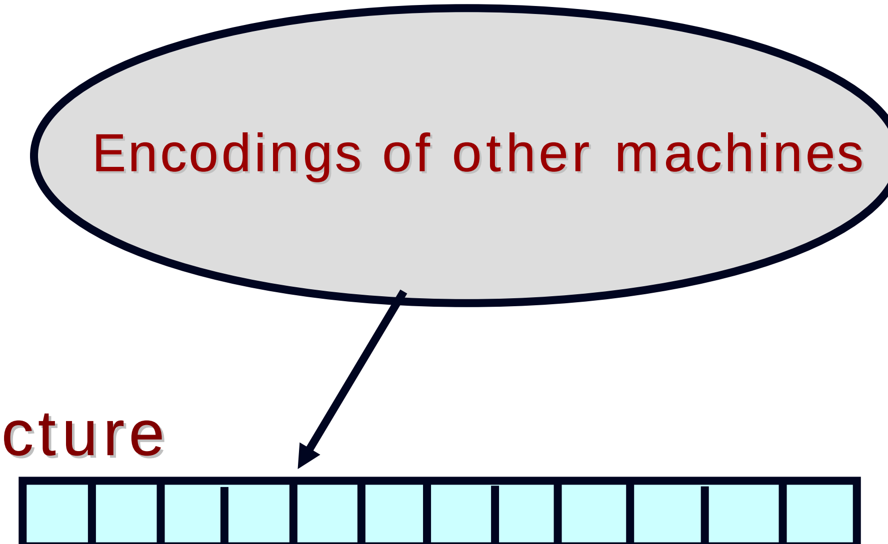
Communication & Computation

A need for a new unifying theory

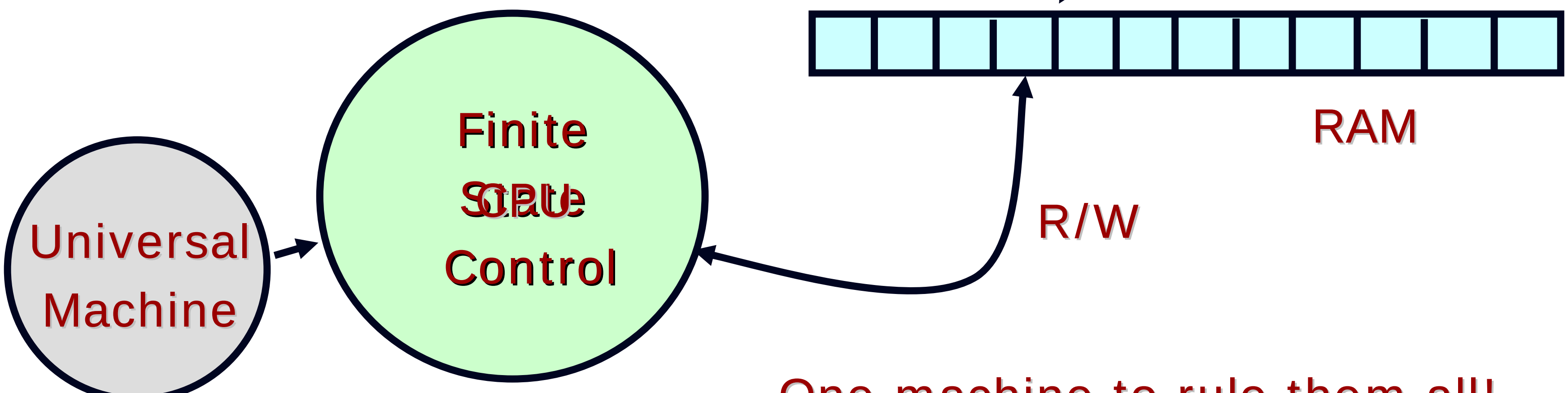
Madhu Sudan

MIT CSAIL

Theory of Computing



- Turing architecture
  → von Neumann architecture



One machine to rule them all!

