CODES for SPEED:

the CLEAR advantage
(Computation, LEarning, Access, and Recovery)

Kannan Ramchandran

University of California, Berkeley
Shannon’s legacy

**SOURCE CODING**
Can represent a source using *bits > entropy*

**CHANNEL CODING**
Can transmit with arbitrarily small error probability at *rates < channel capacity*

*1948*
Impact of coding theory

Source coding
(1952) Huffman
(1967) Tunstall
(1976) Arithmetic
(1977) Lempel-Ziv

Channel coding
Hamming (1947)
Reed-Solomon (1960)
LDPC (1963)
Turbo (1990)
Polar (2009)
Primer on error correcting codes

- 3 data chunks $X_1, X_2, X_3$ to deliver
- 4 skiers, each can carry 1 chunk-size data
- You know 1 of them will get delayed but don’t know which

$X_1 \quad X_2 \quad X_3 \quad X_4 = X_1 + X_2 + X_3$
Primer on error correcting codes

- **3 data chunks** $X_1, X_2, X_3$ to deliver
- **4 skiers**, each can carry 1 chunk-size data
- You know **1 of them will get delayed** but don’t know which

\[
X_4 = X_1 + X_2 + X_3
\]

Recover $X_2 = X_4 - X_1 - X_3$
Error correcting codes

Used extensively in **communications** and **storage**

**Low density parity check (LDPC) codes**
- Wi-Fi 802.11
- DVB-S2
- 10GBase-T Ethernet
- WiMAX

**Reed-Solomon (RS) codes**
- Cloud storage
- QR codes
- CD, DVD, Blu-ray
- DSL
- RAID 6
- WiMAX
Outward view of codes
The CLEAR advantage

- Computing
- LEarning
- Access
- Recovery
Outline

Access

Recovery

LEarning

Computing
Chapter 1

Speeding up data access in distributed storage systems
Hadoop Distributed File System (HDFS)
Default 3x replication

640 MB file => 10 blocks

FB Warehouse Cluster
• Multiple tens of PBs and growing
• Multiple thousands of nodes

Storage efficiency ➔ main driver for cost
HDFS-RAID: (14, 10) MDS code

<p>| | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>P1</td>
<td>P2</td>
<td>P3</td>
<td>P4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

640 MB file => 10 blocks

FB Warehouse Cluster
- Reduce storage overhead from x3 to x1.4 (53% reduction)
- For less frequently accessed data

- What about the *speed of access*? (especially in the event of unavailability)
Codes are good but...

**Good news:** we can now tolerate 4 node failures.
Codes are good but...

**Good news:** we can now tolerate 4 node failures.

User wants 3 but the server is unavailable
Codes are good but...

**Good news:** we can now tolerate 4 node failures.

User wants 3 but the server is unavailable
Codes are good but...

**Good news:** we can now tolerate 4 node failures.

User wants **3** but the server is unavailable

Read from any 10 nodes, send all data to 3’ to repair the lost block.
Codes are good but...

**Good news:** we can now tolerate 4 node failures.

User wants but the server is unavailable

Read from any 10 nodes, send all data to 3’ to repair the lost block.

**Bad news:**
- ‘Degraded’ read is slow
- High network traffic
- High disk read (10x more than the lost information)
Can we have

- Storage efficiency of codes
- BW efficiency (speed) of replication

Almost...

... there exists (an optimal) tradeoff

**Regenerating Codes**

*Dimakis, Godfrey, Wainwright & R, 2010*
Storage-Bandwidth tradeoff

<table>
<thead>
<tr>
<th>Code</th>
<th>Storage</th>
<th>Bandwidth</th>
<th>Savings</th>
</tr>
</thead>
<tbody>
<tr>
<td>RS</td>
<td>1 MB</td>
<td>20 MB</td>
<td></td>
</tr>
<tr>
<td>MSR</td>
<td>1 MB</td>
<td>2 MB</td>
<td>10x</td>
</tr>
<tr>
<td>MBR</td>
<td>1.33 MB</td>
<td>1.33 MB</td>
<td>15x</td>
</tr>
</tbody>
</table>

File = 20 MB, \((n=40, k=20)\) code

Repair BW

Storage

Min. Bandwidth regime

Min. Storage regime

MBR

MSR

Reed-Solomon
Coding for Storage: Active Area of Research

**Goal:** Improve access/recovery time

- **Erasure Code**
- **Replication**

- More...

**Research Papers:**
- Rashmi et al. [2011]
- Cadambe et al. [2013]
- Tamo et al. [2014]
- Tian [2014]
- Sasidharan et al. [2015]
- Goparaju et al. [2015]
- Ye-Barg [2017]
- Chowdhury-Vardy [2017]
- Vajha et al. [2018]
- More...

**Locally Recoverable Codes**
- Papailiopoulos-Dimakis [2013]
- Tamo et al. [2013]
- Rawat et al. [2014]
- Tamo-Barg [2014]
- Prakash et al. [2014]
- Cadambe-Mazumdar [2015]
- Balaji et al. [2017]
- More...

**Other Variants of Repair-Efficient Codes**
- \( \epsilon \)-MSR codes
- Maximally Recoverable Codes
- Disjoint Recovering Sets (Availability Codes)
- Locality + Regeneration
- More...

**Codes with Redundant Request**
- Joshi et al. [2012]
- Liang-Kozat [2013]
- Shah-Lee-R [2014]
- Li et al. [2015]
- Gardner et al. [2015]
- More...

