Lecture 36

CSE 331

Nov 24, 2014

HW 9 due today

Place Q1, Q2 and Q3 in separate piles

I will not accept HWs after 1:15pm

DO NOT FORGET TO WRITE DOWN YOUR SOURCES

Other HW related stuff

HW 10 has been posted (due last day of class)

Solutions to HW 9 at the END of the lecture

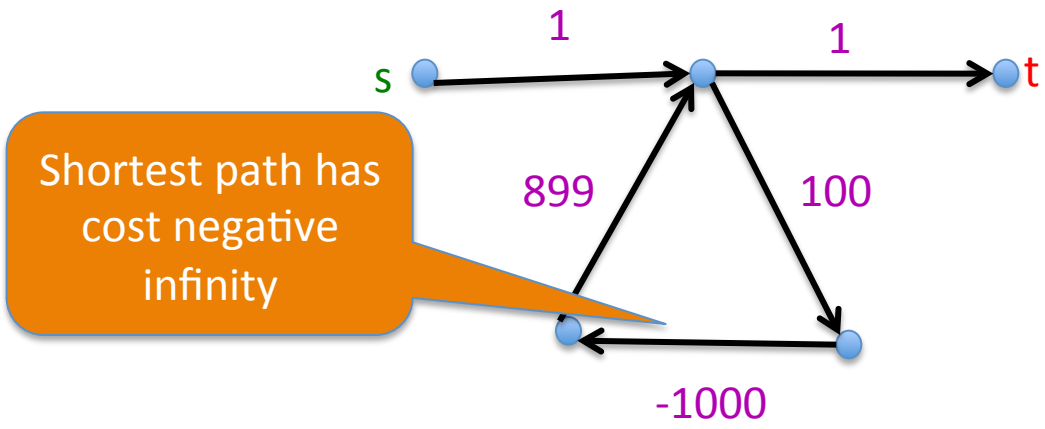
Some finals related stuff

Shortest Path Problem

Input: (Directed) Graph G=(V,E) and for every edge e has a cost c_e (can be <0)

t in V

Output: Shortest path from every s to t



Assume that G has no negative cycle

When to use Dynamic Programming

Richard Bellman

There are polynomially many sub-problems

Optimal solution can be computed from solutions to sub-problems

There is an ordering among sub-problem that allows for iterative solution

Recurrence Relation

OPT(u,i) = cost of shortest path from u to t with at most i edges

$$OPT(u,i) = \min \left\{ OPT(u, i-1), \min_{(u,w) \text{ in } E} \left\{ c_{u,w} + OPT(w, i-1) \right\} \right\}$$

Path uses ≤ i-1 edges

Best path through all neighbors

When to use Dynamic Programming

Richard Bellman

There are polynomially many sub-problems

Optimal solution can be computed from solutions to sub-problems

There is an ordering among sub-problem that allows for iterative solution

HW 9 due today

Place Q1, Q2 and Q3 in separate piles

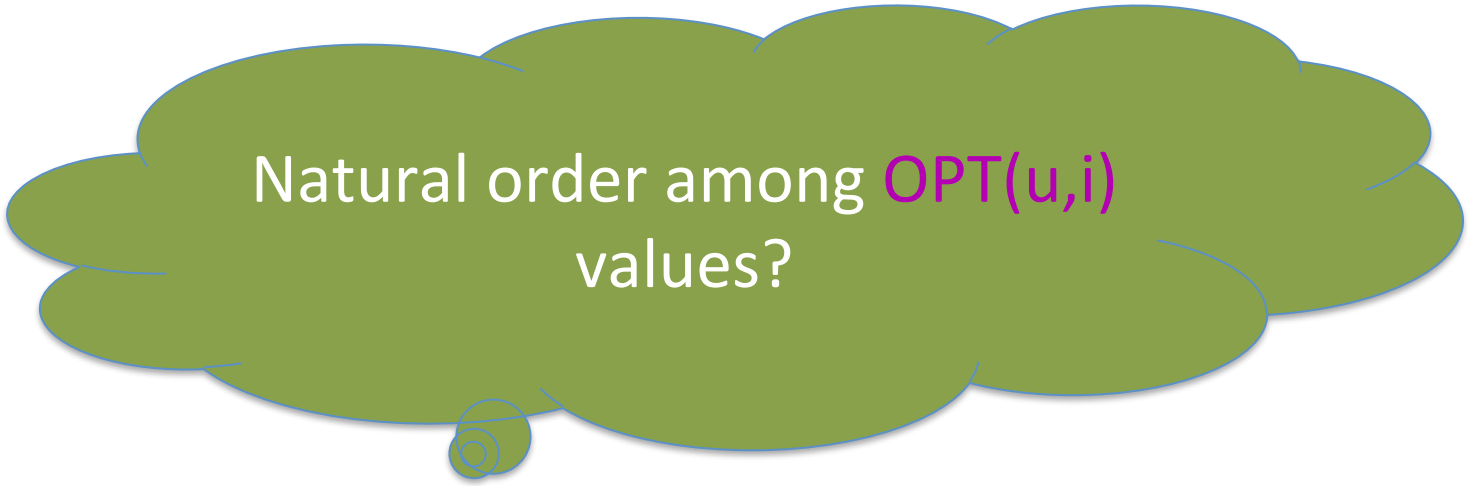
I will not accept HWs after 1:15pm

DO NOT FORGET TO WRITE DOWN YOUR SOURCES

Today's agenda

Finish Bellman-Ford algorithm

Analyze the run time



The recurrence

OPT(u,i) = shortest path from u to t with at most i edges

$$OPT(u,i) = \min \left\{ OPT(u,i-1), \min_{(u,w) \text{ in } E} \left\{ c_{u,w} + OPT(w, i-1) \right\} \right\}$$

Some consequences

OPT(u,i) = shortest path from u to t with at most i edges

$$OPT(u,i) = \min \{ OPT(u, i-1), \min_{(u,w) \text{ in } E} \{ c_{u,w} + OPT(w,i-1) \} \}$$

OPT(u,n-1) is shortest path cost between u and t

Group talk time:
How to compute the shortest path between s and t given all OPT(u,i) values

Longest path problem

Given G, does there exist a simple path of length n-1 ?

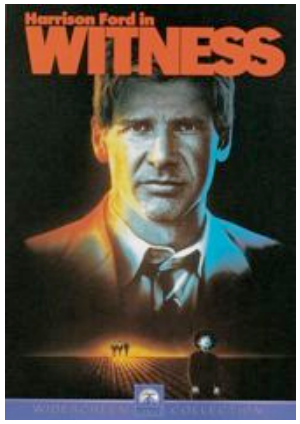
Longest vs Shortest Paths

Two sides of the "same" coin

Shortest Path problem

Can be solved by a polynomial time algorithm

Is there a longest path of length n-1?



Given a path can verify in polynomial time if the answer is yes

Poly time algo for longest path?

