Lecture 39

CSE 331

Dec 6, 2017

Grading back on track

Re-grading request deadlines

note

stop following **13 views**

Re-grading request

I will be leaving for a trip to India on Dec 20, which basically means there will be a very quick turn-around after the final exam on Dec 15 (I hope to have the final exams graded by Dec 18).

In light of this, there will be strict re-grading request deadlines. **If you send in a re-grading request after the deadline, we will not consider them.**

So here are the dates:

- For everything up to Quiz 2: **Monday, Dec 11, 5pm**
- HW 9, HW 10 and final exam: Whichever is earlier:
    - One week after the grades have been released
    - **Monday, Dec 19, noon**

Please note that it is official policy for re-grading requests to be submitted within a week: see the HW policy document, so I'm just enforcing this now.

BTW remember the protocol for re-grading requests: contact the grader first and then me if needed.
#pin

grading

edit · good note | 0

Updated 6 minutes ago by Atri Rudra

Now relax…

Randomized algorithms

What is different?

Algorithms can toss coins and make decisions

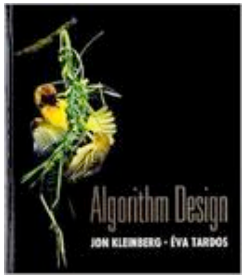
http://calculator.mathcaptain.com/coin-toss-probability-calculator.html

A Representative Problem

Hashing

Further Reading

Chapter 13 of the textbook



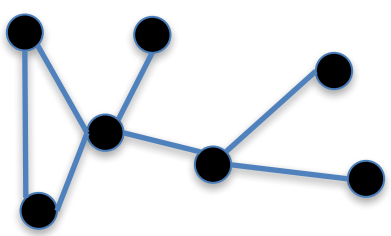
Approximation algorithms

What is different?

Algorithms can output a solution that is say 50% as good as the optimal

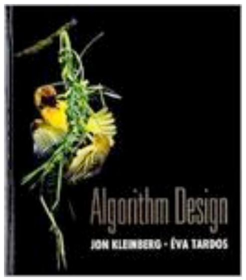
A Representative Problem

Vertex Cover



Further Reading

Chapter 12 of the textbook



Online algorithms

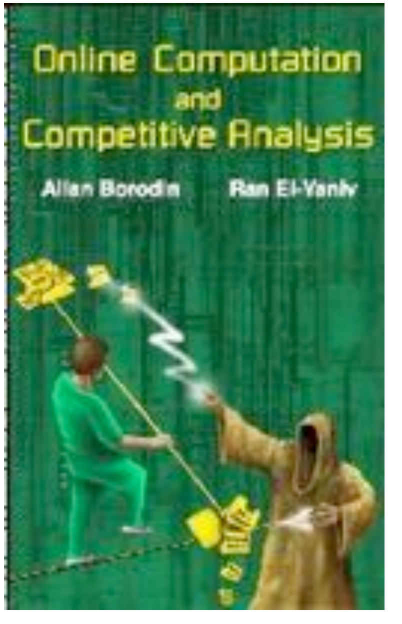
What is different?

Algorithms have to make decisions before they see all the input

A Representative Problem

Secretary Problem

Further Reading



Data streaming algorithms

https://www.flickr.com/photos/midom/2134991985/

What is different?

One pass on the input with severely limited memory

A Representative Problem

Compute the top-10 source IP addresses

Further Reading

Distributed algorithms

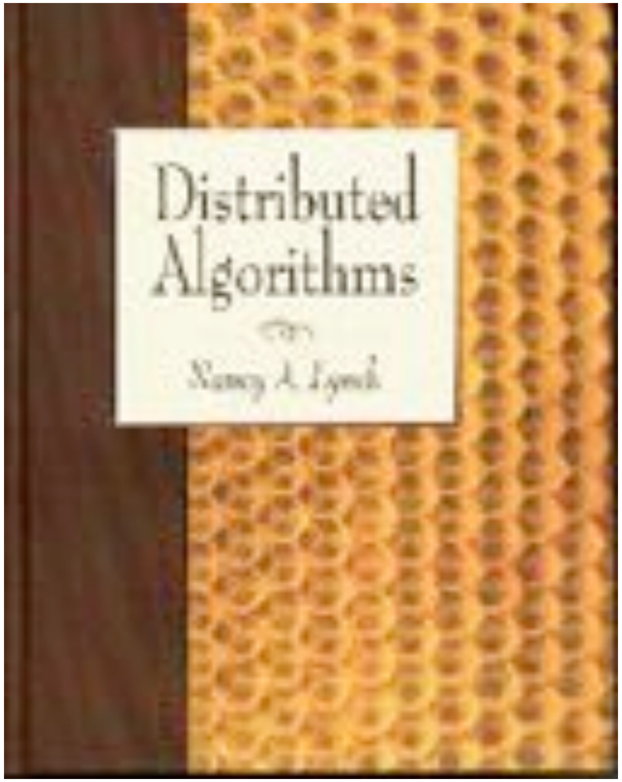
What is different?

Input is distributed over a network

A Representative Problem

Consensus

Further Reading



Beyond-worst case analysis

What is different?

Analyze algorithms in a more instance specific way

A Representative Problem

Intersect two sorted sets

Further Reading

http://theory.stanford.edu/~tim/f14/f14.html

Algorithms for Data Science

What is different?