**Repairing RS Codes**
- Guruswami-Wooters [2016]
- Dau-Milenkovic [2017]
- Dau et al. [2017]
- Tamo-Ye-Barg [2017]
- More...
<table>
<thead>
<tr>
<th>Industry impact</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Piggyback codes &amp; Hitchhiker</strong></td>
</tr>
<tr>
<td>[Rashmi et al. ISIT 13, IT-Tran 17]</td>
</tr>
<tr>
<td>[Rashmi et al. SIGCOMM 14]</td>
</tr>
<tr>
<td><img src="https://example.com/facebook.png" alt="Facebook" /> <img src="https://example.com/apache.png" alt="Apache" /> <img src="https://example.com/netapp.png" alt="NetApp" /></td>
</tr>
<tr>
<td><strong>Butterfly codes</strong></td>
</tr>
<tr>
<td>[En Gad et al. ISIT 13]</td>
</tr>
<tr>
<td>[Pamies-Juarez et al. FAST 16]</td>
</tr>
<tr>
<td><img src="https://example.com/western-digital.png" alt="Western Digital" /></td>
</tr>
<tr>
<td><strong>Ye-Barg codes &amp; Clay codes (IT-Tran 17, FAST 18)</strong></td>
</tr>
<tr>
<td>[Ye and Barg IT-Tran 17]</td>
</tr>
<tr>
<td>[Vajha et al. FAST 18]</td>
</tr>
<tr>
<td><img src="https://example.com/ceph.png" alt="Ceph" /></td>
</tr>
</tbody>
</table>
Chapter 2

Speeding up learning and recovery of sparse signals
Motivation
Motivation

- Given training data points \((x, y)\), our goal is to learn

\[(x : \text{feature}, y : \text{label})\]
Motivation

- Given training data points $(x, y)$, our goal is to learn
  - a certain rule $f$

$(x : \text{feature}, y : \text{label})$
Motivation

- Given training data points \((x, y)\), our goal is to learn
  - a certain rule \(f\) that explains the label \(y\)

\[(x : \text{feature}, y : \text{label})\]

Dataset

\[y = f(x)\]
Motivation

- Given training data points \((x, y)\), our goal is to learn
  - a certain rule \(f\) that explains the label \(y\) based on features \(x\):

\[
(x : \text{feature}, y : \text{label})
\]

\[
\begin{array}{c}
x \rightarrow f(\cdot) \\
y = f(x)
\end{array}
\]
Motivation

- Given training data points \((x, y)\), our goal is to learn
  - a **certain rule** \(f\) that explains the label \(y\) based on features \(x\):

\[
(x : \text{feature, } y : \text{label}) \quad \xrightarrow{} \quad x \xrightarrow{f(\cdot)} y = f(x)
\]
Motivation

- Given training data points \((x, y)\), our goal is to learn
  
  - a certain rule \( f \) that explains the label \( y \) based on features \( x \):

\[
(x : \text{feature}, y : \text{label})
\]

Dataset

\[
x \xrightarrow{f(\cdot)} y = f(x)
\]

(e.g. area, bedrooms)

(e.g. house prices)
Motivation

- Given training data points \((x, y)\), our goal is to learn a certain rule \(f\) that explains the label \(y\) based on features \(x\):

\[
(x: \text{feature}, y: \text{label}) \quad \rightarrow \quad x \quad \rightarrow \quad f(\cdot) \quad \rightarrow \quad y = f(x)
\]

- Questions of interest
  - Sample complexity: how many data points do we need?
Motivation

- Given training data points \((x, y)\), our goal is to learn
  - a certain rule \(f\) that explains the label \(y\) based on features \(x\):

\[
(x : \text{feature}, y : \text{label}) \rightarrow x \rightarrow f(\cdot) \rightarrow y = f(x)
\]

(e.g. area, bedrooms) (e.g. house prices)

- Questions of interest
  - Sample complexity: how many data points do we need?
  - Computational complexity: how much time does it take?
Motivation

- Given training data points \((x, y)\), our goal is to learn
  - a certain rule \(f\) that explains the label \(y\) based on features \(x\):
    
    \[ (x : \text{feature}, y : \text{label}) \]

    \[ x \rightarrow f(\cdot) \rightarrow y = f(x) + \varepsilon \]

    
    (e.g. area, bedrooms) \quad (e.g. house prices)

- Questions of interest
  - Sample complexity: how many data points do we need?
  - Computational complexity: how much time does it take?
  - Robustness: how accurate and stable is it?
Motivation

- Given training data points \((x, y)\), our goal is to learn
  - a certain rule \(f\) that explains the label \(y\) based on features \(x\):

\[
(x : \text{feature}, y : \text{label}) \quad \xrightarrow{\text{Dataset}} \quad x \quad \xrightarrow{f(\cdot)} \quad f(x) + \varepsilon
\]

(e.g. area, bedrooms) \quad (e.g. house prices)

\[
e.g. \quad f(x_1, x_2) = a_1 x_1 + a_2 x_2 + b
\]
Motivation

- Given training data points \((x, y)\), our goal is to learn
  
  \[ y = f(x) + \varepsilon \]

  where \(f\) is a certain rule that explains the label \(y\) based on features \(x\):

  \((x: \text{feature}, y: \text{label})\)

  (e.g. area, bedrooms) \rightarrow f(\cdot) \rightarrow (e.g. house prices)

  e.g. \(f(x_1, x_2) = a_1 x_1 + a_2 x_2 + b\)

  Problem Dimension \(N = 3\)
Motivation

- Given training data points \((x, y)\), our goal is to learn
  
  - a certain rule \(f\) that explains the label \(y\) based on features \(x\):

\[
\begin{align*}
(x : \text{feature}, y : \text{label}) & \quad \xrightarrow{\text{Dataset}} \quad x \quad \xrightarrow{\text{f(\cdot)}} \quad f(x) + \varepsilon \\
(\text{e.g. area, bedrooms}) & \quad \xrightarrow{\text{f(\cdot)}} \quad \text{(e.g. house prices)}
\end{align*}
\]

\[
e.g. \quad f(x_1, x_2) = a_1 x_1 + a_2 x_2 + b
\]

EASY!

Problem Dimension \(N = 3\)
Motivation

- Given training data points \((x, y)\), our goal is to learn
  - a certain rule \(f\) that explains the label \(y\) based on features \(x\):

\[
(x : \text{feature}, y : \text{label}) \quad \xrightarrow{x} \quad f(\cdot) \quad \xrightarrow{y = f(x) + \varepsilon} \quad \text{(e.g. area, bedrooms) \quad (e.g. house prices)}
\]

\[
e.g. \quad f(x_1, x_2) = a_1x_1 + a_2x_2 + b
\]

However...