Clay Mathematics Institute

Dedicated to increasing and disseminating mathematical knowledge

HOME | ABOUT CMI | PROGRAMS | NEWS & EVENTS | AWARDS | SCHOLARS | PUBLICATIONS

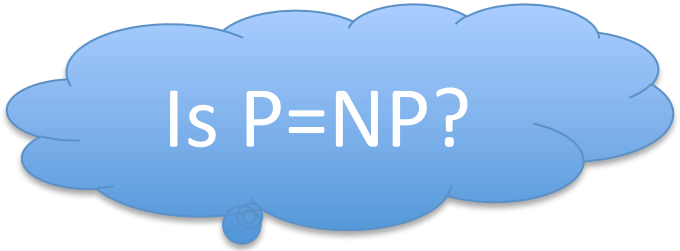
First Clay Mathematics Institute Millennium Prize Announced

Prize for Resolution of the Poincaré Conjecture Awarded to Dr. Grigoriy Perelman

- Birch and Swinnerton-Dyer Conjecture
- Hodge Conjecture
- Navier-Stokes Equations
- P vs NP
- Poincaré Conjecture

P vs NP question

P: problems that can be solved by poly time algorithms

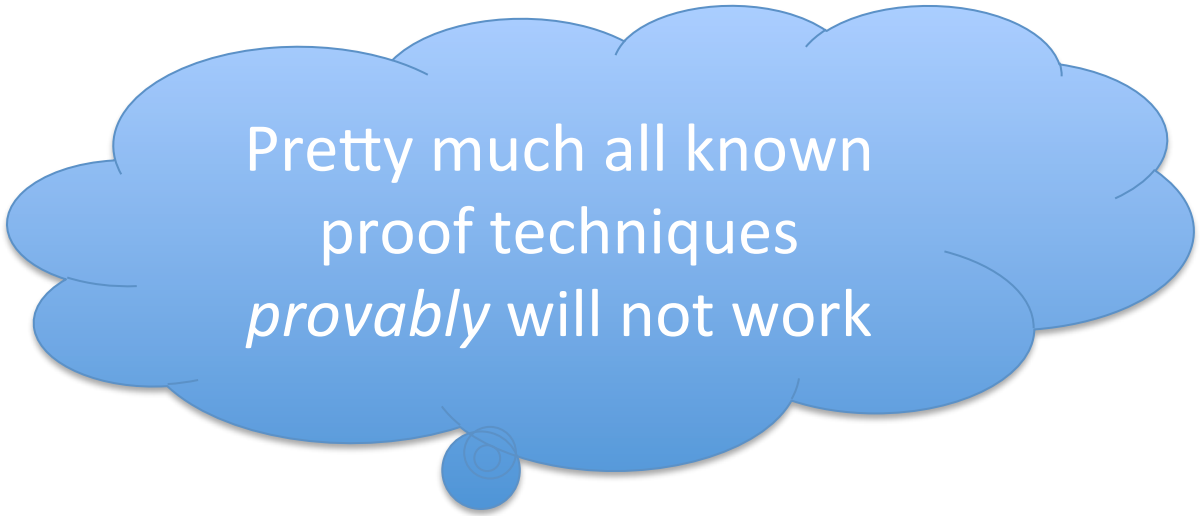


NP: problems that have polynomial time verifiable witness to optimal solution

Alternate NP definition: Guess witness and verify!

Proving P ≠ NP

Pick any one problem in NP and show it cannot be solved in poly time

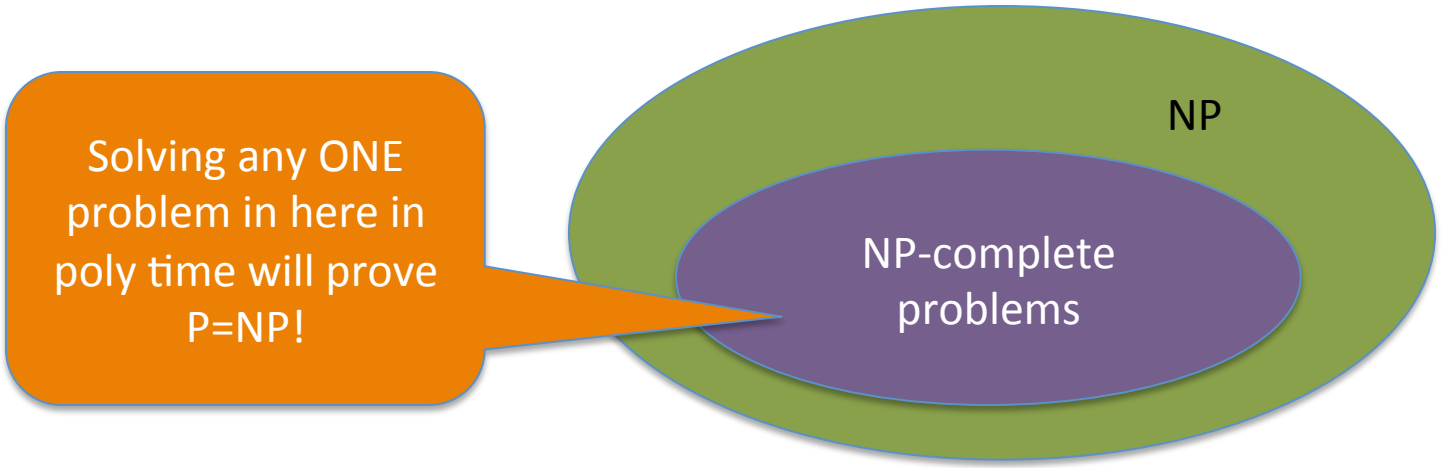


Proving P = NP

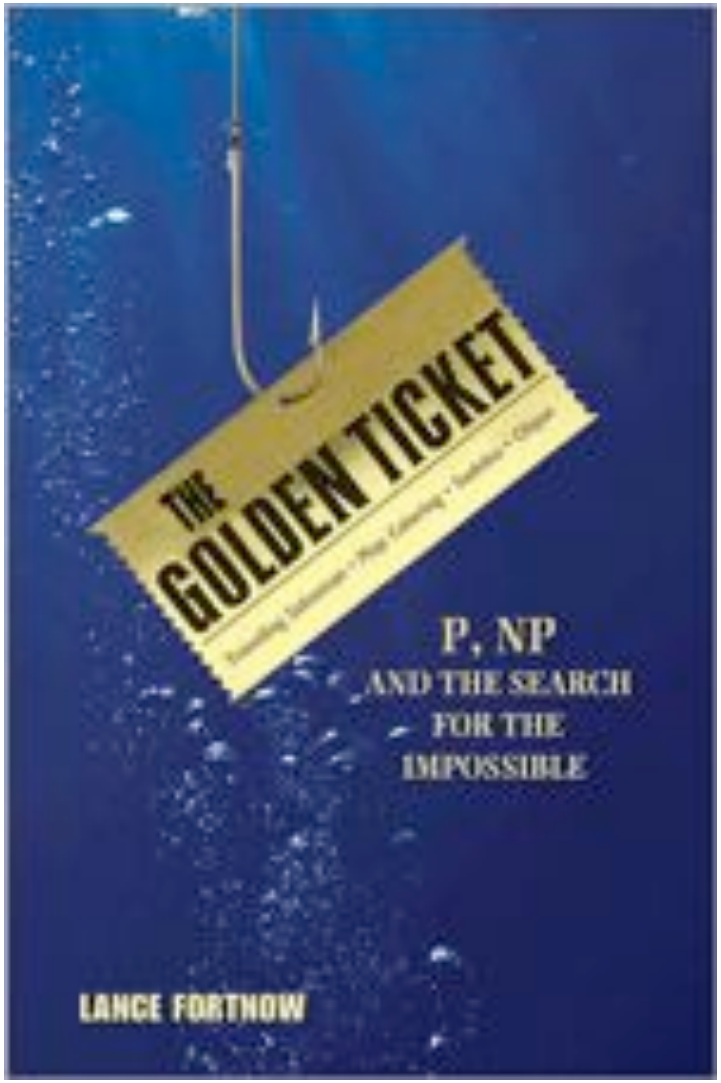
Will make cryptography collapse

Compute the encryption key!