Algorithms for non-discrete inputs

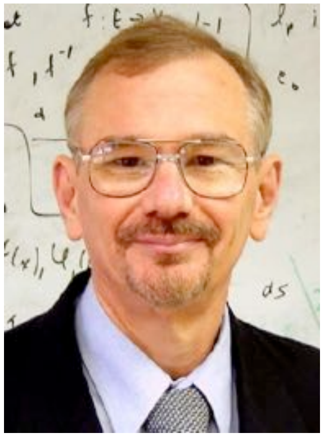
A Representative Problem

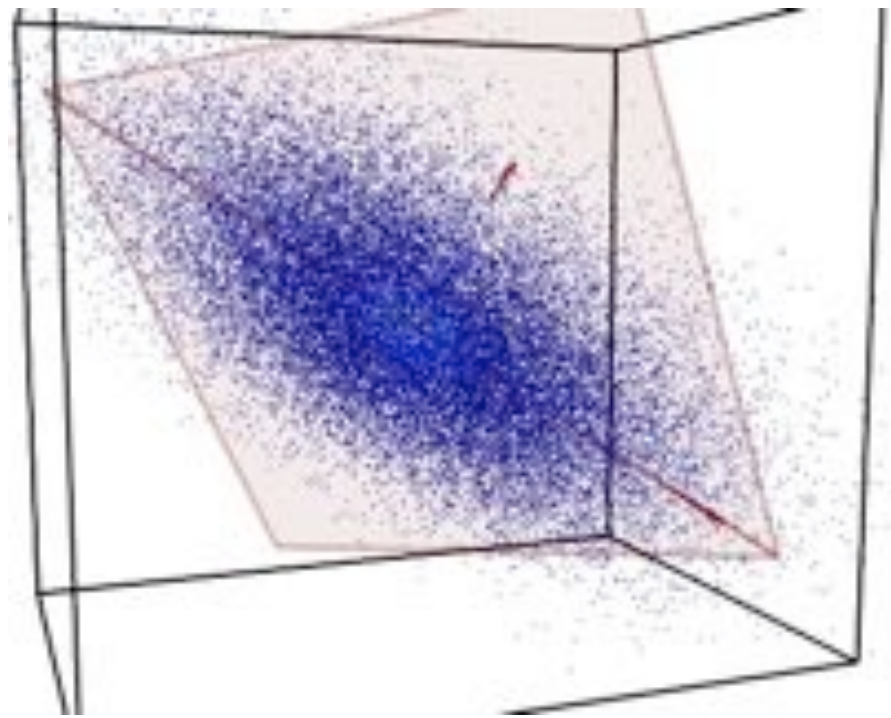
Compute Eigenvalues

Further Reading

http://research.microsoft.com/en-US/people/kannan/book-no-solutions-aug-21-2014.pdf

Johnson Lindenstrauss Lemma





http://www.scipy-lectures.org/_images/pca_3d_axis.jpg

Questions?

$Ax = y$

$$\begin{bmatrix} a_{0,0} & a_{0,2} & \cdots & a_{0,N-1} \\ \vdots & \vdots & \cdots & \vdots \\ a_{N-1,0} & a_{N-1,1} & \cdots & a_{N-1,N-1} \end{bmatrix} \times \begin{bmatrix} x_0 \\ \vdots \\ x_{N-1} \end{bmatrix} = \begin{bmatrix} y_0 \\ \vdots \\ y_{N-1} \end{bmatrix}$$

A $\quad$ x $\quad$ y

$\Theta(N^2)$ time in worst-case

In practice A has structure

$$\begin{bmatrix} a_{0,0} & a_{0,2} & \cdots & a_{0,N-1} \\ \vdots & \vdots & \cdots & \vdots \\ a_{N-1,0} & a_{N-1,1} & \cdots & a_{N-1,N-1} \end{bmatrix} \times \begin{bmatrix} x_0 \\ \vdots \\ x_{N-1} \end{bmatrix} = \begin{bmatrix} y_0 \\ \vdots \\ y_{N-1} \end{bmatrix}$$

$A \qquad x \qquad y$

Can we exploit the structure for faster algorithms?

Discrete Fourier Transform

$$\underbrace{\begin{bmatrix} a_{0,0} & a_{0,2} & \cdots & a_{0,N-1} \\ \vdots & \vdots & \cdots & \vdots \\ a_{N-1,0} & a_{N-1,1} & \cdots & a_{N-1,N-1} \end{bmatrix}}_{A} \times \underbrace{\begin{bmatrix} b_{10} \\ \vdots \\ b_{N-1} \end{bmatrix}}_{b}$$

$$a_{x,y} = \exp(2\pi i\, x \cdot y/N)$$

Cooley

Tukey

FFT (1965)

Can compute DFT in O(N log N) time

Cauchy Matrix

$$\begin{bmatrix} a_{0,0} & a_{0,2} & \cdots & a_{0,N-1} \\ \vdots & \vdots & \cdots & \vdots \\ a_{N-1,0} & a_{N-1,1} & \cdots & a_{N-1,N-1} \end{bmatrix} \times \begin{bmatrix} b_{10} \\ \vdots \\ b_{N-1} \end{bmatrix}$$

A b

Can be computed in $O(N \log^2 N)$ time

$$a_{x,y} = \frac{1}{r_x - s_y}$$

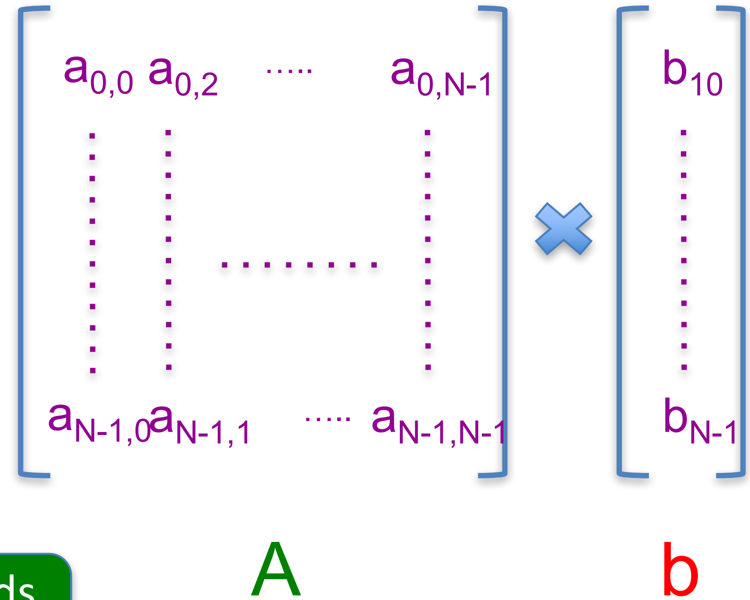
Superfast = N poly-log(N)