Problem Dimension \(N = 3\)
Motivation

- Given training data points \((x, y)\), our goal is to learn
  - a certain rule \(f\) that explains the label \(y\) based on features \(x\):

\[
(x : \text{feature}, y : \text{label}) 
\xrightarrow{\text{Dataset}} 
\xrightarrow{(\text{e.g. area, bedrooms})} 
\xrightarrow{\text{(e.g. house prices)}} 
\]

\[y = f(x) + \varepsilon\]

\text{e.g. } f(x_1, x_2) = a_1 x_1 + a_2 x_2 + b

in reality...

Problem Dimension \(N = 3\)
Motivation

- Given training data points \((x, y)\), our goal is to learn
  - a certain rule \(f\) that explains the label \(y\) based on features \(x\):

\[
(x : \text{feature, } y : \text{label}) \quad \xrightarrow{} \quad x \quad \xrightarrow{f(\cdot)} \quad y = f(x) + \varepsilon
\]

\[
e.g. \quad f(x) = \frac{\partial}{\partial t} \frac{1}{\beta} \left( \psi \left( \frac{\psi}{\beta} \right) - \varepsilon \right) + \frac{1}{\beta} \psi \psi \frac{\partial \psi}{\partial t} + \frac{1}{\beta} \psi \frac{\partial \psi}{\partial x} \frac{\partial \psi}{\partial t} + \frac{1}{\beta} \psi \frac{\partial \psi}{\partial x} \frac{\partial \psi}{\partial x} + \frac{1}{\beta} \psi \psi \frac{\partial \psi}{\partial t} + \frac{1}{\beta} \psi \frac{\partial \psi}{\partial t} \frac{\partial \psi}{\partial x} + \frac{1}{\beta} \psi \frac{\partial \psi}{\partial t} \frac{\partial \psi}{\partial x} + \frac{1}{\beta} \psi \frac{\partial \psi}{\partial x} \frac{\partial \psi}{\partial x} + \frac{1}{\beta} \psi \psi \frac{\partial \psi}{\partial t} + \frac{1}{\beta} \psi \frac{\partial \psi}{\partial t} \frac{\partial \psi}{\partial x} + \frac{1}{\beta} \psi \frac{\partial \psi}{\partial x} \frac{\partial \psi}{\partial x} + \frac{1}{\beta} \psi \psi \frac{\partial \psi}{\partial t} + \frac{1}{\beta} \psi \frac{\partial \psi}{\partial t} \frac{\partial \psi}{\partial x} + \frac{1}{\beta} \psi \frac{\partial \psi}{\partial x} \frac{\partial \psi}{\partial x} + \frac{1}{\beta} \psi \psi \frac{\partial \psi}{\partial t} + \frac{1}{\beta} \psi \frac{\partial \psi}{\partial t} \frac{\partial \psi}{\partial x} + \frac{1}{\beta} \psi \frac{\partial \psi}{\partial x} \frac{\partial \psi}{\partial x} + \frac{1}{\beta} \psi \psi \frac{\partial \psi}{\partial t} + \frac{1}{\beta} \psi \frac{\partial \psi}{\partial t} \frac{\partial \psi}{\partial x} + \frac{1}{\beta} \psi \frac{\partial \psi}{\partial x} \frac{\partial \psi}{\partial x}
\]

in reality...
Motivation

- Given training data points \((x, y)\), our goal is to learn

  - a certain rule \(f\) that explains the label \(y\) based on features \(x\):

\[
(x : \text{feature}, y : \text{label})
\]

\[
\begin{align*}
\text{Dataset} & \quad \xrightarrow{\text{x}} \quad f(\cdot) \quad \xrightarrow{\text{y}} \quad y = f(x) + \varepsilon \\
\text{e.g. } f(x) &= \frac{D}{\text{Dr}} \left( \frac{\text{w}^T}{\text{w}} + \frac{\text{w}^T \text{v}}{\text{w}^T} \right) + \frac{1}{\beta} \left( \frac{\text{w}^T \text{w}}{\text{w}^T} + \frac{\text{w}^T \text{w}}{\text{w}^T} - \frac{\text{w}^T \text{w}}{\text{w}^T} \right) + \frac{1}{\beta} \left( \frac{\text{w}^T \text{w}}{\text{w}^T} + \frac{\text{w}^T \text{w}}{\text{w}^T} - \frac{\text{w}^T \text{w}}{\text{w}^T} \right) \left( \frac{\text{w}^T \text{w}}{\text{w}^T} + \frac{\text{w}^T \text{w}}{\text{w}^T} - \frac{\text{w}^T \text{w}}{\text{w}^T} \right) \\
\end{align*}
\]

in reality…

Problem Dimension \(N \rightarrow \infty\)
Motivation

- Given training data points \((x, y)\), our goal is to learn

  - a certain rule \(f\) that explains the label \(y\) based on features \(x\):

\[
(x : \text{feature}, y : \text{label}) \quad \xrightarrow{\text{Dataset}} \quad x \xrightarrow{f(\cdot)} y = f(x) + \varepsilon
\]

\[
e.g. \quad f(x) = \frac{D}{\partial_t} \frac{\partial w}{\partial u} + \frac{\partial w}{\partial u} \frac{\partial u}{\partial t} - \left(\frac{\partial w}{\partial u} + \frac{\partial w}{\partial t}\right) v + \frac{\partial w}{\partial \Phi} + \frac{\partial w}{\partial \Phi} = \frac{3 + e_1}{\partial_t} \frac{\partial w}{\partial u} + \frac{\partial w}{\partial u} \frac{\partial u}{\partial t} - \left(\frac{\partial w}{\partial u} + \frac{\partial w}{\partial t}\right) v + \frac{\partial w}{\partial \Phi} + \frac{\partial w}{\partial \Phi} - \frac{\partial \Phi}{\partial t} = -\varepsilon,
\]