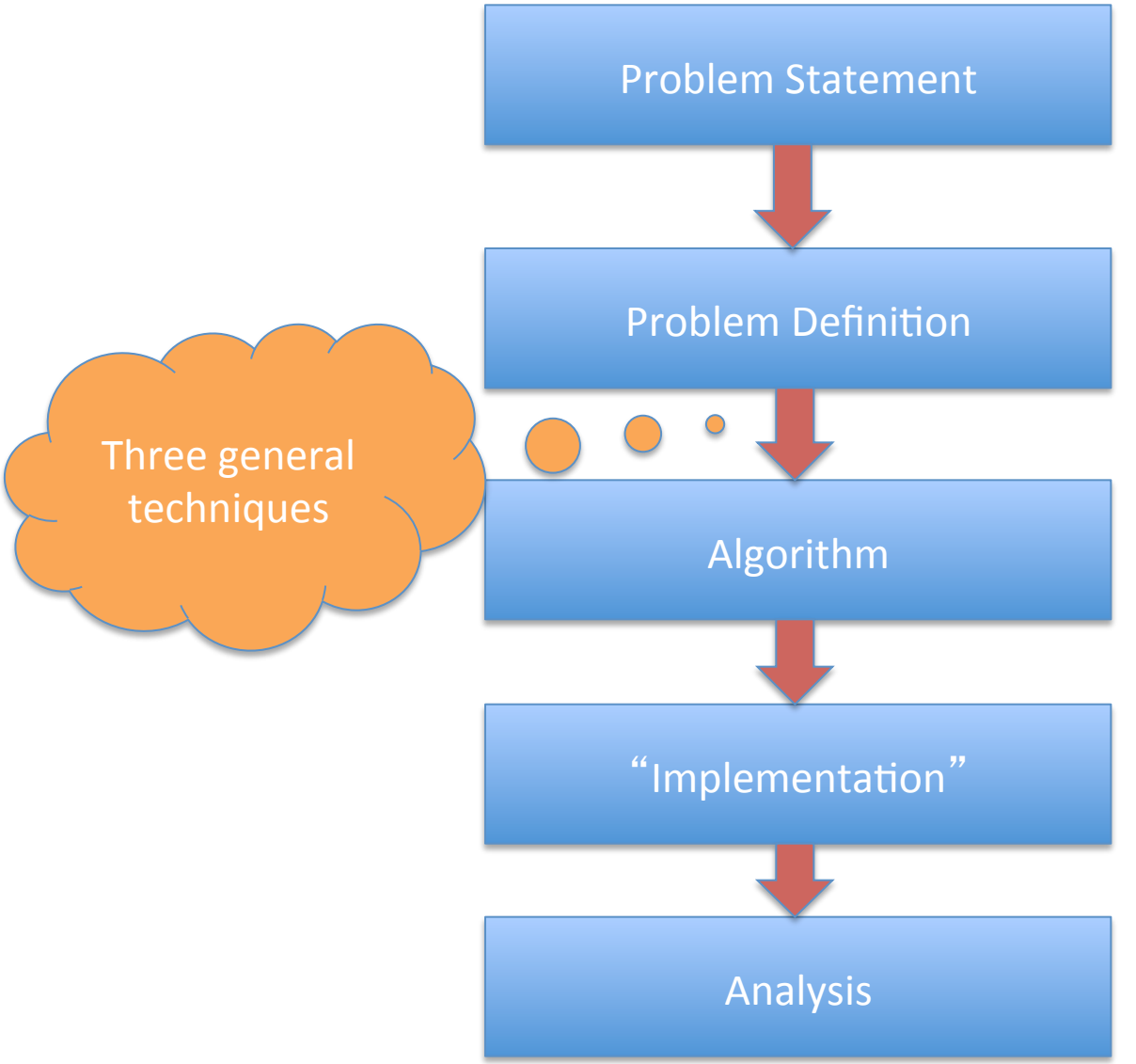
Prove that all problems in NP can be solved by polynomial time algorithms



A book on P vs. NP



High level view of CSE 331



Data Structures

Correctness+Runtime Analysis

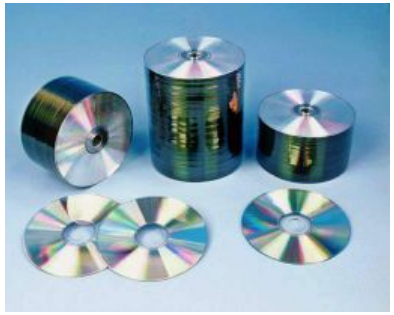
If you are curious for more

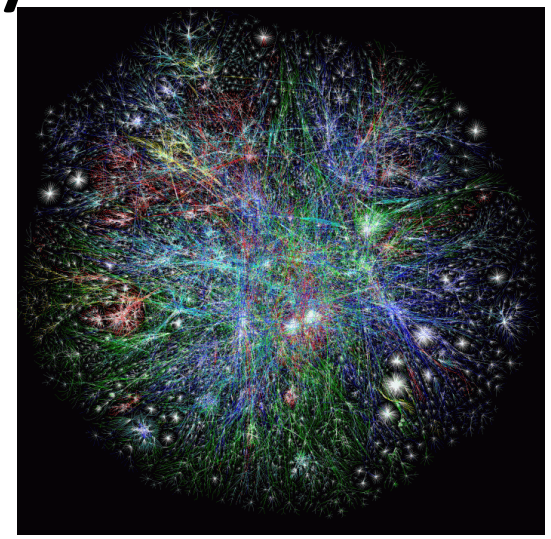
CSE 429 or 431: Algorithms

CSE 396: Theory of Computation

Now relax…

Coding Theory

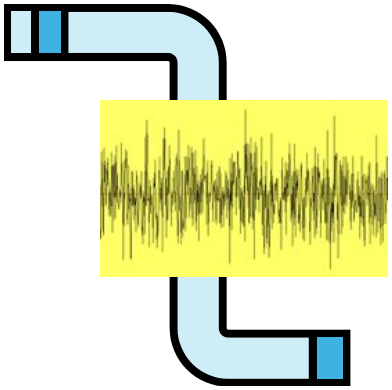




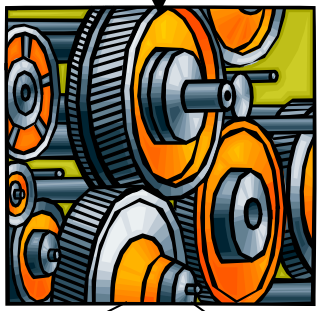
Communicating with my 5 year old

x

C(x)



y = C(x)+error



x

Give up

"Code" **C**

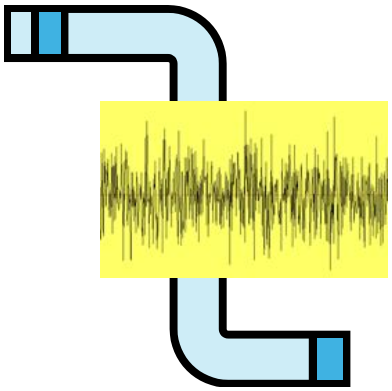
"Akash English"

C(x) is a "codeword"

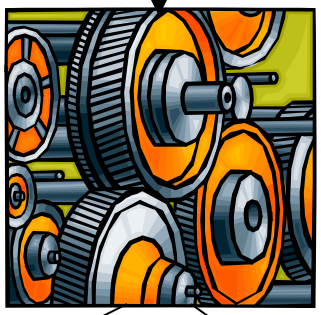
The setup

x

C(x)



y = C(x)+error



x

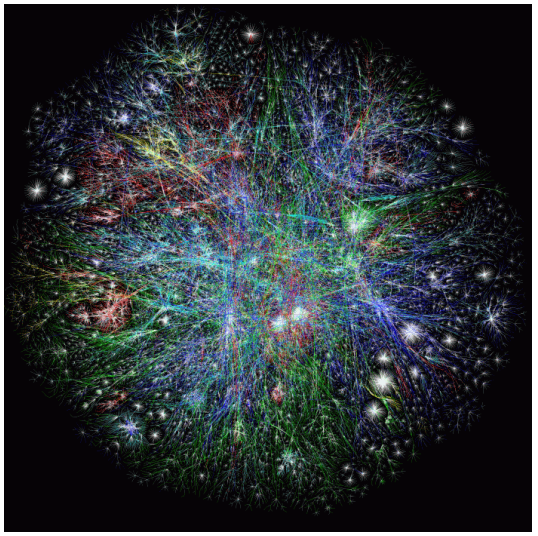
Give up

Mapping **C**

- Error-correcting code or just code
- Encoding: x → C(x)
- Decoding: y → x
- C(x) is a codeword