The main Question

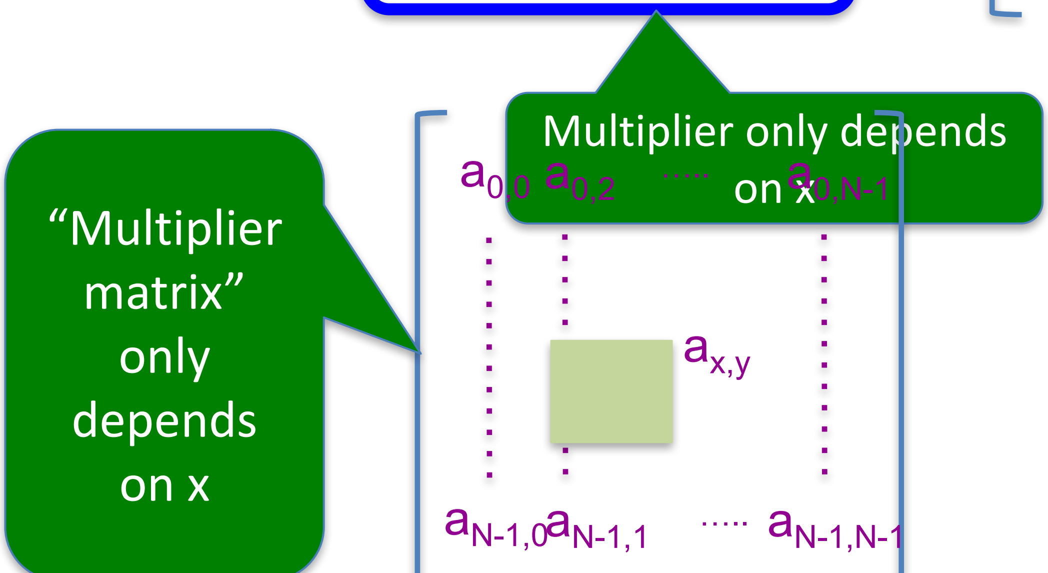
What is the largest class of matrices A for which we can have superfast algo to compute Ax?