Problem Dimension \(N \rightarrow \infty\)

poly(\(N\))

sample cost

run-time

Problem Dimension \(N\)
Motivation

- Given training data points \((x, y)\), our goal is to learn
  - a certain rule \(f\) that explains the label \(y\) based on features \(x\):

\[
(x: \text{feature}, y: \text{label}) \quad \rightarrow \quad x \rightarrow f(\cdot) \rightarrow y = f(x) + \epsilon
\]
Motivation

- Given training data points \((x, y)\), our goal is to learn
  - a certain rule \(f\) that explains the label \(y\) based on features \(x\):

\[
(x : \text{feature}, y : \text{label}) \rightarrow x \rightarrow f(\cdot) \rightarrow y = f(x) + \epsilon
\]

What if
Motivation

- Given training data points \((x, y)\), our goal is to learn
  - a certain rule \(f\) that explains the label \(y\) based on features \(x\):

\[
(x : \text{feature}, y : \text{label}) \rightarrow f(\cdot) \rightarrow y = f(x) + \varepsilon
\]

What if
- we can actively choose training data
Motivation

- Given training data points \((x, y)\), our goal is to learn
  - a certain rule \(f\) that explains the label \(y\) based on features \(x\):

\[(x : \text{feature}, y : \text{label}) \rightarrow f(\cdot) \rightarrow y = f(x) + \varepsilon\]

- What if we can **actively choose** training data
  - the model has **sublinear d.o.f**
Motivation

- Given training data points \((x, y)\), our goal is to learn
  - a certain rule \(f\) that explains the label \(y\) based on features \(x\):

\[
(x : \text{feature}, y : \text{label}) \quad \xrightarrow{f(\cdot)} \quad y = f(x) + \varepsilon
\]

What if
- we can actively choose training data
- the model has sublinear d.o.f.

Can we achieve fast & robust learning with active sampling + coding theory?
Potential Impact

MRI

Radio Astronomy

Sub-Nyquist Sampling

Machine Learning

Computational Imaging

IoT
Sparse Spectrum (DFT/WHT)

Sparse-graph codes

Sameer Pawar

Xiao (Simon) Li

Orhan Ocal
Given \( f(x) = \sum_{n=0}^{N-1} F_n x^n \)

• Find coefficients \( \{F_n\}_{n=0}^{N-1} \)

Q. How many evaluations do we need?

A. \( N \) evaluations
Recovering the coefficients

- Given \( f(x) = \sum_{n=0}^{N-1} F_n x^n \)
- Find coefficients \( \{F_n\}_{n=0}^{N-1} \)

\[
\begin{bmatrix}
    f(X_0) \\
f(X_1) \\
f(X_2) \\
\vdots \\
f(X_{19})
\end{bmatrix} =
\begin{bmatrix}
    1 & X_0 & \cdots & X_0^{19} \\
    1 & X_1 & \cdots & X_1^{19} \\
    1 & X_2 & \cdots & X_2^{19} \\
    \vdots \\
    1 & X_{19} & \cdots & X_{19}^{19}
\end{bmatrix}
\begin{bmatrix}
    F_0 \\
    F_1 \\
    F_2 \\
    \vdots \\
    F_{19}
\end{bmatrix}
\]

inverse Discrete Fourier Transform (DFT)\n
if \( X_m = e^{\frac{j2\pi}{N} m} \)
What if only $K$ of $N$ coeffs. non-zero?

$K$ sublinear in $N$ ($K/N \rightarrow 0$)

Example:
Degree $N=1$ million
Sparsity $K = 200$

(spoiler alert)
# evaluations = 616
Compute the $K$-sparse DFT of $x \in \mathbb{C}^N$ with $K \ll N$:

$$x[n] = \frac{1}{\sqrt{N}} \sum_{k \in K} X[k] e^{i \frac{2\pi k}{N} n} \quad n = 0, \cdots, N - 1$$

Support $\mathcal{K}$ chosen from $[N]$ uniformly at random
Results: FFAST
(Fast Fourier Aliasing-Based Sparse Transform)

• Noiseless: For $K$ sublinear in $N$
  • Can compute $K$-sparse $N$-length DFT
  • Using **fewer than $4K$** samples in $O(K \log K)$ time

Robust to noise: same framework: $O(K \log^{4/3} N)$ samples in $O(K \log^{7/3} N)$ time

Sub-linear time recovery when d.o.f. sublinear!

Pawar, R, IEEE Trans. Inf. Theory, 2018
Puzzle: Gold thief

- One unknown thief.
- Steals unknown but fixed amount from each coin.
- What is min. no. of weighings needed?
  - 2 are enough.

100 grams each

Differential weight

\[
\begin{bmatrix}
1 & 1 & 1 & 1 & 1 \\
1 & 2 & 3 & 4 & 5
\end{bmatrix}
\begin{bmatrix}
-5 \\
-20
\end{bmatrix}
= \begin{bmatrix}
y_1 \\
y_2
\end{bmatrix}
\]

Ratio-test identifies the location
4-thieves among 12-treasurers

Key Ideas:
1. Randomly group the treasurers.
2. If there is a single thief problem
   ✓ Ratio test
   ✓ Iterate

Questions:
1. How many groups needed?
2. How to form groups?
3. How to identify if a group has a single thief?
Aliasing

Signal and its spectrum

Sampling

Sub-sampling
We use coding-theoretic tools

**Design:**
- Randomized constructions of good sparse-graph codes

**Analysis:**
- Density evolution
- Martingale
- Expander graph theory

**Definitions:**
- **Sub-sampling** below Nyquist rate
- **Chinese-Remainder-Theorem guided subsampling**
- **Sparse graph codes**

**Insights:**
- **Aliasing** in the frequency domain
- **Good** “alias” code

Clever sub-sampling (for **sparse** case)
Main idea

time-domain $x[n]$ length $N = 20$

frequency-domain $X[k]$ sparsity $K = 5$

$\iff$ DFT $\iff$

(length = 20)
Main idea

time-domain $x[n]$ length $N = 20$

frequency-domain $X[k]$ sparsity $K = 5$

$X[1] = 1$
$X[3] = 4$
$X[5] = 1$
$X[10] = 3$
$X[13] = 7$

$\Downarrow 5$

subsample by 5
Main idea

time-domain $x[n]$ length $N = 20$

frequency-domain $X[k]$ sparsity $K = 5$

Our Measurements

$\Downarrow 5$

subsample by 5

$\leftrightarrow \text{DFT} \quad \Longrightarrow$

(length = 20)