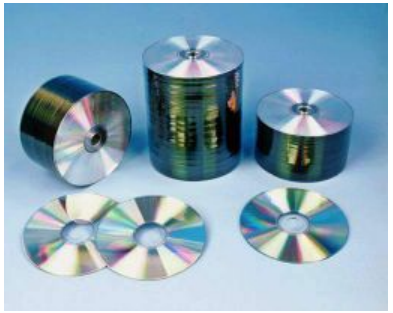
Different Channels and Codes

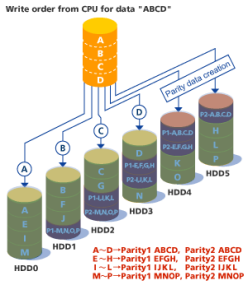
- Internet
  - Checksum used in mult layers of TCP/IP stack
- Cell phones
- Satellite broadcast
  - TV
- Deep space telecommunications
  - Mars Rover



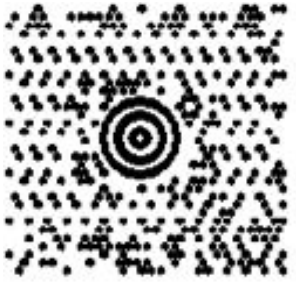
“Unusual” Channels

- Data Storage
  - CDs and DVDs
  - RAID
  - ECC memory





- Paper bar codes
  - UPS (MaxiCode)



Codes are all around us

Redundancy vs. Error-correction

- Repetition code: Repeat every bit say 100 times
  - Good error correcting properties
  - Too much redundancy
- Parity code: Add a parity bit
  - Minimum amount of redundancy
  - Bad error correcting properties
    - Two errors go completely undetected
- Neither of these codes are satisfactory

1 1 1 0 0	1

1 0 0 0 0	1

Two main challenges in coding theory

- Problem with parity example
  - Messages mapped to codewords which do not differ in many places
- Need to pick a lot of codewords that differ a lot from each other

- Efficient decoding
  - Naive algorithm: check received word with all codewords

The fundamental tradeoff

- Correct as many errors as possible with as little redundancy as possible

Can one achieve the "optimal" tradeoff with *efficient* encoding and decoding ?

Interested in more?

CSE 545, Spring 201?

Passwords

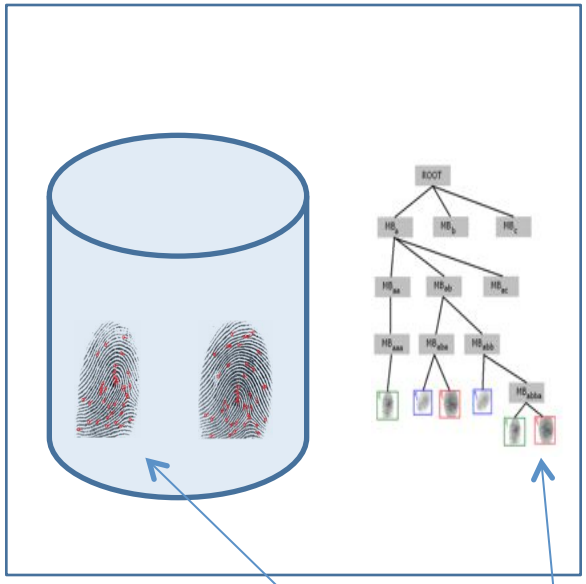
- Fast matching
- Secure
- "Cancelable"
- One can forget them!

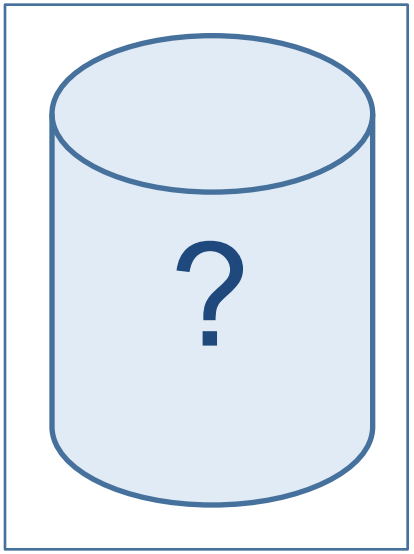
Fingerprints

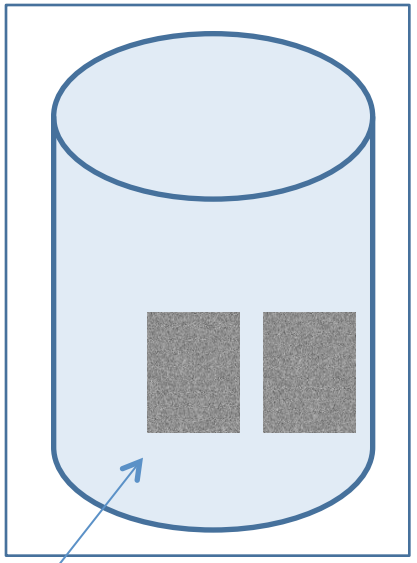
- \+ Fast matching
- ? Secure
- ? “Cancelable”
- \+ Cannot forget them!

Fingerprints as Passwords

Or making “Forgot password” links obsolete



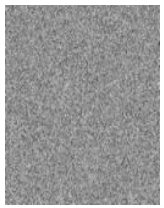




Only Fast

Fast & Secure

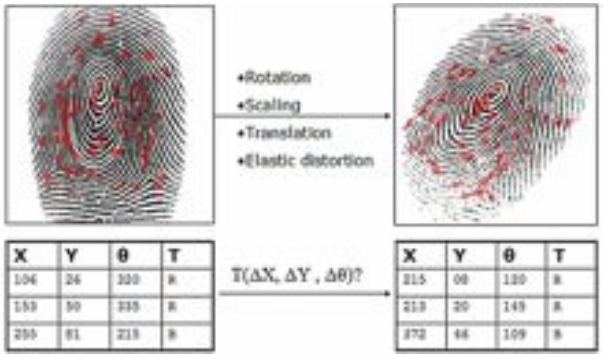
Only Secure



Challenges in Fingerprint Matching

Fingerprint readings are inconsistent

Matching Algorithms exist

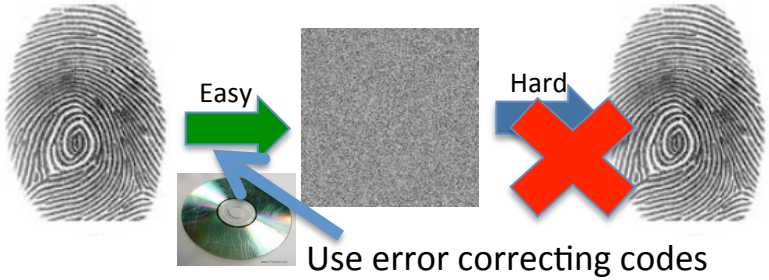


even in practice…

Security for fingerprints?

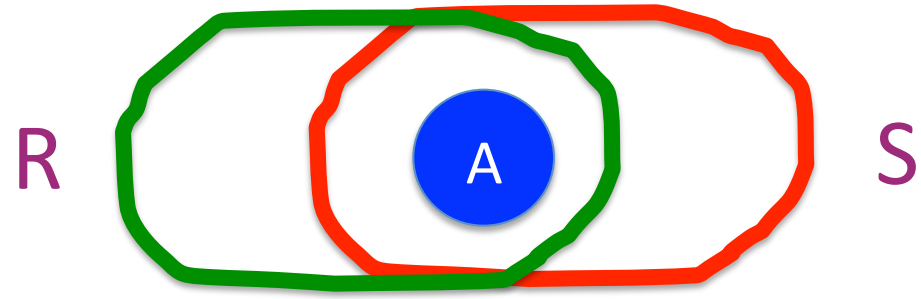
Stored fingerprints can be stolen

Main idea: obfuscate the fingerprint!