Structure 1: Recurrence

$$a_{x,y} = \exp(2\pi i\, x \bullet y/N)$$



$$a_{x,y+1} = a_{x,y} \bullet \exp(2\pi i\, x/N)$$



Structure 2: Low Displacement Rank

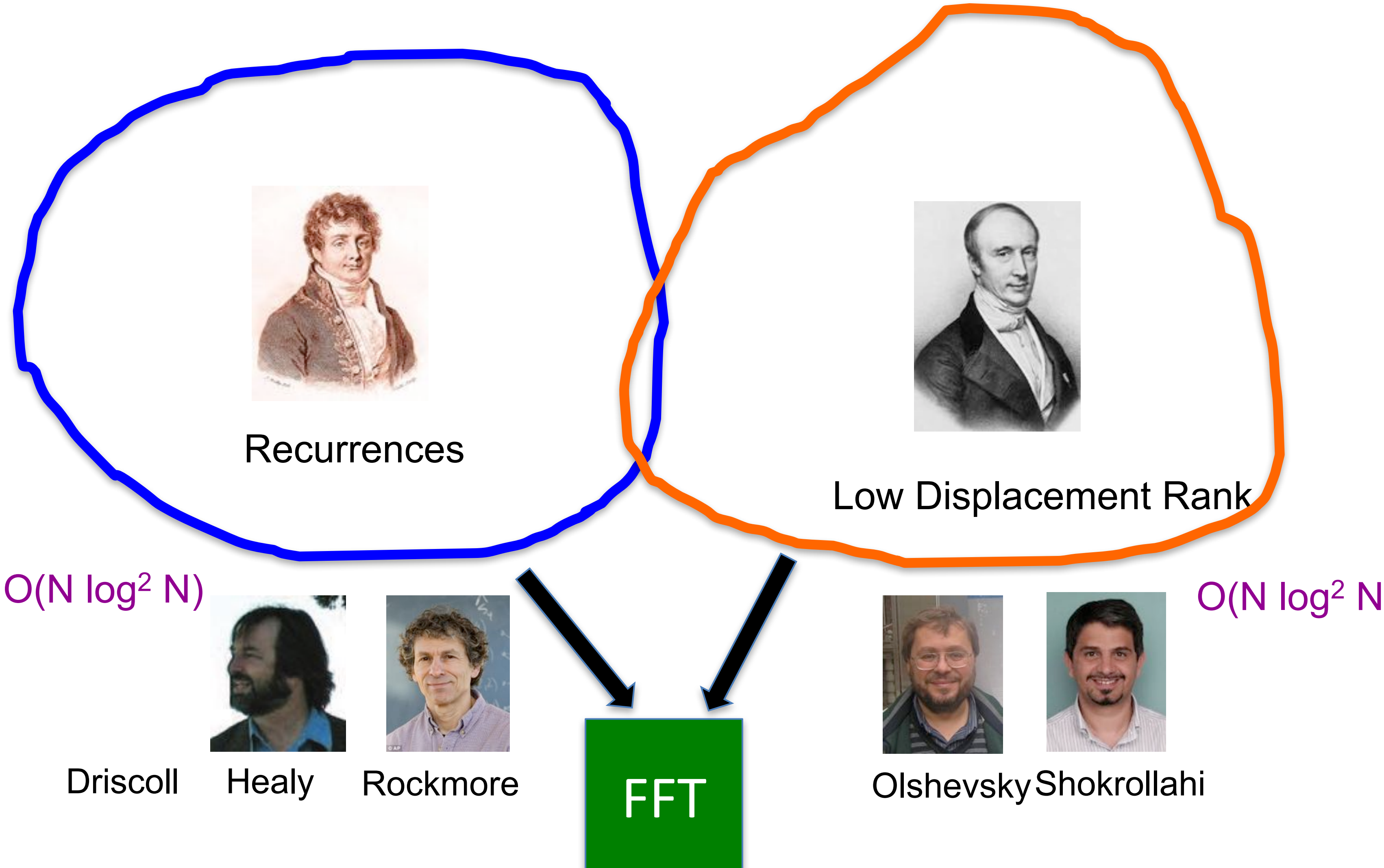
$$a_{x,y} = \frac{1}{r_x - s_y}$$

LA – AR has low rank

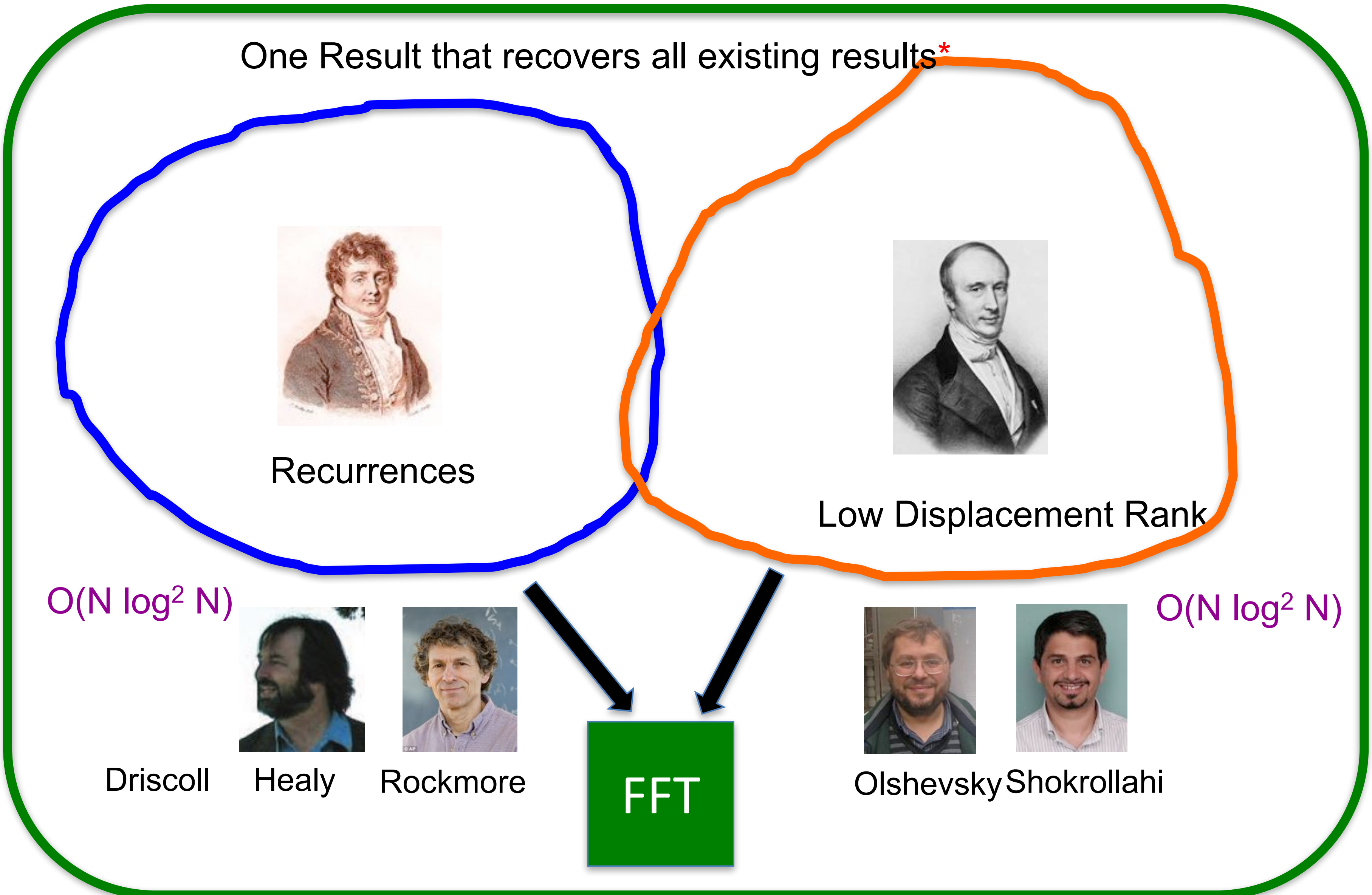
$$r_x \bullet a_{x,y} - a_{x,y} \bullet s_y = 1$$

$$\begin{bmatrix} r_0 & & 0 \\ & \ddots & \\ 0 & & r_{N-1} \end{bmatrix} \times A - A \times \begin{bmatrix} s_0 & & 0 \\ & \ddots & \\ 0 & & s_{N-1} \end{bmatrix} = 1$$

Known Results

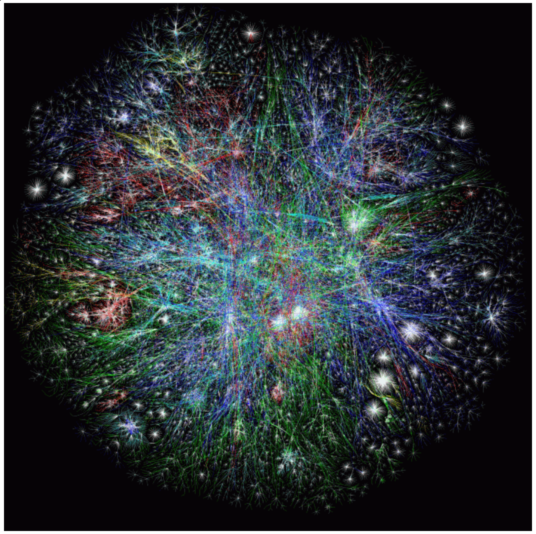


Our Main Result*



Questions?