$X[3] = 4$

$X[10] = 3$

$X[1] = 1$

$X[5] = 1$

$(length = 4)$

$\leftrightarrow \text{DFT} \quad \Longrightarrow$

$U[0] \quad U[1] \quad U[2] \quad U[3]$
Main idea

time-domain $x[n]$ length $N = 20$

frequency-domain $X[k]$ sparsity $K = 5$

$\Downarrow 5$

subsample by 5

$\iff \text{DFT} \iff$

(length = 20)

Aliasing

$\iff \text{DFT} \iff$

(length = 4)

$U[0] \quad U[1] \quad U[2] \quad U[3]$
Main idea

time-domain $x[n]$  length $N = 20$

frequency-domain $X[k]$  sparsity $K = 5$

$X[3] = 4$
$X[10] = 3$


$\downarrow 5$

subsample by 5

$\Leftrightarrow$ DFT  $(length = 20)$

Aliasing

$\Leftrightarrow$ DFT  $(length = 4)$

Main idea

- Time-domain $x[n]$ length $N = 20$
- Frequency-domain $X[k]$ sparsity $K = 5$

Subsample by 5

Aliasing

$X[3] = 4$
$X[10] = 3$
$X[13] = 7$

$\downarrow 5$

(DFT) $\Leftrightarrow$ (length = 20)

(DFT) $\Leftrightarrow$ (length = 4)

Main idea

time-domain $x[n]$ length $N = 20$

frequency-domain $X[k]$ sparsity $K = 5$

↓5

subsample by 5

Aliasing

$\Leftrightarrow$ DFT $\Rightarrow$
(length = 20)

$\Leftrightarrow$ DFT $\Rightarrow$
(length = 4)
Main idea

- **Time-domain** $x[n]$ length $N = 20$
- **Frequency-domain** $X[k]$ sparsity $K = 5$

\[ X[0] = 1 \]
\[ X[1] = 1 \]
\[ X[3] = 4 \]
\[ X[5] = 1 \]
\[ X[10] = 3 \]
\[ X[13] = 7 \]

$\downarrow 5$

Subsample by 5

\[ U[0] \]
\[ U[1] \]
\[ U[2] \]
\[ U[3] \]

Zero-ton

$\leftrightarrow$ DFT $\leftrightarrow$

(length = 20)

(length = 4)
Main idea

time-domain $x[n]$ length $N = 20$

frequency-domain $X[k]$ sparsity $K = 5$

$\downarrow 5$

subsample by 5

$\leftrightarrow \text{DFT} \leftrightarrow$

(length = 20)

$\leftrightarrow \text{DFT} \leftrightarrow$

(length = 4)

zero-ton multi-ton

Main idea

time-domain $x[n]$ length $N = 20$

frequency-domain $X[k]$ sparsity $K = 5$

↓ 5

subsample by 5

$\Leftrightarrow$ DFT $\Rightarrow$
(length = 20)

$\Leftrightarrow$ DFT $\Rightarrow$
(length = 4)

$X[13] = 7$

zero-ton multi-ton single-ton single-ton
Main idea

time-domain $x[n]$ length $N = 20$

frequency-domain $X[k]$ sparsity $K = 5$


subsample by 5

$\downarrow 5$

$\iff \text{DFT} \iff (\text{length} = 20)$

$\iff \text{DFT} \iff (\text{length} = 4)$

zero-ton multi-ton single-ton single-ton
Main idea

- **Time-domain** $x[n]$ length $N = 20$
- **Frequency-domain** $X[k]$ sparsity $K = 5$

**Shift & subsample by 5**

**Our Measurements**


Subscript $U_S$ suggests **shift**
Main idea

time-domain \( x[n] \) length \( N = 20 \)

frequency-domain \( X[k] \) sparsity \( K = 5 \)

\[
\omega = e^{-j \frac{2\pi}{20}}
\]

\( \Downarrow \) 5

shift & subsample by 5

\[\iff \text{DFT} \] (length = 20)

\[\iff \text{DFT} \] (length = 4)

zero-ton multi-ton single-ton single-ton
Main idea

Stage 1
downsample by 5

$X[13]$

$X[10]$

$X[5]$

$X[3]$

$X[1]$

$U[0].$

$U[1].$

$U[2].$

$U[3].$

DFT

$\downarrow$

$x[n]$
Main idea

Stage 1
downsampling by 5

\[ U[0] \quad U_s[0] \]
\[ U[1] \quad U_s[1] \]
\[ U[2] \quad U_s[2] \]
\[ U[3] \quad U_s[3] \]

Stage 1 DFT \downarrow\quad \text{shift}\quad \downarrow\quad x[n]
Main idea

Stage 1
downsampling by 5

Stage 2
downsampling by 4

X[13]
X[10]
X[5]
X[3]
X[1]

Stage 1
DFT
DFT
shift

Stage 2
Main idea

Stage 1
downsamples by 5

Stage 2
downsamples by 4

Stage $d$
Main idea

Stage 1
downsample by 5

Stage 2
downsample by 4

single-ton
multi-ton
zero-ton
Main idea

Stage 1
donsample by 5

Stage 2
donsample by 4

multi-ton
zero-ton
single-ton

multi-ton
zero-ton
single-ton

multi-ton
zero-ton
single-ton

peeling decoder
Main idea

Stage 1
downsampling by 5

Stage 2
downsampling by 4

$X[13]$

$X[10]$

$X[5]$

$X[3]$

$X[1]$

single-ton

multi-ton

zero-ton

multi-ton

zero-ton

single-ton

single-ton
Main idea

Stage 1
downsample by 5

X[13]

X[10]

X[5]

X[3]

X[1]