The simplest non-trivial join query

Intersection of R and S

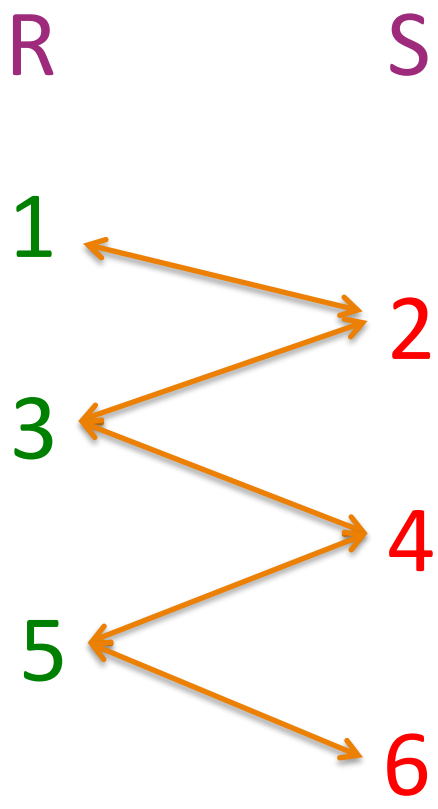


Assume R and S are sorted

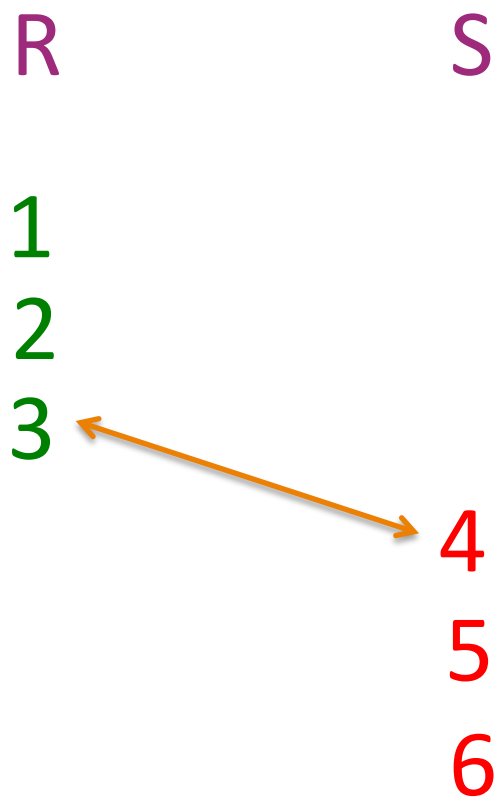
Let us concentrate on comparison based algorithms

Assume |R| = |S| = N

Not all inputs are created equal



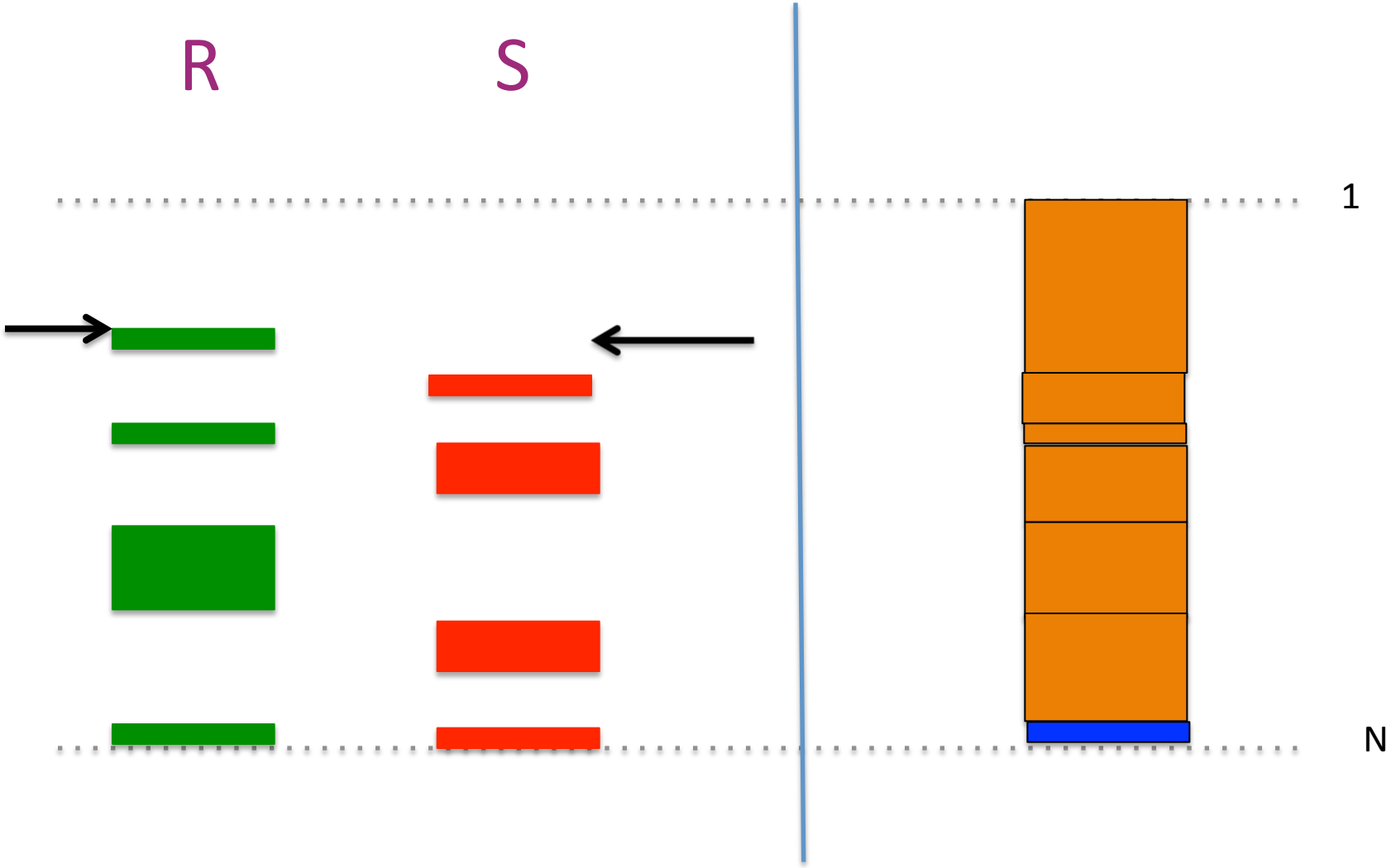
Ω(N) comparisons



1 comparison!