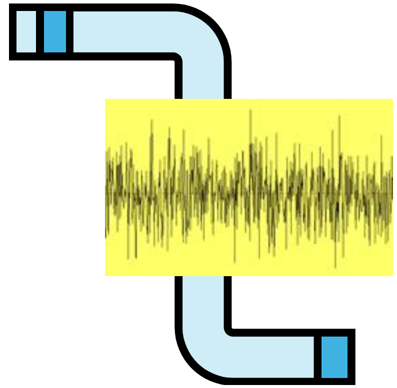
Coding Theory



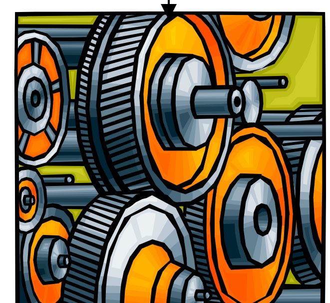
Communicating with my 3 year old

x

C(x)



y = C(x)+error



x

Give up

"Code" **C**

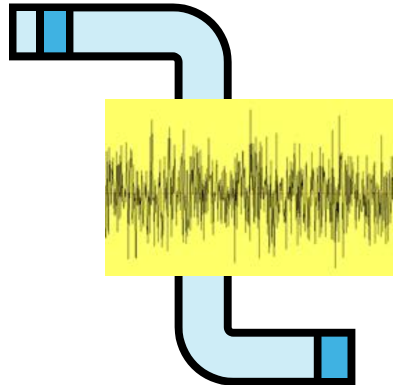
"Kiran English"

C(x) is a "codeword"

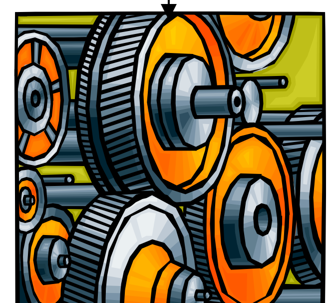
The setup

x

C(x)



y = C(x)+error



x

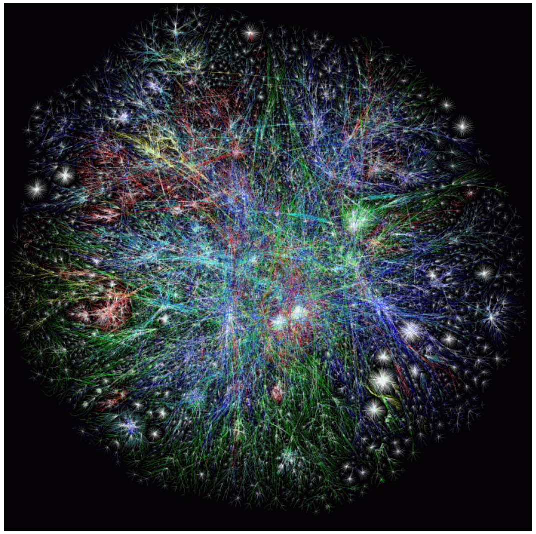
Give up

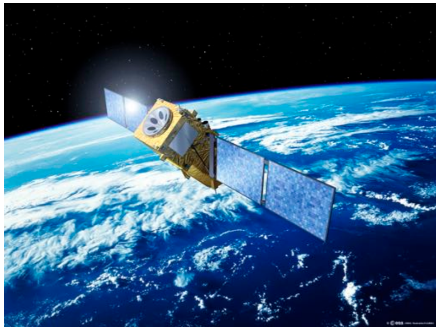
Mapping **C**

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

Different Channels and Codes

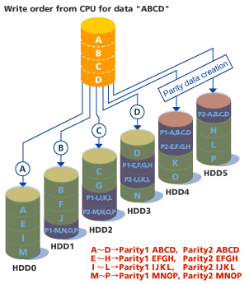
- Internet
  - Checksum used in mult layers of TCP/IP stack
- Cell phones
- Satellite broadcast
  - TV
- Deep space telecommunications
  - Mars Rover





"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

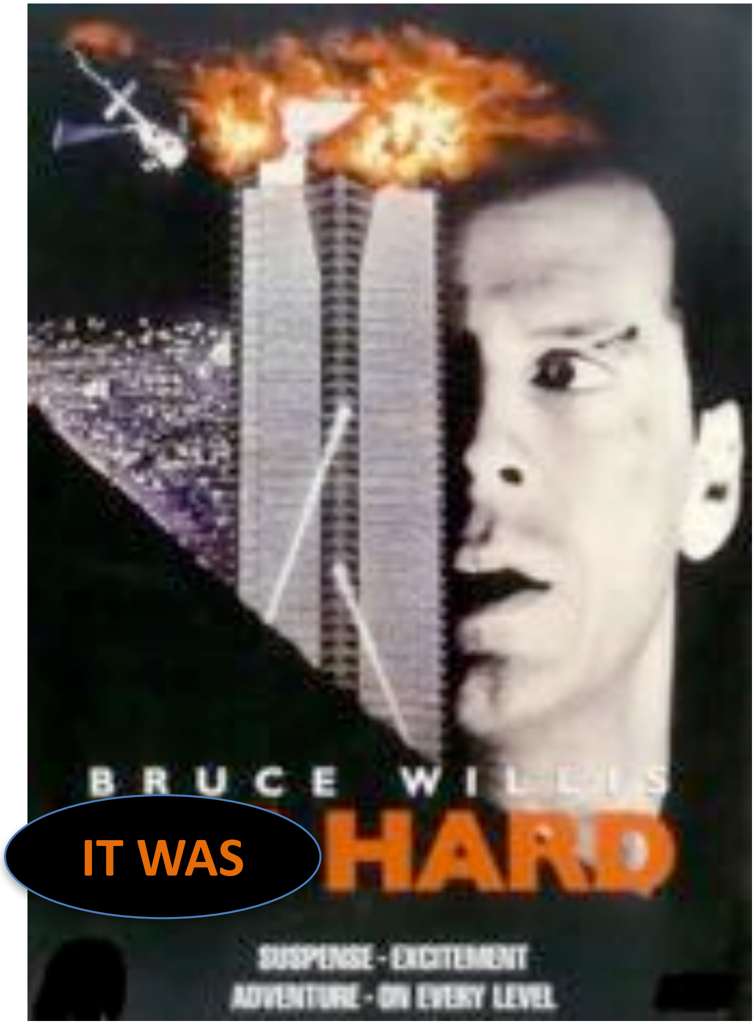
- Correct as many errors as possible with as little redundancy as possible