Stage 2
downsample by 4

single-ton

multi-ton

zero-ton

single-ton

multi-ton

zero-ton

single-ton

peeling decoder
Main idea

Stage 1
downsampling by 5

$X[13]$

Stage 2
downsampling by 4

$X[10]$

$X[5]$

$X[3]$

$X[1]$

peeling decoder
Main idea

Stage 1
downsampling by 5

\[ X[13] \]
\[ X[10] \]
\[ X[5] \]
\[ X[3] \]
\[ X[1] \]

Stage 2
downsampling by 4

Single-ton

Zero-ton

peeling decoder
Main idea

Stage 1

downsample by 5

$X[13]$

$X[10]$

$X[5]$

$X[3]$

$X[1]$

zero-ton

zero-ton

zero-ton

zero-ton

zero-ton

zero-ton

zero-ton

Sparse DFT Computation
= Decoding over Sparse Graphs

peeling decoder
Sparse DFT Computation = Decoding over Sparse Graphs

- Erasure symbols/packets
  - erasure#1
  - erasure#2
  - erasure#3
  - erasure#4
  - erasure#5

- Parity checks

- Non-zero DFT coefficients
  - non-zero DFT #1
  - non-zero DFT #2
  - non-zero DFT #3
  - non-zero DFT #4
  - non-zero DFT #5

- Aliased frequency bins

- Explicit graph: design well-understood.
- Implicit graph: induced by subsampling.
Main Idea

Stage 1
downsample by 5

Stage 2
downsample by 4

How do we induce good graphs that will work?
**Chinese-Remainder-Theorem**: A number between 0-19 is uniquely represented by its remainders modulo (4,5) > The two graph ensembles are **identical**.
\[ N = 100 \times 103 \times 107; K \approx 200; M \approx 600 \]
$N = 100 \times 103 \times 107; K \approx 200; M \approx 600$

- Subsample by $100 \times 103$
- Shift & Subsample by $100 \times 103$
- Subsample by $100 \times 107$
- Shift & Subsample by $100 \times 107$
- Subsample by $103 \times 107$
- Shift & Subsample by $103 \times 107$

- DFT 107-length
- DFT 107-length
- DFT 103-length
- DFT 103-length
- DFT 100-length
- DFT 100-length

Peeling Decoder
Sparse polynomial learning

What if only (very few) \( K \) of the \( N \) polynomial coeffs. \( \{F_n\} \) are non-zero?

E.g. deg. \( N=1 \) million
Sparsity \( K=200 \)

\[ f(x) = F_{N-1}x^{N-1} + F_{N-2}x^{N-2} + \cdots + F_0 \]

- \( N=100 \times 103 \times 107 \)
- \( K \approx N^{1/3} \)
- \( M=2 \times (100+103+107)-4 \)

# evals. \( M = 616 \)
Noisy Recovery of Sparse DFT
From Noiseless to Noisy

Noiseless - FFAST

Stage 1

DFT \rightarrow \downarrow \rightarrow x[n]

DFT \rightarrow \downarrow \rightarrow \text{shift}

Stage d
From Noiseless to Noisy

Noiseless - FFAST

Stage 1

DFT

DFT

shift

Stage $d$

Noisy - R-FFAST

Stage 1

DFT

DFT

shift

DFT

shift

Stage $d$
Magnetic resonance imaging

Fourier Transform

Inverse Fourier Transform
Numerical Phantoms for Cardiovascular MR

http://www.biomed.ee.ethz.ch/research/bioimaging/cardiac/mrxcat

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

336 = 16 \times 21
323 = 17 \times 19
Temporal difference is sparse

*temporal difference* across different frames of the phantom
Real Time Reconstruction in MATLAB on a Macbook

Measurements: 35.33% of Nyquist rate
Hardware Implementation

**App.:** Detection of “heavy hitters” for IoT/spectrum sensing

A Real-Time, Analog/Digital Co-Designed 1.89-GHz Bandwidth, 175-kHz Resolution Sparse Spectral Analysis RISC-V SoC in 16-nm FinFET

---

**Wang et al,**

*ESSIRC 2018*
Walsh-Hadamard Transform

• N-point Discrete Fourier Transform (DFT)

\[ f[m] = \frac{1}{N} \sum_{k=0}^{N-1} F[k] e^{i \frac{2\pi k m}{N}}, \quad m = 0, \cdots, N - 1 \]

• N-point Walsh-Hadamard (WHT) with \( N = 2^n \)

\[ f[\mathbf{m}] = \sum_{\mathbf{k} \in \{0,1\}^n} F[\mathbf{k}] (-1)^{\langle \mathbf{k}, \mathbf{m} \rangle}, \quad \mathbf{m} \in \{0,1\}^n \]

• \( F[k] \) is sparse in many machine learning applications:
  – Decision tree and regression tree
  – Boolean expression (digital logic)
  – Evolutionary biology
  – Hypergraphs
Walsh-Hadamard Transform

\[ f = (c_0, c_1, c_2, c_3, c_4, c_5, c_6, c_7)^T \]

WHT is equivalent to the high-dimensional DFT over the hyper-cube
Walsh-Hadamard Transform

“time” domain

WH domain
Walsh-Hadamard Transform

“time” domain

WH domain
Walsh-Hadamard Transform

“time” domain

WH domain
Walsh-Hadamard Transform

“time” domain

WH domain
Walsh-Hadamard Transform

- Connection to polynomials? Let \( x_1 = (-1)^{m_1} \) and \( x_2 = (-1)^{m_2} \), then

\[
f(x_1, x_2) = \sum_{k_1, k_2 \in \{0, 1\}} F[k_1, k_2] x_1^{k_1} x_2^{k_2}
\]

\[
f(x_1, x_2) = 3x_1x_2 + 4x_2, \quad x_1, x_2 \in \{-1, 1\}
\]

\[
= 0 \times x_2^0 x_1^0 + 0 \times x_2^0 x_1^1 + 4 \times x_2^1 x_1^0 + 3 \times x_2^1 x_1^1
\]
WHT – Hypergraph Sketching

\begin{equation}
 n = \# \text{ of books} \\
 s = \# \text{ of sale patterns}
\end{equation}

\[ 2^n \approx 10^9 \text{ possible hyperedges if } n = 30 \]