We need a faster/adaptive algorithm

The MERGE algorithm works



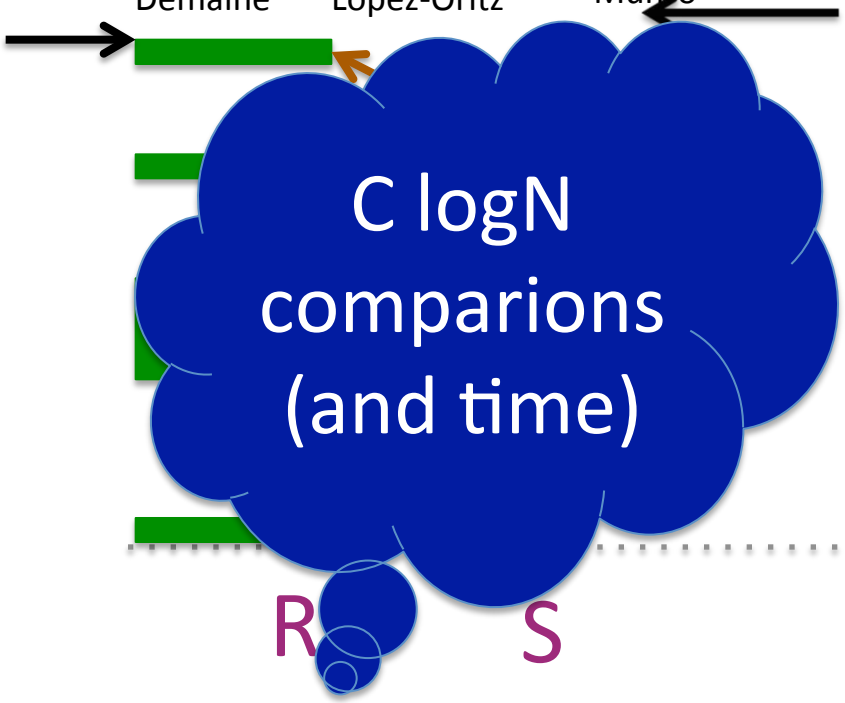
An assumption

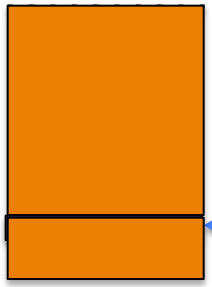
Output of the join is empty

MERGE is (near) instance optimal

Benchmark: Minimum number of comparisons (C) to "certify" output

Demaine Lopez-Oritz Munro





1

Value not ruled out yet

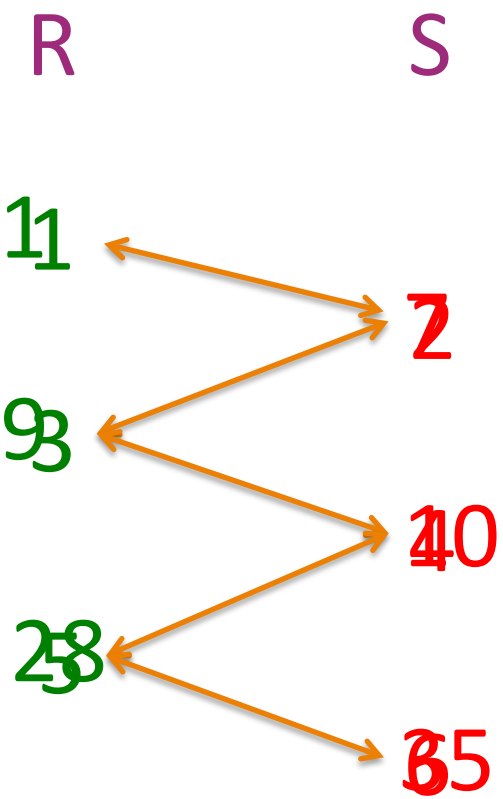
Need a comparison to rule the value out

Each value involved with ≤2 comparisons

Once the pointer moves the value is never seen again

Each move takes log N comparisons

Comparisons are value oblivious



A certificate is a set of comparison

s.t. for any instance satisfying all comparisons

output is empty

R[1] < S[1]

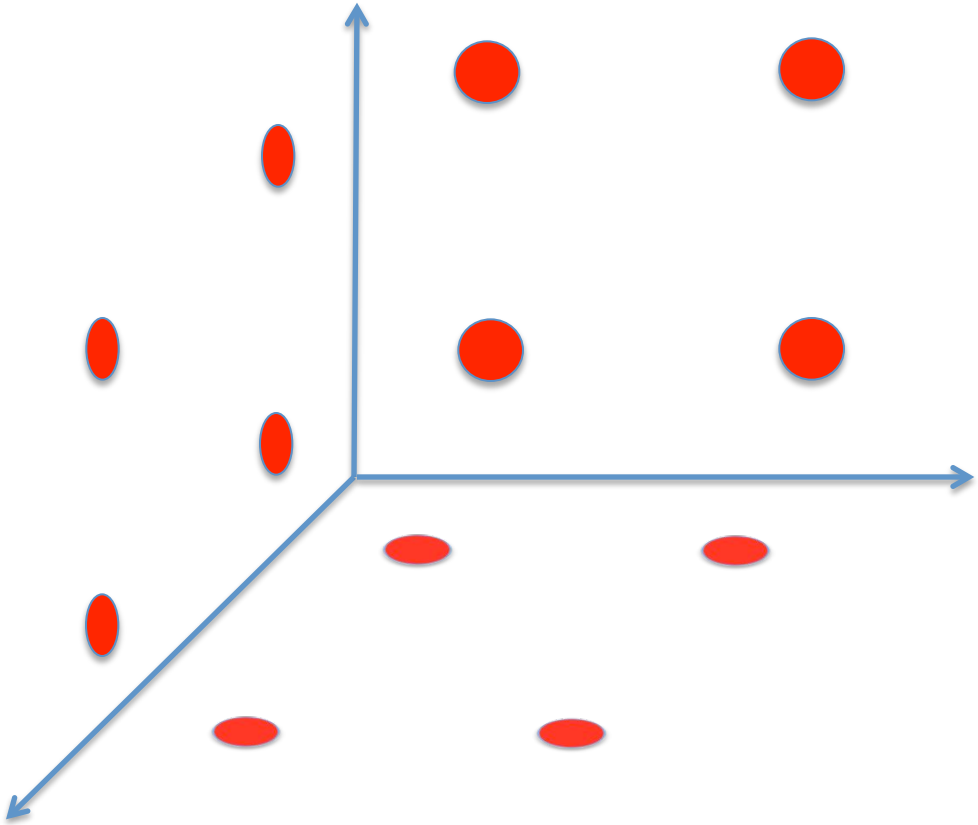
S[1] < R[2]

R[2] < S[2]

S[2] < R[3]

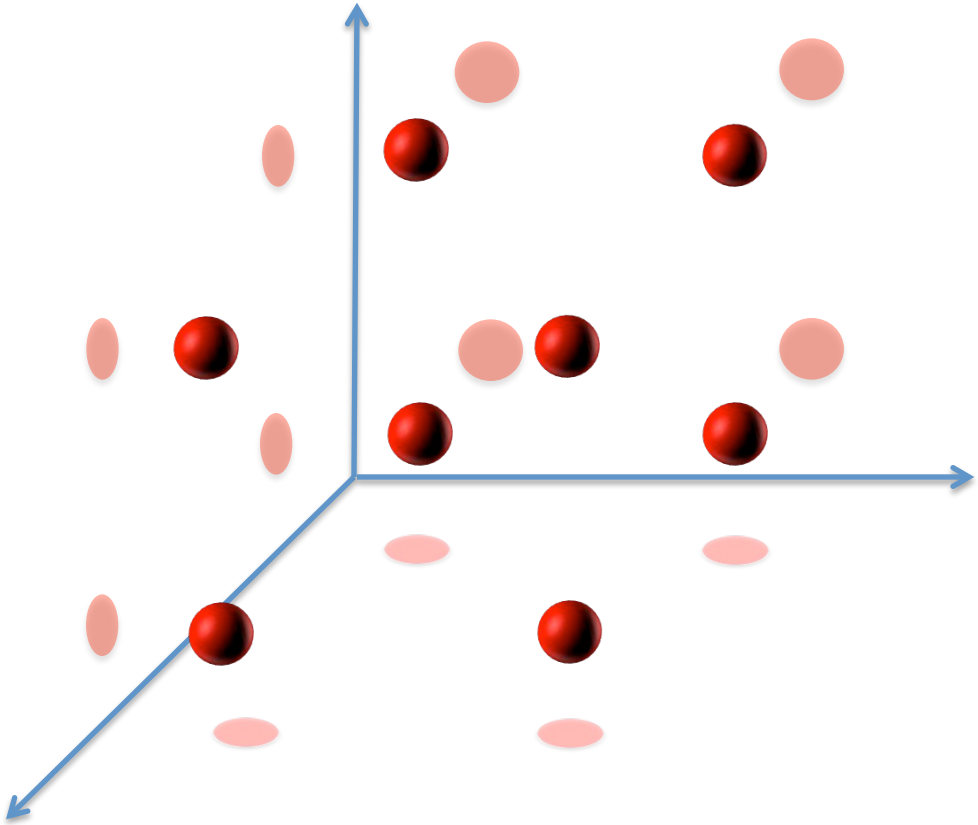
R[3] < S[3]

A toy problem



Given the three projections, what is the largest size of the original set of points?