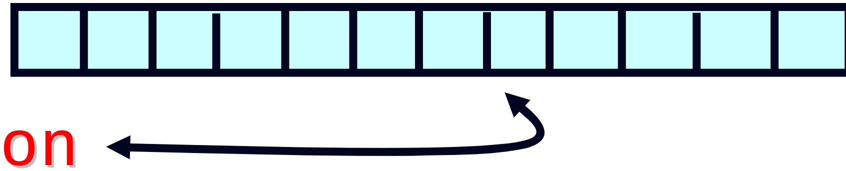
Theory of Communication

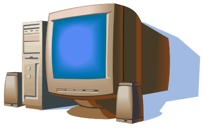
- Shannon's architecture for communication over noisy channel

- Yields reliable communication
  - (and storage (= communication across time)).

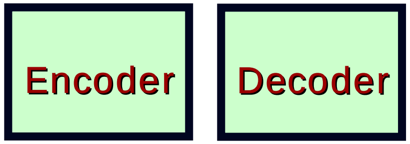
Turing ⟷ Shannon

- Turing
  - Assumes perfect storage
  - and perfect communication
  - To get computation





- Shannon
  - Assumes computation
  - To get reliable storage + communication



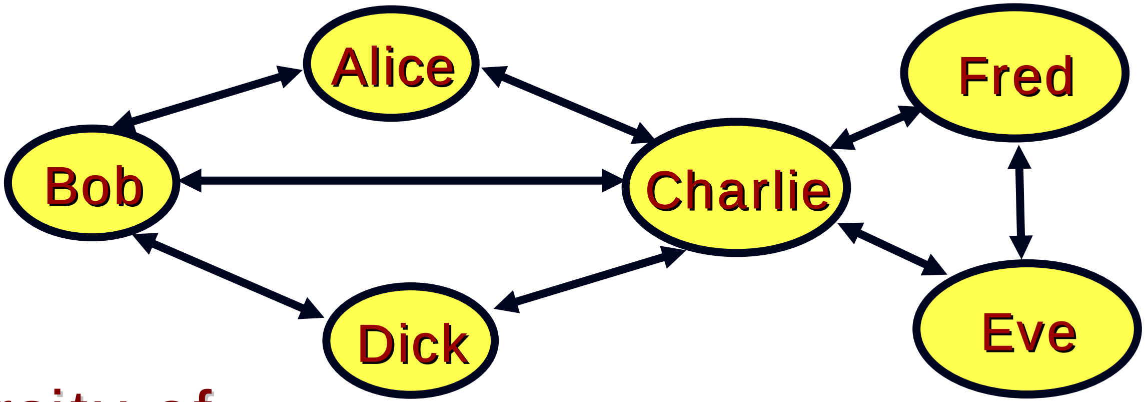
- Chicken vs. Egg?
  - Fortunately both realized!

1940s – 2000:

- Theories developed mostly independently.
  - Shannon abstraction (separating information theoretic properties of encoder/decoder from computational issues) – mostly successful.
  - Turing assumption (reliable storage/communication) – mostly realistic.

Modern Theory (of Comm. & Comp.)

- Network (society?) of communicating computers



- Diversity of
  - Capability
  - Protocols
  - Objectives
  - Concerns

Modern Challenges (to communication)

- Nature of communication is more complex.
  - Channels are more complex (composed of many smaller, potentially *clever* sub-channels)
    - Alters nature of errors

  - Scale of information being stored/communicated is much larger.
    - Does scaling enhance reliability or decrease it?

  - The Meaning of Information
    - Entities constantly evolving. Can they preserve meaning of information?

Part I: Modeling errors

Shannon (1948) vs. Hamming (1950)

- q-ary channel:
  - Input: n element string Y over Σ= {1,..., q}
  - Output: n element string Ŷ over Σ= {1,..., q}

- Shannon: Errors = Random
  - $\hat{Y}_i = Y_i$ w.p. 1 – p, uniform in Σ – $\{Y_i\}$ w.p. p.

  $p < 1 - \frac{1}{q} \Rightarrow$ Channel can be used reliably

  $q \to \infty \Rightarrow p \to 1$

- Hamming: Errors = Adversarial
  - p-fraction of i's satisfy $\hat{Y}_i \neq Y_i$
  - p can never exceed ½!

Shannon (1948) vs. Hamming (1950)

- q-ary channel:
  - Input: n element string Y over $\Sigma = \{1,\ldots, q\}$
  - Output: n element string $\hat{Y}$ over $\Sigma = \{1,\ldots, q\}$

- Shannon: Errors = Random
  - $\hat{Y}_i = Y_i$ w.p. $1 - p$, uniform in $\Sigma - \{Y_i\}$ w.p. p.

