15-750: Algorithms in the Real World

Algorithms for coding
(Error Correcting Codes)

Announcement:
Midterm exam on March 1.
More details in the Piazza post.
What do these sentences say?

Why did this work?

Redundancy!

Codes are clever ways of judiciously adding redundancy to enable recovery under “noise”.


**General Model**

“Noise” introduced by the channel:

- changed fields in the codeword vector (e.g. a flipped bit).
  - Called **errors**

- missing fields in the codeword vector (e.g. a lost byte).
  - Called **erasures**

How the decoder deals with errors and/or erasures?

- **detection** (only needed for errors)
- **correction**
Applications

Numerous applications:
Some examples

- **Storage**: Hard disks, cloud storage, NAND flash…
- **Wireless**: Cell phones, wireless links,
- **Satellite and Space**: TV, Mars rover, …

Reed-Solomon codes are by far the most used in practice.

Low density parity check codes (LDPC) codes used for 4G (and 5G) communication and NAND flash
Block Codes

symbols (e.g., bits)

message 1

message 2

block 1

block 2

Other kind: convolutional codes (we won’t cover it)...
Block Codes

- Each message and codeword is of fixed size
- Notation:
  
  $k = |m|$  
  length of the message  
  
  $n = |c|$  
  length of the codeword  

$C = \text{“code”} = \text{set of codewords}$
Simple Examples

3-Repetition code: $k=1, n=3$

<table>
<thead>
<tr>
<th>Message</th>
<th>Codeword</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>000</td>
</tr>
<tr>
<td>1</td>
<td>111</td>
</tr>
</tbody>
</table>

- How many erasures can be recovered?
- How many errors can be detected?
- Up to how many errors can be corrected?

Errors are much harder to deal with than erasures.

Why?

Need to find out where the errors are!
Simple Examples

Single parity check code: k=2, n=3

<table>
<thead>
<tr>
<th>Message</th>
<th>Codeword</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>000</td>
</tr>
<tr>
<td>01</td>
<td>011</td>
</tr>
<tr>
<td>10</td>
<td>101</td>
</tr>
<tr>
<td>11</td>
<td>110</td>
</tr>
</tbody>
</table>

Consider codewords as vertices on a hypercube.

- codeword
- \( n = 3 \) (hypercube dimensionality)
- \( 2^n = 8 \) (number of nodes)
Simple Examples

Single parity check code: $k=2$, $n=3$

- How many **erasures** can be recovered?
- How many **errors** can be **detected**?
- Up to how many **errors** can be **corrected**?
**Systematic codes**

**Definition:** A **Systematic code** is one in which the message symbols appear in the codeword in uncoded form.

<table>
<thead>
<tr>
<th>message</th>
<th>codeword</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>000000</td>
</tr>
<tr>
<td>001</td>
<td>001011</td>
</tr>
<tr>
<td>010</td>
<td>010101</td>
</tr>
<tr>
<td>011</td>
<td>011110</td>
</tr>
<tr>
<td>100</td>
<td>100110</td>
</tr>
<tr>
<td>101</td>
<td>101101</td>
</tr>
<tr>
<td>110</td>
<td>110011</td>
</tr>
<tr>
<td>111</td>
<td>111000</td>
</tr>
</tbody>
</table>
Large-scale distributed storage systems

1000s of interconnected servers
100s of petabytes of data

- Commodity components
- Software issues, power failures, maintenance shutdowns
Large-scale distributed storage systems

1000s of interconnected servers

Unavailabilities are the norm rather than the exception

- Commodity components
- Software issues, power failures, maintenance shutdowns
Facebook analytics cluster in production: unavailability statistics

- Multiple thousands of servers
- Unavailability event: server unresponsive for > 15 min

![Graph showing unavailability statistics](image)

#unavailability events

[Rashmi, Shah, Gu, Kuang, Borthakur, Ramchandran, USENIX HotStorage 2013 and ACM SIGCOMM 2014]
Facebook analytics cluster in production: unavailability statistics

- Multiple thousands of servers
- Unavailability event: server unresponsive for > 15 min

Daily server unavailability = 0.5 - 1%

[Rashmi, Shah, Gu, Kuang, Borthakur, Ramchandran, USENIX HotStorage 2013 and ACM SIGCOMM 2014]
Data needs to be stored in a redundant fashion

Servers unavailable

Data inaccessible

Applications cannot wait, Data cannot be lost
Traditional approach: Replication

- Storing **multiple copies** of data: Typically 3x-replication

  “blocks”

  ```
  a b c d
  ```

  3 replicas

  - a b c d
  - a b c d
  - a b c d

  distributed on servers across network
Traditional approach: Replication

• Storing **multiple copies** of data: Typically 3x-replication

```
|  a | b | c | d |
```

“blocks”

```
|  a | b | c | d |
|  a | b | c | d |
|  a | b | c | d |
```

3 replicas

distributed on servers across network
Traditional approach: Replication

- Storing **multiple copies** of data: Typically 3x-replication

Too expensive for large-scale data

Better alternative: **codes**!
Two data blocks to be stored: a and b

Tolerate any 2 failures

3-replication

Storage overhead = 3x

Erasure code

Storage overhead = 2x

"parity blocks"
Two data blocks to be stored: \(a\) and \(b\)

Tolerate any 2 failures

Much less storage for desired fault tolerance

3-replication

Storage overhead = 3x

Erasure code

Storage overhead = 2x

"parity blocks"
Erasure codes: how are they used in distributed storage systems?

Example: 
[n=14, k=10]

10 data blocks

4 parity blocks

distributed to servers
Almost all large-scale storage systems today employ erasure codes

Facebook, Google, Amazon, Microsoft...

“Considering trends in data growth & datacenter hardware, we foresee HDFS erasure coding being an important feature in years to come”

- Cloudera Engineering (September, 2016)
Simple Examples

Single parity check code: k=2, n=3

• How many erasures can be recovered?
• How many errors can be detected?
• Up to how many errors can be corrected?

Erasure correction = 1, error detection = 1, error correction = 0

Cannot even correct single error. Why?
Codewords are too “close by”

Let’s formalize this notion of distance..
Block Codes

Notion of distance between codewords: **Hamming distance**

\[ \Delta(x,y) = \text{number of positions s.t. } x_i \neq y_i \]

Minimum distance of a code

\[ d = \min\{\Delta(x,y) : x,y \in C, x \neq y\} \]

Code described as: \((n, k, d)_q\) \[ \begin{aligned} \Sigma &= \text{alphabet} \\
q &= |\Sigma| = \text{alphabet size} \\
C &\subseteq \Sigma^n \text{ (codewords)} \end{aligned} \]

Question:

What alphabet did we use so far?
Error Correcting One Bit Messages

How many bits do we need to correct a one bit error on a one bit message?

2 bits
0 -> 00, 1 -> 11
(n=2, k=1, d=2)

3 bits
0 -> 000, 1 -> 111
(n=3, k=1, d=3)

In general need $d \geq 3$ to correct one error. Why?
Role of Minimum Distance

Theorem:
A code $C$ with minimum distance “$d$” can:
1. detect any $(d-1)$ errors
2. recover any $(d-1)$ erasures
3. correct any $\left\lfloor \frac{d-1}{2} \right\rfloor$ errors

Intuition: <board>

Stated another way:
For $s$-bit error detection $d \geq s + 1$
For $s$-bit error correction $d \geq 2s + 1$
To correct $a$ erasures and $b$ errors if $d \geq a + 2b + 1$