The key technical problem

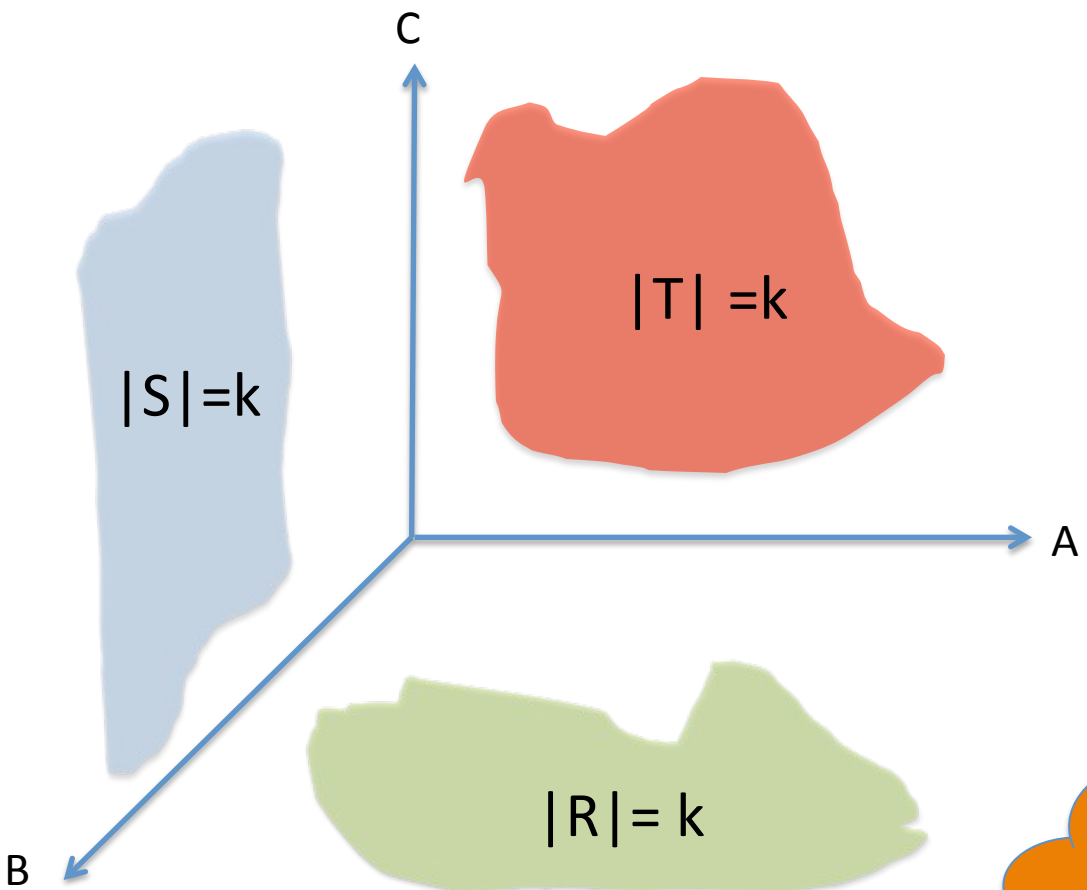


Highly trivial: $4^3 = 64$

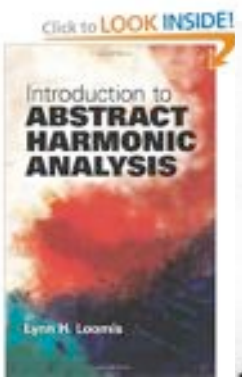
Still trivial: $4^2 = 16$

Correct answer: $4^{1.5} = 8$

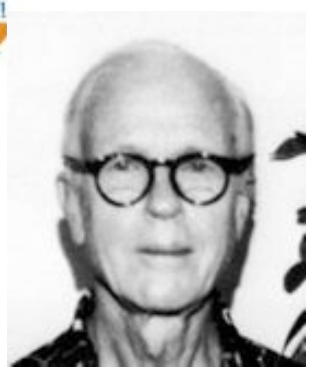
The key technical problem



$k^{3/2}$



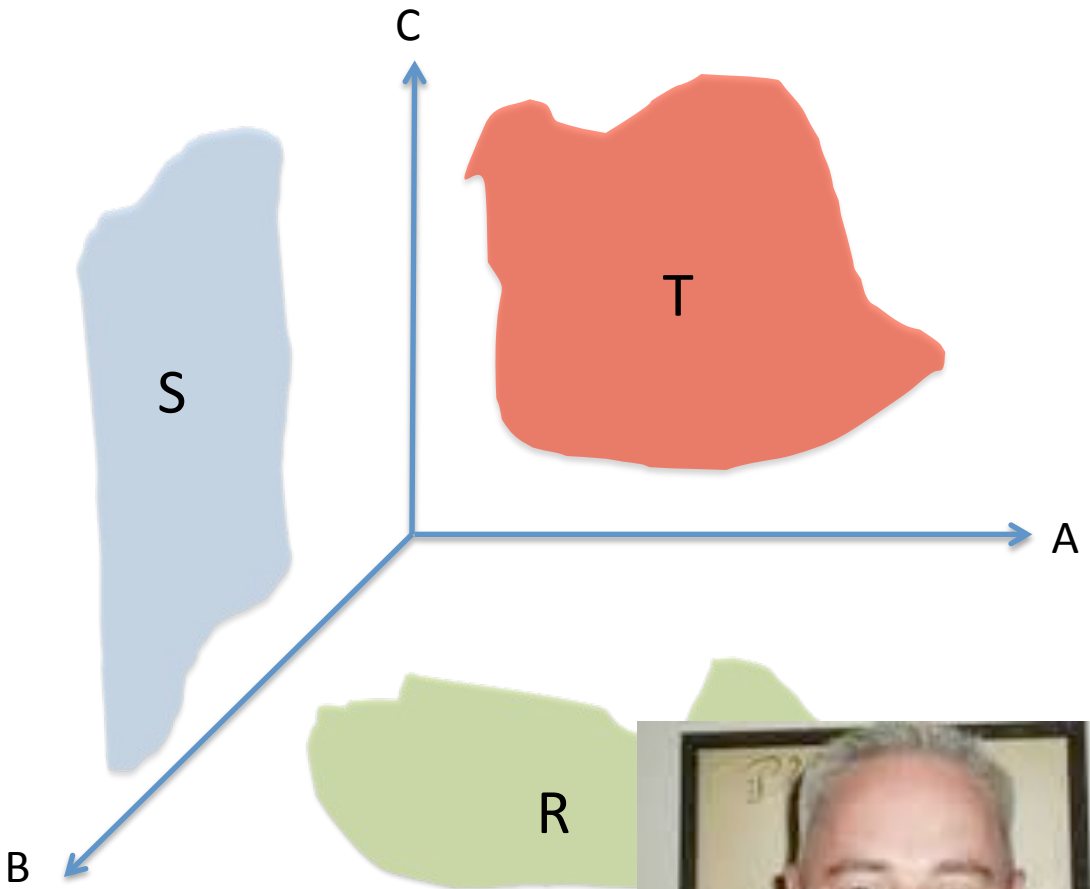
Loomis

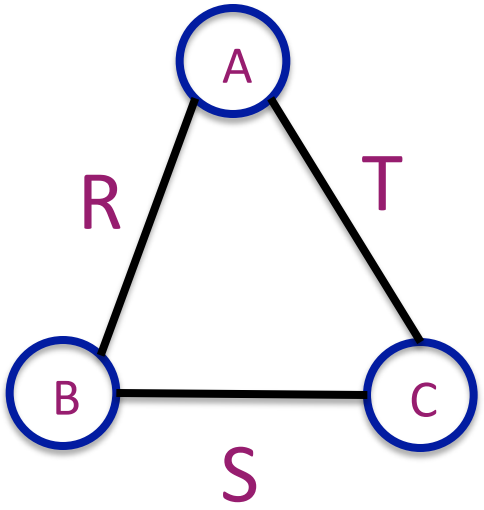


Whitney

Algorithmic Loomis-Whitney?