  $p < 1 - \frac{1}{q} \Rightarrow$ Channel can be used reliably

  $q \to \infty \Rightarrow p \to 1$

- Hamming: Errors = Adversarial
  - p-fraction of i's satisfy $\hat{Y}_i \neq Y_i$
  - p can never exceed ½!

Which is the right model?

- 60 years of wisdom …
  - Error model can be fine-tuned …
  - Fresh combinatorics, algorithms, probabilistic models can be built …
  - … to fit Shannon Model.

Corrects More Errors!

- An alternative – List-Decoding [Elias '56]!
  - Decoder allowed to produce list $\{m_1,\ldots,m_l\}$
  - "Successful" if $\{m_1,\ldots,m_l\}$ contains m.
  - "60 years of wisdom" $\Rightarrow$ this is good enough!
  - [70s]: Corrects as many adversarial errors as random ones!

Safer Model!

Challenges in List-decoding!

- Algorithms?
  - Correcting a few errors is already challenging!
    - Can we really correct 70% errors? 99% errors?
    - When an adversary injects them?
    - Note: More errors than data!
- Till 1988 … no list-decoding algorithms.
  - [Goldreich-Levin '88] – Raised question
    - Gave non-trivial algorithm (for weak code).
    - Gave cryptographic applications.

Algorithms for List-decoding

- [S. '96], [Guruswami + S. '98]:
  - List-decoding of Reed-Solomon codes.
  - Corrected p-fraction error with linear "rate".
- ['98 – '06] Many algorithmic innovations ...
  - [Guruswami, Shokrollahi, Koetter-Vardy, Indyk]
- [Parvaresh-Vardy '05 + Guruswami-Rudra '06]
  - List-decoding of new variant of Reed-Solomon codes.
  - Correct p-fraction error with optimal "rate".

Reed-Solomon List-Decoding Problem

- Given:
  - Parameters: n,k,t
  - Points: $(x_1,y_1),\ldots,(x_n,y_n)$ in the plane

    (over finite fields, actually)

- Find:
  - All degree k polynomials that pass through t of the n points.

    i.e., p such that

    - $\deg(p) \le k$
    - $|\{i \text{ s.t. } p(x_i) = y_i\}| \ge t$

Decoding by Example + Picture [S. '96]

n=14;k=1;t=5

Algorithm Idea:

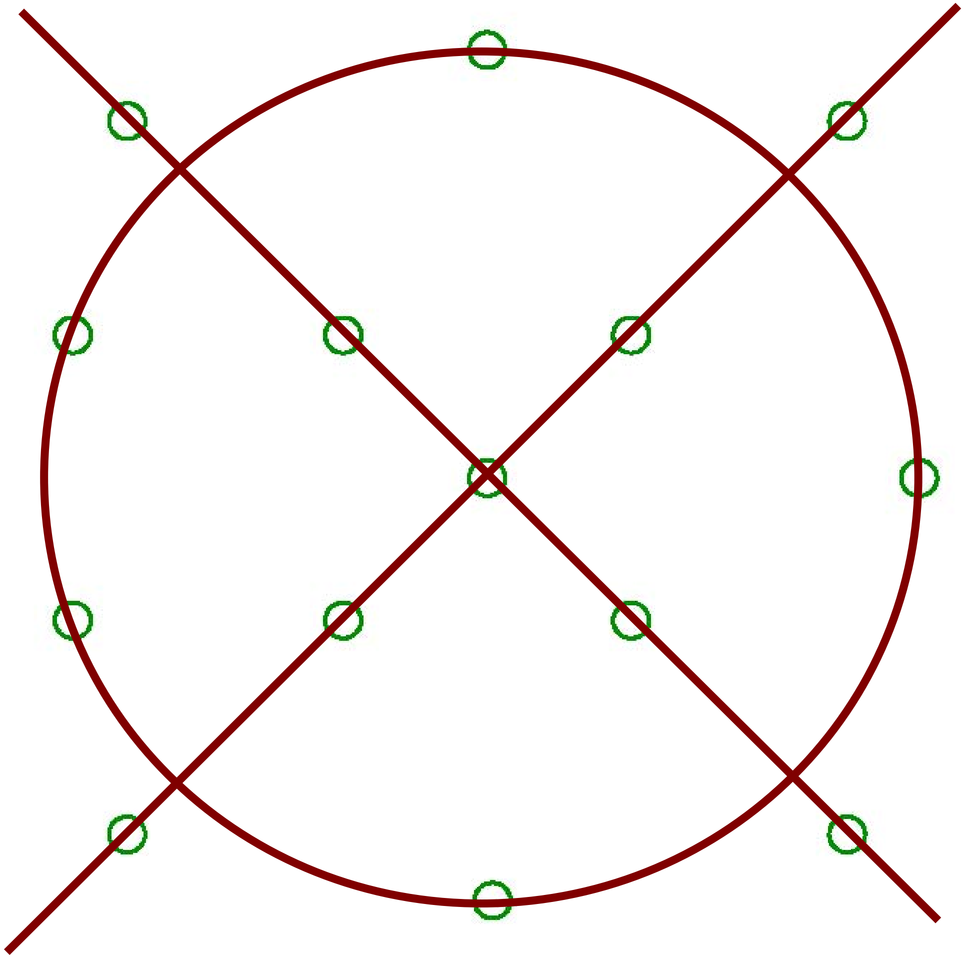
- Find algebraic explanation of *all* points.

