Announcements:


- Final Exam (any 3hrs between [1:21pm 12/10 – 1:21pm 12/12])

  Prime time: 1:21 - 5:21 on 12/10
  Next: 8:00am - 12:00 noon on 12/11

- Homework 6 due today

- Piazza posts on review material for final
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 ≠ P inefficient
  - NP ≠ 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

\[
\Pr[\text{hit}] = \Pr[\text{hit}|\text{top}] \cdot \Pr[\text{top}] + \Pr[\text{hit}|\text{bot}] \cdot \Pr[\text{bot}]
\]

\[
= \frac{1}{10} \cdot \frac{1}{10} + \left(1 - \frac{1}{10}\right) \cdot \frac{1}{10} = \frac{2}{100} = 0.02
\]

\[
\Pr[\text{hit}] \text{ only grows if both slits are open}
\]

\[
10^5
\]

\[
\text{one slit open: } 10^4
\]

\[
0 \text{ open slits}
\]

\[
0 \text{ by two slits open}
\]
Double slit experiment: quantum view*

\[ P[hit] = (\alpha[hit|top] \cdot \alpha[top] + \alpha[hit|bot] \cdot \alpha[bot])^2 \]

\[ = \left( \frac{1}{\sqrt{10}} \cdot \frac{1}{\sqrt{10}} - \frac{1}{\sqrt{10}} \cdot \frac{1}{\sqrt{10}} \right)^2 = 0 \]

\[ P[hit|top] = \frac{1}{10} \]

\[ P[hit|bot] = \frac{1}{10} \]

Pr[hit] can be smaller when both slits open

Destructive interference
Quantum weirdness II

Measuring if an event happens **collapses** the amplitude – the event happens with probability $\alpha^2$ 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 \( f: \{0,1\}^n \rightarrow \{0,1\}^m \) 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 \( f: \{0,1\}^n \rightarrow \{0,1\}^m \) 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 $f: \mathbb{Z}_2^n \rightarrow \mathbb{Z}_2^n$.

• **Fall 1994**: Shor gives “period finding” algorithm for functions $f: \mathbb{Z}_N \rightarrow \mathbb{Z}_N$ and show it implies a polynomial-time factoring algorithm. Field explodes. Given integer $A$, can factor it in time $\text{poly}(\log A)$.
More details:

- Model: Quantum Turing Machine or (uniform) Quantum circuits
  - 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....

Chapter 23 in text!
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
Fix crypto system
Die
System is broken
Believe impossible to break
Use for life or death applications
History of Crypto: 3000BC-1976

“Human ingenuity cannot concoct a cipher which human ingenuity cannot resolve.”

Edgar Allan Poe, 1841
Example 1: Mary’s cipher

Mary, queen of Scots, planned to assassinate her cousin queen Elisabeth in 1587.

Communicated the plan using a substitution cipher.

Sir Francis Walsingham broke it using frequency analysis.
Example 2: Enigma

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

<table>
<thead>
<tr>
<th>current state</th>
<th>new state</th>
</tr>
</thead>
<tbody>
<tr>
<td>letter typed</td>
<td>letter output</td>
</tr>
</tbody>
</table>

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 $\sim 90\%$
- 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

<table>
<thead>
<tr>
<th></th>
<th>Data</th>
<th>Cycles</th>
<th>Person-years</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mary’s cipher</td>
<td>$10^4$ bytes</td>
<td>N/A</td>
<td>1</td>
<td>Broken</td>
</tr>
<tr>
<td>Enigma</td>
<td>$10^7$ bytes</td>
<td>$10^{13}$</td>
<td>$10^5$</td>
<td>Broken</td>
</tr>
<tr>
<td><strong>1976</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Diffie-Hellman/RSA</td>
<td>$10^{22}$ bytes</td>
<td>$10^{25}$</td>
<td>$10^8$</td>
<td>Unbroken!</td>
</tr>
</tbody>
</table>

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 \( \text{poly}(n) \) 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
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.)
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

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

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

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

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

Answer query "$\text{dist}(u, v) < k$?" 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

- Shannon capacity
- KL div.
- VC dim.
- Communication Complexity

Error correcting codes  CS 229
Machine Learning  CS 18x/28x/ 228
Data structures lower bounds  CS 224/226
Thank You to ...