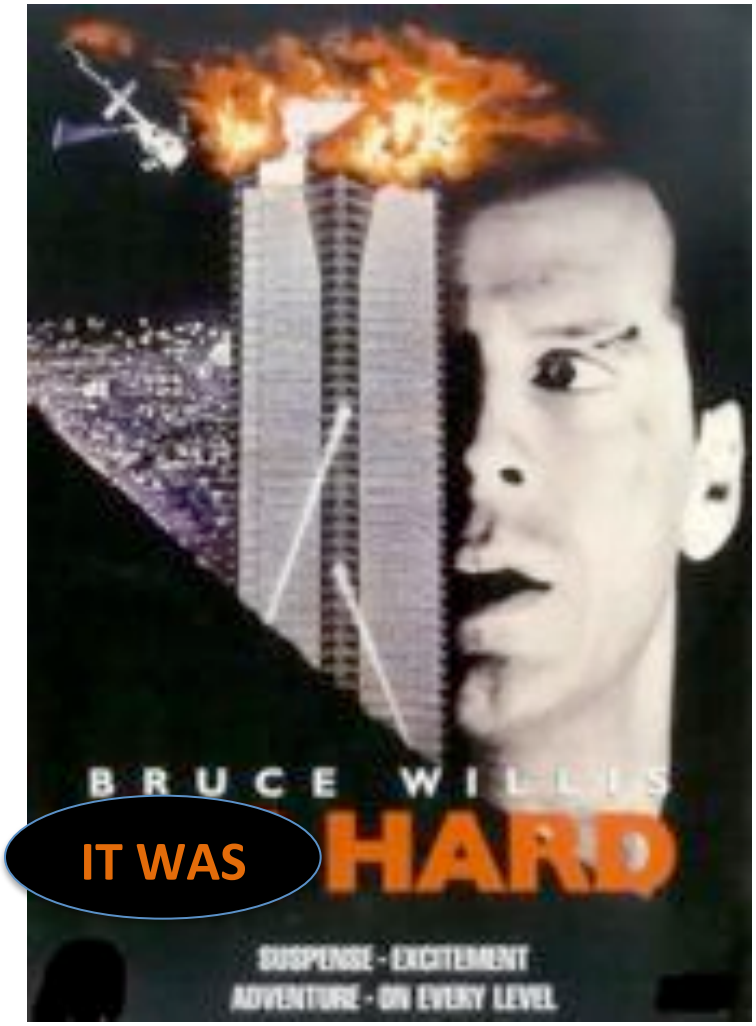
An equivalent view





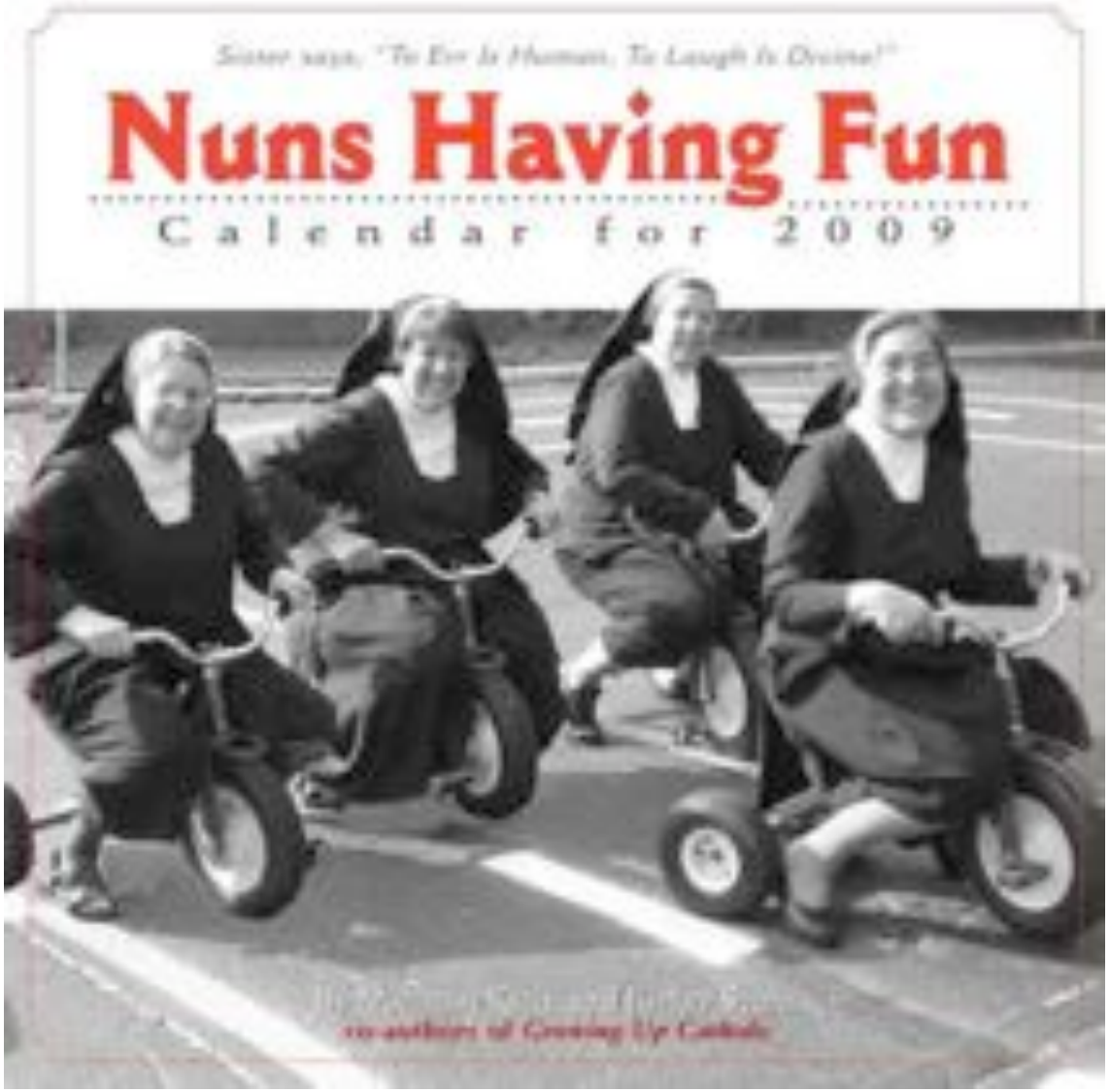
Output all (a,b,c) s.t. (a,b) in R, (b,c) in S and (c,a) in T

Whatever your impression of the 331



IT WAS

Hopefully it was fun!



Thanks!

Except of course, HW 10, quiz 2, presentations and the final exam