$$x^4 - y^4 + x^2 - y^2 = 0$$

- Stare at it!

  Factor the polynomial!

$$(x^2 + y^2 - 1)(x + y)(x - y)$$



Decoding Algorithm

- Fact: There is always a degree $2\sqrt{n}$ polynomial thru n points
  - Can be found in polynomial time (solving linear system).
- [80s]: Polynomials can be factored in polynomial time [Grigoriev, Kaltofen, Lenstra]
- Leads to (simple, efficient) list-decoding correcting p fraction errors for $p \rightarrow 1$

Conclusion

- More errors (than data!) can be dealt with …
  - More computational power leads to better error-correction.

- Theoretical Challenge: List-decoding on binary channel (with optimal (Shannon) rates).
  - Important to clarify the right model.

Part II: Massive Data; Local Algorithms

Reliability vs. Size of Data

- Q: How reliably can one store data as the amount of data increases?
  - [Shannon]: Can store information at close to "optimal" rate, and prob. decoding error <u>drops</u> exponentially with <u>length</u> of data.
    - Surprising at the time?
  - Decoding time grows with length of data
    - Exponentially in Shannon
    - Subsequently polynomial, even linear.
  - Is the bad news necessary?

Sublinear time algorithmics

- Algorithms don't always need to run in linear time (!), provided …
  - They have random access to input,
  - Output is short (relative to input),
  - Answers don't have usual, exact, guarantee!

- Applies, in particular, to $\boxed{\text{Decoder}}$
  - Given CD, "test" to see if it has (too many) errors? [Locally Testable Codes]
  - Given CD, recover particular block. [Locally Decodable Codes]

Progress [1990-2008]

- Question raised in context of results in complexity and privacy
  - Probabilistically checkable proofs
  - Private Information Retrieval
- Summary:
  - Many non-trivial tradeoffs possible.
  - Locality can be reduced to n^{ϵ} at $O(1)$ penalty to rate, fairly easily.
  - Much better effects possible with more intricate constructions.
    - [Ben-Sasson+S. '05, Dinur '06]: O(1)-local testing with poly(log n) penalty in rate.
    - [Yekhanin '07, Raghavendra '07, Efremenko '08]: 3-local decoding with subexponential penalty in rate.