- recover all sale patterns (hyperedges) without logging every transaction?
- sketch the \textbf{cuts} of the graph instead!
\[ n = \# \text{ of books} \]
\[ s = \# \text{ of sale patterns} \]
\[ 2^n \approx 10^9 \text{ possible hyperedges when } n = 30 \]

- recover all sale patterns (hyperedges) without logging every transaction?
- sketch the cuts of the graph instead!

consider a cut:

\[ x_1 = \cdots = x_5 = +1 \]
\[ x_6 = \cdots = x_{25} = -1 \]

\[ \implies \text{cut value } f(x) = 0 \]
WHT – Hypergraph Sketching

\[ n = \# \text{ of books} \]
\[ s = \# \text{ of sale patterns} \]

\[ 2^n \approx 10^9 \] possible hyperedges if \( n = 30 \)

- recover all sale patterns (hyperedges) without logging every transaction?
- sketch the cuts of the graph instead!

consider a cut:

\[ x_1 = \cdots = x_{10} = +1 \]
\[ x_{11} = \cdots = x_{25} = -1 \]

\[ \implies \text{cut value } f(\mathbf{x}) = 1 \]
WHT – Hypergraph Sketching

\[ n = \# \text{ of books} \]
\[ s = \# \text{ of sale patterns} \]

\[ 2^n \approx 10^9 \text{ possible hyperedges if } n = 30 \]

- recover all sale patterns (hyperedges) without logging every transaction?
- sketch the \textbf{cuts} of the graph instead!
• recover all sale patterns (hyperedges) without logging every transaction?
• sketch the cuts of the graph instead!
• Generally speaking, we have the cut function

\[
 f(x) = \frac{3}{2} - \frac{1}{2}x_1x_2 - \frac{1}{2}x_9x_{14} - \frac{1}{2}x_{22}x_{23}
\]

\( n = \# \text{ of books} \)
\( s = \# \text{ of sale patterns} \)

\( 2^n \approx 10^9 \) possible hyperedges if \( n = 30 \)

Frequently Bought Together

- Twilight (The Twilight Saga, Book 1) by Stephenie Meyer Paperback $10.29
- New Moon (The Twilight Saga) by Stephenie Meyer Paperback $10.07
- Eclipse (The Twilight Saga, Book 3) by Stephenie Meyer Paperback $11.39

Price for all three: $31.75
Add all three to cart
Add all three to Wish List

Show availability and shipping details
n = # of books
s = # of sale patterns

\[ 2^n \approx 10^9 \text{ possible hyperedges if } n = 30 \]

- small # of sale patterns \( s \ll n \)
- small # of items per sale \( d \ll n \)

- Generally speaking, we have the \textbf{cut function}

\[ f(x) = \frac{3}{2} - \frac{1}{2}x_1x_2 - \frac{1}{2}x_9x_{14} - \frac{1}{2}x_{22}x_{23} \]
WHT – Hypergraph Sketching

\( n = \# \text{ of books} \)

\( s = \# \text{ of sale patterns} \)

\( 2^n \approx 10^9 \) possible hyperedges if \( n = 30 \)

- Generally speaking, we have the cut function

\[
 f(x) = \frac{3}{2} - \frac{1}{2} x_1 x_2 - \frac{1}{2} x_9 x_{14} - \frac{1}{2} x_{22} x_{23}
\]
WHT – Hypergraph Sketching

- total cut values $2^n = 2^{50}$
- sparsity $K \leq s2^{d-1} = 500$
- # of cut queries $O(Kn) \approx 25000$

$n = 50$ books
$d = 2$ items/sale
$s = 250$ sale patterns
$n = 50$ books
$d = 2$ items/sale
$s = 250$ sale patterns

- total cut values $2^n = 2^{50}$
- sparsity $K \leq s2^{d-1} = 500$
- # of cut queries $O(Kn) \approx 25000$
WHT – Hypergraph Sketching

- $n = 50$ books
- $d = 2$ items/sale
- $s = 250$ sale patterns

- Total cut values $2^n = 2^{50}$
- Sparsity $K \leq s2^{d-1} = 500$
- # of cut queries $O(Kn) \approx 25000$
WHT – Hypergraph Sketching

- Total cut values $2^n = 2^{50}$
- Sparsity $K \leq s2^{d-1} = 500$
- # of cut queries $O(Kn) \approx 25000$

$n = 50$ books
$d = 2$ items/sale
$s = 250$ sale patterns

Hyperedge at iteration 1

Hyperedge at iteration 2
WHT – Hypergraph Sketching

- total cut values \(2^n = 2^{50}\)
- sparsity \(K \leq s2^{d-1} = 500\)
- # of cut queries \(O(Kn) \approx 25000\)

\[n = 50 \text{ books}\]
\[d = 2 \text{ items/sale}\]
\[s = 250 \text{ sale patterns}\]
\[
\begin{align*}
    n &= 50 \text{ books} \\
    d &= 2 \text{ items/sale} \\
    s &= 250 \text{ sale patterns}
\end{align*}
\]

- total cut values \(2^n = 2^{50}\)
- sparsity \(K \leq s^{d-1} = 500\)
- \# of cut queries \(O(Kn) \approx 25000\)
Open source implementations

• Sparse FFT and Sparse WHT implemented in C++

• Publicly available on GitHub
  https://github.com/ucbasics
Broad scope of applications

Sparse-graph codes

Sparse mixed linear regression

Compressive phase retrieval

Sparse Spectrum (DFT/WHT)

Sub-Nyquist sampling theory

Fast neighbor discovery for IoT (group testing)

Compressed sensing

---

Pedarsani, Lee, R., 2014

Lee, Pedarsani, R., 2015

Yin, Pedarsani, Chen, R., 2016

Ocal, Li, R., 2016

Li, Pawar, R., 2014

Pawar, R., 2013

Li, Pawar, R., 2014
Unifying separation architecture for sparse-signal processing