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

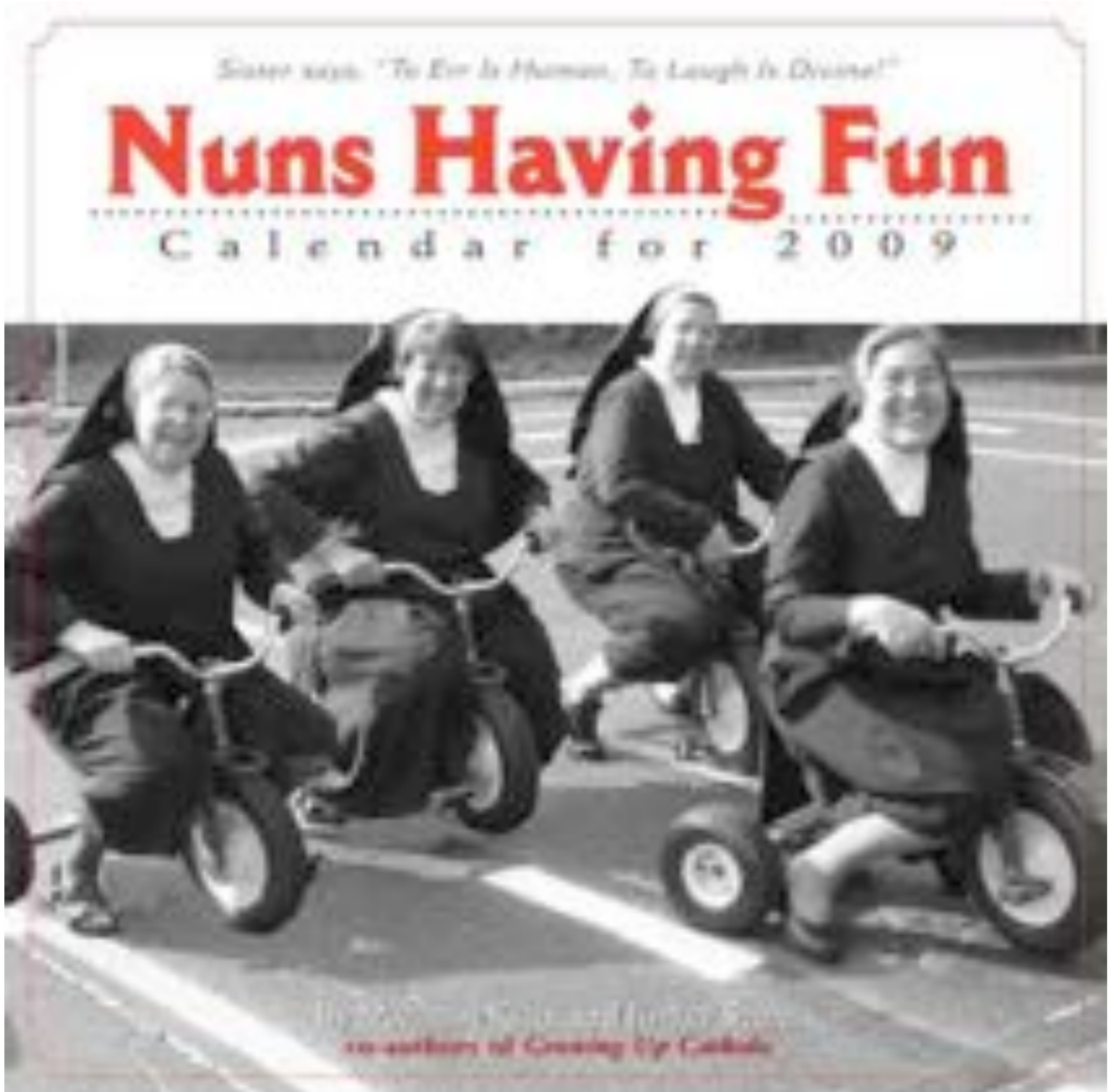
Interested in more?

CSE 545, Spring 2019

Whatever your impression of the 331



Hopefully it was fun!

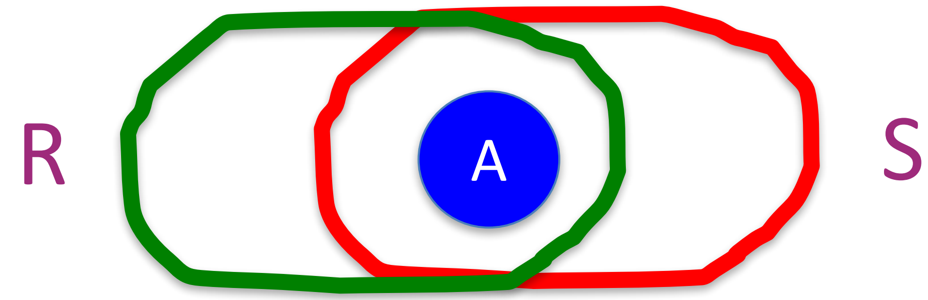


Thanks!