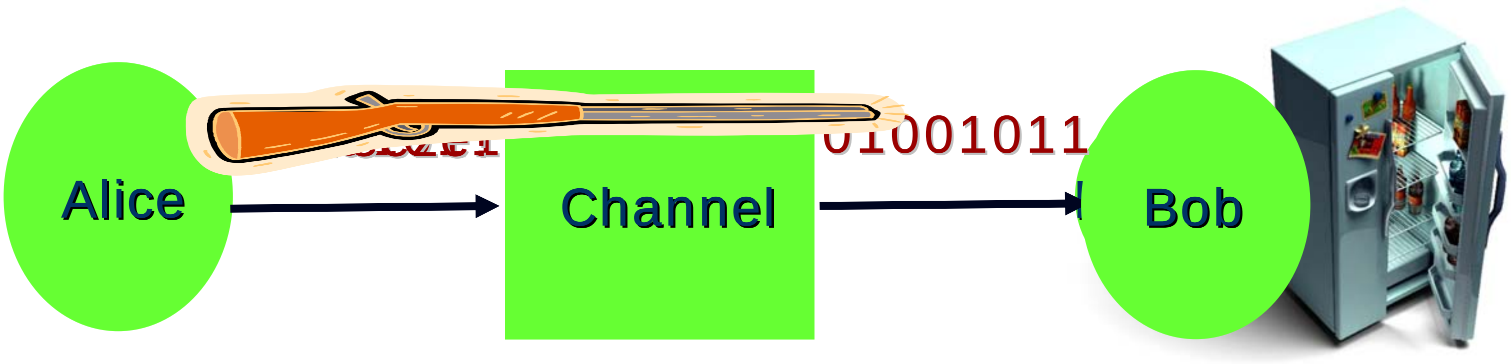
Challenges ahead

- Technical challenges
  - Linear rate testability?
  - Polynomial rate decodability?

- Bigger Challenge
  - What is the model for the future storage of information?
  - How are we going to cope with increasing drive to digital information?

Part III: The Meaning of Information

The Meaning of Bits



- Is this perfect communication?

- What if Alice is trying to send instructions?
  - In other words … an algorithm
  - Does Bob understand the correct algorithm?
  - What if Alice and Bob speak in different (programming) languages?

Motivation: Better Computing

- Networked computers use common languages:
  - Interaction between computers (getting your computer onto internet).
  - Interaction between pieces of software.
  - Interaction between software, data and devices.

- Getting two computing environments to “talk” to each other is getting problematic:
  - time consuming, unreliable, insecure.

- Can we communicate more like humans do?

Some modelling

- Say, Alice and Bob know different programming languages. Alice wishes to send an algorithm A to Bob.

- Bad News: Can't be done
  - For every Bob, there exist algorithms A and A', and Alices, Alice and Alice', such that Alice sending A is indistinguishable (to Bob) from Alice' sending A'

- Good News: Need not be done.
  - From Bob's perspective, if A and A' are indistinguishable, then they are equally useful to him.

- Question: What should be communicated? Why?

Ongoing Work [Juba & S.]

- Assertion/Assumption: Communication happens when communicators have (explicit) goals.
- Goals:
  - (Remote) Control:
    - Actuating some change in environment
      - Example
        - Printing on printer
        - Buying from Amazon
  - Intellectual:
    - Learn something from (about?) environment
      - Example
        - This lecture (what's in it for you? For me?)

Example: Computational Goal

- Bob (weak computer) communicating with Alice (strong computer) to solve hard problem.
- Alice "Helpful" if she can help some (weak) Bob' solve the problem.
- Theorem [Juba & S.]: Bob can use Alice's help to solve his problem iff problem is verifiable (for every Helpful Alice).
- "Misunderstanding" = "Mistrust"

Example Problems

- Bob wishes to ...
  - ... solve undecidable problem (virus-detection)
    - Not verifiable; so solves problems incorrectly for some Alices.
    - Hence does not learn her language.
  - ... break cryptosystem
    - Verifiable; so Bob can use her help.
    - Must be learning her language!
  - ... Sort integers
    - Verifiable; so Bob does solve her problem.
    - Trivial: Might still not be learning her language.

Generalizing

- Generic Goals
  - Typical goals: Wishful
    - Is Alice a human? or computer?
    - Does she understand me?
    - Will she listen to me (and do what I say)?

  - Achievable goals: Verifiable
    - Bob should be able to test achievement by looking at his input/output exchanges with Alice.
  - Question: Which wishful goals are verifiable?

Concluding

- More, complex, errors can be dealt with, thanks to improved computational abilities
- Need to build/study tradeoffs between global reliability and local computation.
- Meaning of information needs to be preserved!
- Need to merge computation and communication more tightly!

Thank You!