.cs.princeton.edu/complexity/

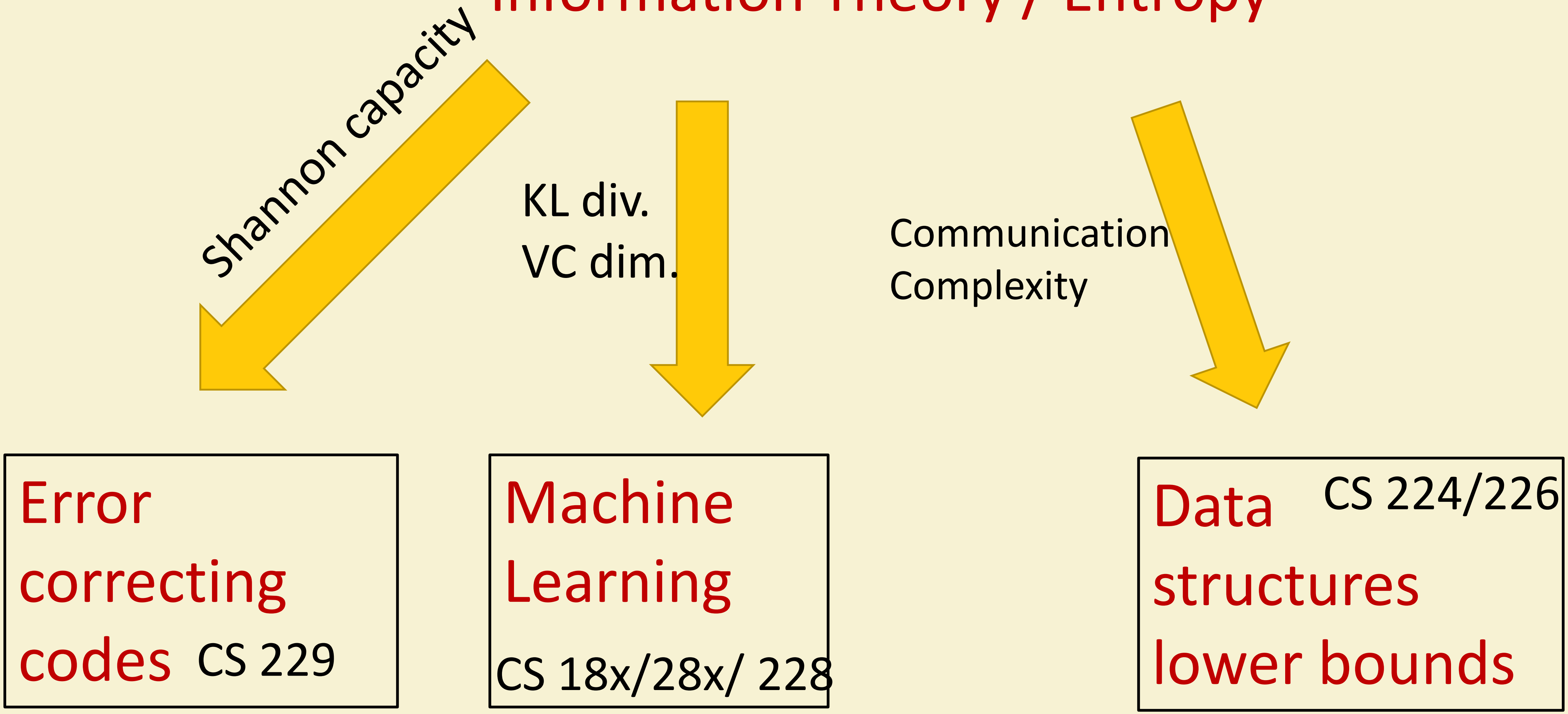
https://people.seas.harvard.edu/~salil/pseudorandomness/