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

The simplest non-trivial join query

Intersection of R and S

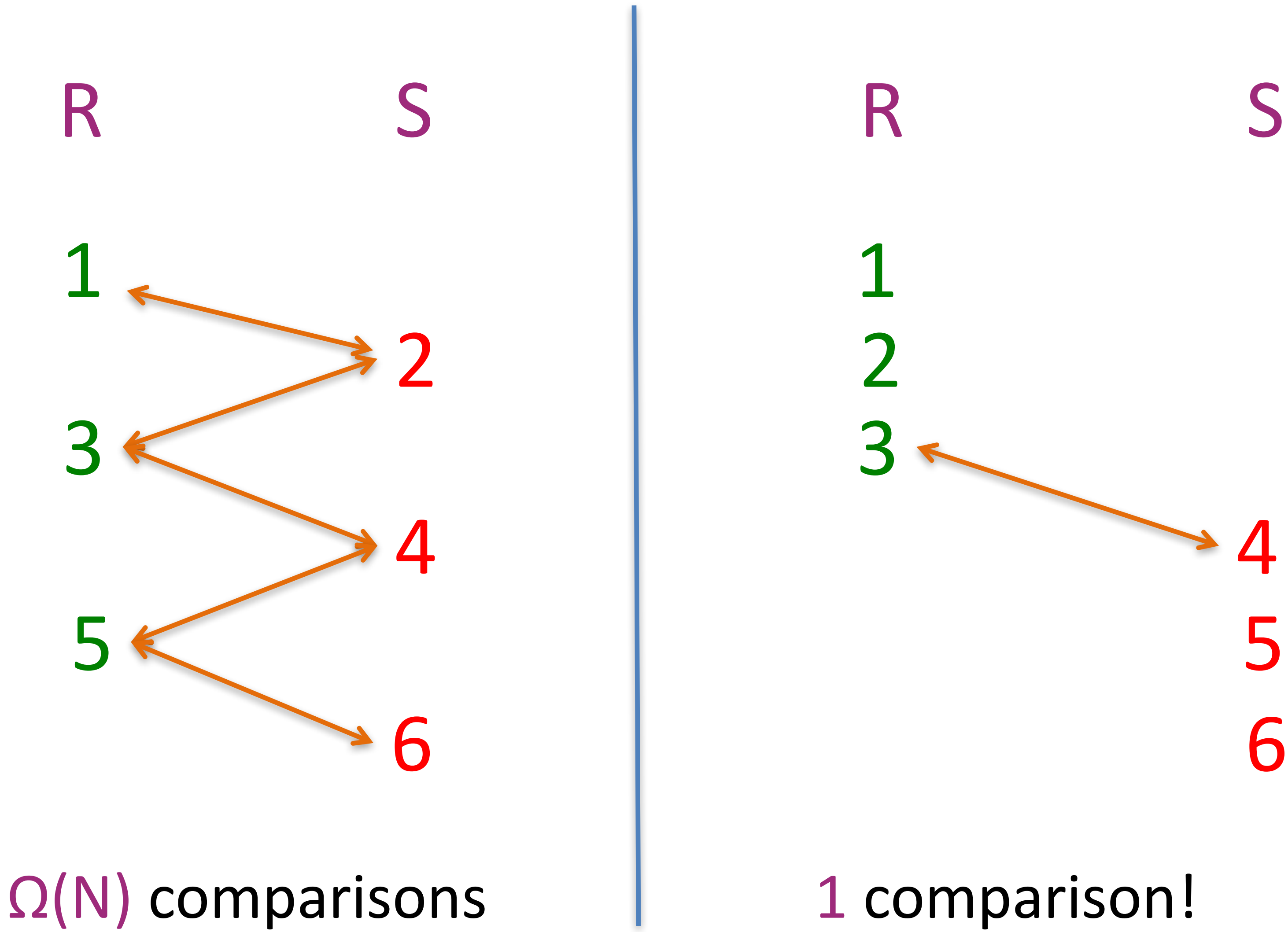


Assume R and S are sorted

Let us concentrate on comparison based algorithms

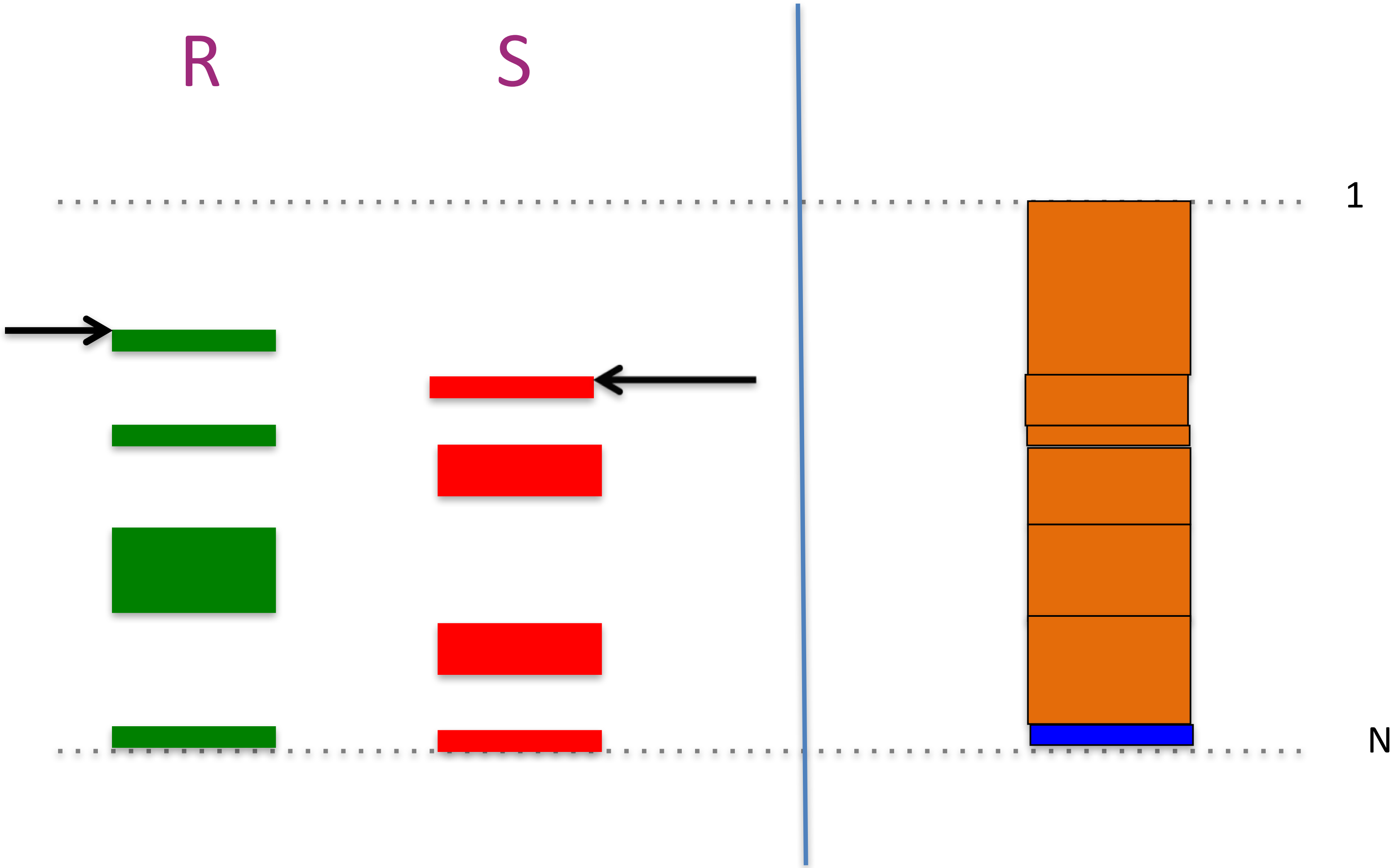
Assume $|R| = |S| = N$

Not all inputs are created equal



We need a faster/adaptive algorithm