“Peeling-based” turbo engine

Sparse-Graph Code

Divide

Concur

“Solve-if-trivial” sub-engine

Problem-specific sub-code
Sparse-graph codes

Sparse Spectrum (DFT/WHT)

Fast neighbor discovery for IoT (group testing)

Sub-Nyquist sampling theory

Compressive phase retrieval

Sparse mixed linear regression

Compressed sensing

Broad scope of applications

1. Ratio test
   - Pawar, R., 2013
   - Li, Pawar, R., 2014

2. Signature test
   - Lee, Pedarsani, R., 2015

3. Trig. test
   - Pedarsani, Lee, R., 2014

4. Group test
   - Yin, Pedarsani, Chen, R., 2016

5. Ratio test
   - Ocal, Li, R., 2016

6. Ratio test
   - Li, Pawar, R., 2014
Chapter 3

Speeding up distributed computing on the cloud
System Noise

Network bottlenecks

HW failures

Maintenance, etc.
System Noise = Latency Variability

Computing $f(A)$...
Completed in 1s.
System Noise = Latency Variability

Computing $f(A)$…
Completed in 1s.

Computing $f(A)$…
Still computing…
Still…
Completed in 3s.
Straggler Problem: Data Centers

• “The Tail at Scale”, Comm. of the ACM 2013

“The scale and complexity of modern Web services make it infeasible to eliminate all latency variability.”

<table>
<thead>
<tr>
<th></th>
<th>50%ile latency</th>
<th>95%ile latency</th>
<th>99%ile latency</th>
</tr>
</thead>
<tbody>
<tr>
<td>One random leaf finishes</td>
<td>1ms</td>
<td>5ms</td>
<td>10ms</td>
</tr>
<tr>
<td>95% of all leaf requests finish</td>
<td>12ms</td>
<td>32ms</td>
<td>70ms</td>
</tr>
<tr>
<td>100% of all leaf requests finish</td>
<td>40ms</td>
<td>87ms</td>
<td>140ms</td>
</tr>
</tbody>
</table>
Distributed Matrix-Vector Multiplication

\[ A \times b \]
Distributed Matrix-Vector Multiplication

\[ A \times b = \begin{pmatrix} A_1 \\ A_2 \\ A_3 \end{pmatrix} \times b \]
Distributed Matrix-Vector Multiplication

\[ A \times b \]

\[
\begin{pmatrix}
A_1 \\
A_2 \\
A_3
\end{pmatrix} \times b
\]

\[
= \begin{pmatrix}
A_1 \\
A_2 \\
A_3
\end{pmatrix} \times b
\]

\[
= \begin{pmatrix}
A_1 \times b \\
A_2 \times b \\
A_3 \times b
\end{pmatrix}
\]

Master

Worker 1

Worker 2

Worker 3
Distributed Matrix-Vector Multiplication

\[ A \times b \]
\[
\begin{pmatrix}
A_1 \\
A_2 \\
A_3
\end{pmatrix}
\times b
\]
\[
\begin{pmatrix}
A_1 \\
A_2 \\
A_3
\end{pmatrix}
\times b
\]
\[
\begin{pmatrix}
y_1 \\
y_2 \\
y_3
\end{pmatrix}
\]

Master

Worker 1

Worker 2

Worker 3
Distributed Matrix-Vector Multiplication

\[
A \times b = \begin{pmatrix} A_1 \\ A_2 \\ A_3 \end{pmatrix} \times b = \begin{pmatrix} A_1 \times b \\ A_2 \times b \\ A_3 \times b \end{pmatrix}
\]

Q. Can codes provide the distributed algorithms with robustness against stragglers?
Coded Matrix-Vector Multiplication

\[ A \times b \]
\[ = \left( \begin{array}{c} A_1' \\ A_2' \end{array} \right) \times b \]
\[ = \left( \begin{array}{c} A_1' \times b \\ A_2' \times b \end{array} \right) \]
\[ := \left( \begin{array}{c} y_1' \\ y_2' \end{array} \right) \]
\[ A_3' := A_1' + A_2' \]
\[ y_3' := y_1' + y_2' \]
MDS-Coded Matrix-Vector Multiplication

Under exponential latency model

On Amazon AWS
Applications

- Distributed linear or logistic regression
- Distributed non-linear function computation
- Reducing communication in data shuffling by network coding

Has attracted lots of interest:

- **Coded Matrix Multiplication**
- **Distributed Gradient Coding**
- **Approximation:**
  - *SVD + Sequential Coding, Sketching, Second-order methods, Coded Control*
  - ...

Computing on the Cloud with PyWren (AWS Lambda)

Jonas et al., “Occupy the Cloud, distributed computing for the 99%” (2017)

Recent interest in "microservices" has led to Amazon’s AWS Lambda, where each lambda task gets:
- a single core (exact performance varies a bit)
- 1536 MB of RAM max and 300s max execution time & a runtime of python or Java

Your laptop

Write a function
Upload the function and data to S3
Invoke Lambda machines

The cloud

Poll S3 for results
Stragglers out of 6000

Lambda start

Setup done, job start

Job done

Results returned

AWS Lambda

~400 Stragglers out of 6000

PyWren: Can have up to 16,000 workers on AWS Lambda
Scalable Coded Computing

• **Latency** = **Enc. Time + Computation Time + Dec. Time** (plus comm. cost to move data in distributed setting)

• MDS codes for matrix-vector (Lee et al. ‘15) or matrix-matrix (poly. codes: Yu et al. ‘17) target only the **computation time**

• At scale? (e.g., Lambda w/6K workers & 400+ stragglers)

• Decoding “on the fly”: **real-valued** computations

**Product Codes:** a good tradeoff bet. near-MDS and local enc./dec.
Conclusions

• Codes for **speed**: offer a **CLEAR** advantage – **Computation**, **LEarning**, **Access**, **Recovery**.

• **Access**: Regeneration codes and beyond

• **LEarning & Recovery**: Sparse-graph codes based “peeling” core with “verification” test

• **Computation**: Straggler-proofing large-scale distributed computing
Thank You!

QUESTIONS?