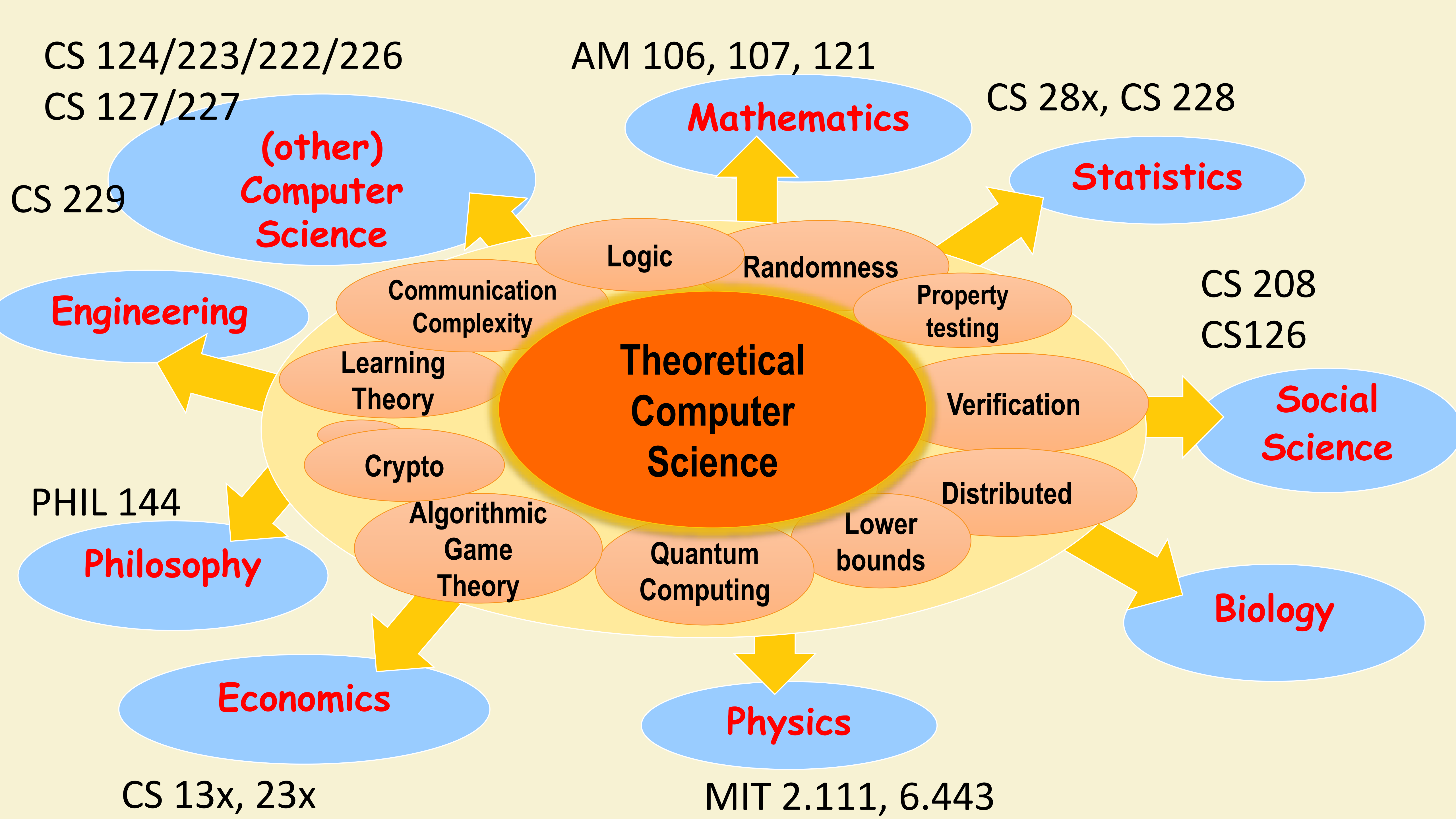




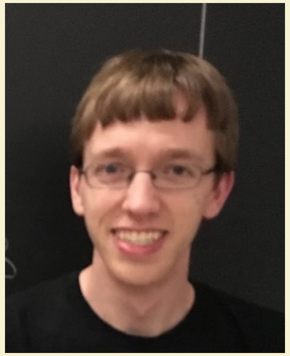


Information Theory / Entropy




CS 124/223/222/226
CS 127/227
AM 106, 107, 121
CS 28x, CS 228
(other) Computer Science
Mathematics
Statistics
CS 229
Logic
Randomness
Communication Complexity
Property testing
CS 208
CS126
Engineering
Learning Theory
Theoretical Computer Science
Verification
Social Science
Crypto
Distributed
PHIL 144
Algorithmic Game Theory
Lower bounds
Quantum Computing
Philosophy
Biology
Economics
Physics
CS 13x, 23x
MIT 2.111, 6.443

Thank You to ...


Adam Hesterberg

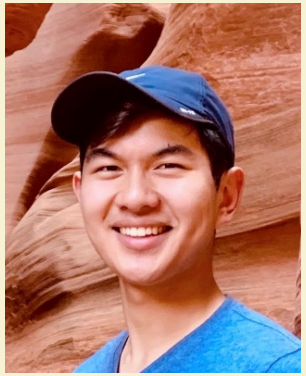

William Burke

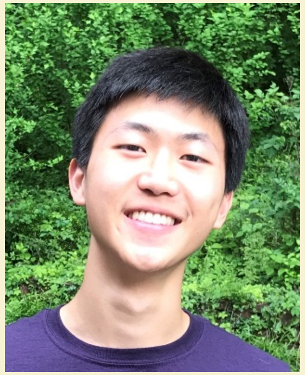

Amy Wang


Serena Davis


Nari Johnson


Joanna Boyland

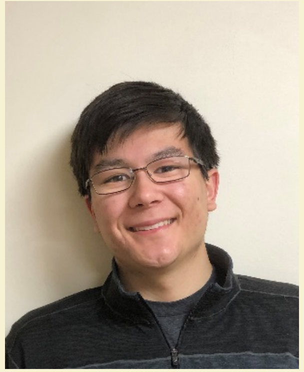

Kevin Chen


Max Guo


Diego Gutierrez


Eric Lin


Ife Omidiran


James Roney


Noah Singer


Zuzanna Skoczylas


Sahana Srinivasan


Richard Xu