The MERGE algorithm works

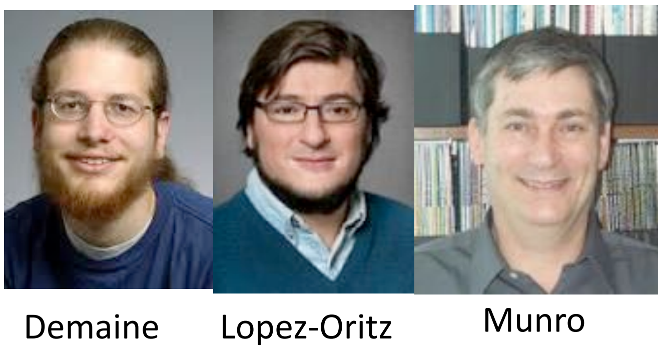


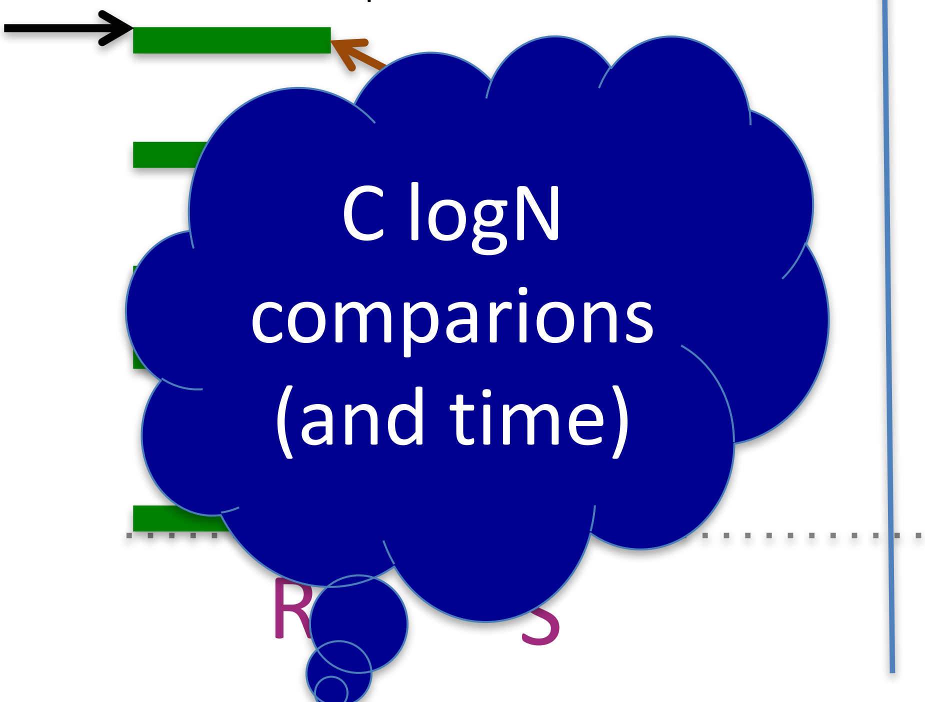
An assumption

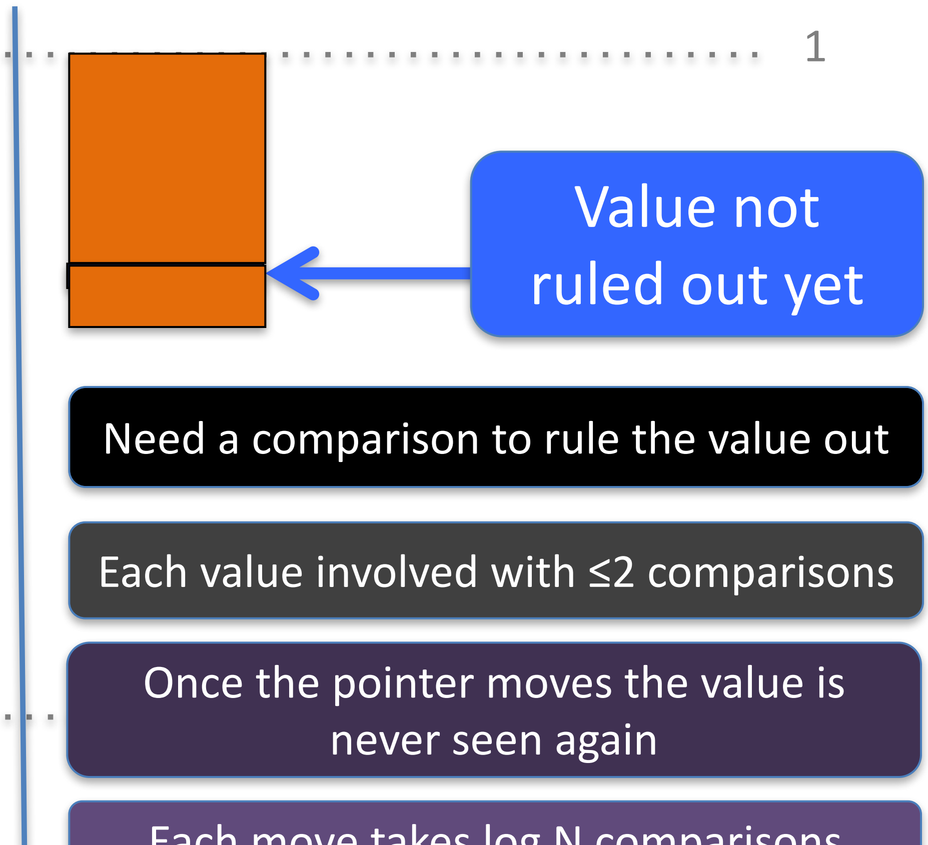
Output of the join is empty

MERGE is (near) instance optimal

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







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