IN THE CLOUD?

CODES

Emina @ Bell Labs
Codes at all Levels and for all Layers

Files have chunks.

Chunks have packets.

Packets have symbols.

Codes we can’t refuse.
The moon is more useful than the sun!

Why?

We need the light more during the night than during the day.
Great codes are not born great, they grow great?

CLOUDS, FAILURES, CROWDS

MULTICAST

FADING

... all users.

... all SNRs.
An IR-HARQ Example

**CLAUDE SHANNON** **THE FATHER OF INFORMATION THEORY**

at the transmitter

transmission # 1

transmission # 2

transmission # 3

transmission # 4

at the receiver

**CLAUDE SHANNON** **THE FATHER OF INFORMATION THEORY**
IR-HARQ in Practice

- Considered one of the most important features of cdma2000
  - on the forward link in Release 0 and
  - on both the reverse and forward links in Revision A
- The mother code is the rate 1/5 turbo code of IS-95.
- Decoding is
  - attempted if the total received power is deemed sufficient,
  - done by performing only a few iterations of message passing.
- There are
  - prescribed puncturing patterns and error prediction methods,
  - either predetermined or no transparent transmission rules.
Multicast is often the most cost effective, e.g., on stadiums. Verizon demoed LTE Multicast over the 2014 Super Bowl week.
Fountain (Rateless) Codes

Claim to Fame
Enable reliable communications over multiple, unknown channels:

1. They are rateless ⇒ their redundancy can be flexibly adapted to changing channel/network conditions of e.g. as in wireless.

2. In the case of multiple erasure channels as in Multimedia Broadcast/Multicast Service (MBMS), they can be made to universally achieve the channel capacity for all erasure rates.

Fountain codes are simple to encode and decode.

Isn’t Hybrid ARQ Enough?
The Importance of Being Earnest

by Oscar Wilde

a trivial comedy for serious people
What is Ratelessness and is it Overrated?

Ratelessness matters and means different things to different people:
- encoder can produce potentially infinite stream of symbols
- each code symbol is statistically identical

Can fixed rate codes be made rateless?

\[
\text{BEC capacity: } \frac{R}{1 - \epsilon^*} \\
\text{LDPC (rate } R, \text{ BP threshold } \epsilon^*): \frac{R}{1 - \epsilon^*} \\
\text{LT (length } 10^4, \text{ BP): } \frac{R}{2}\sqrt{\frac{\epsilon^*}{2}}
\]
The (Un)Bearable Ratelessness of Coding

Two eMBMS Phases: Multicast Delivery & Unicast Repair:
What are Raptor Codes?

for the conception, development, and analysis of *practical* rateless codes
Goal:
Download a content chunk of \( k_p \) packets with \( k_s \) symbols per packet.

Inner and Outer Codes & Channels:

- At the inner level, there is a (possibly rateless) \((n_s, k_s)\) code. Symbols are sent through a channel with erasure probability \( \epsilon_s \).

- At the outer level, there is a (possibly rateless) \((n_p, k_p)\) code. What channel do the packets go through?

⚠️ To guarantee the file download, one code has to be rateless. If the inner code is rateless, there is no need for any outer coding.

Feedback is instantaneous, noiseless, and possible after each symbol.
Infinite Incremental Redundancy (IIR)

- The inner (physical layer) code is rateless
  ⇔ coded symbols are transmitted until \( k_s \) of them are received.
  ⇒ #transmissions/packet is \( (k_s, 1 - \epsilon_s) \) negative Binomial (NB).

- The outer (packet level) code is not needed (no dropped packets).

- #transmissions/chunk \( T^{\text{IIR}} \) is the sum of \( k_p \) i.i.d. \( \text{NB}(k_s, 1 - \epsilon_s) \)
  ⇒ also a negative Binomial with parameters \( (k_p \cdot k_s, 1 - \epsilon_s) \)

\[
\mathbb{E}[T^{\text{IIR}}] = \frac{k_s \cdot k_p}{1 - \epsilon_s}.
\]
Finite Redundancy (FR)

- A fixed-rate code \((n_s, k_s)\) is used at the physical layer (per packet).
  - Each packet is erased independently with probability
    \[
    \epsilon_p = \sum_{j=n_s-k_s+1}^{n_s} \binom{n_s}{j} \epsilon_s^j (1-\epsilon_s)^{n_s-j}.
    \]
- The outer (packet level) code is rateless
  - \(\Leftrightarrow\) coded packets are transmitted until \(k_p\) of them are received.
  - \(\Rightarrow\) #packet-transmissions/chunk is \(\text{NB}(k_p, 1-\epsilon_p)\).
- #symbol-transmissions/chunk \(T^{\text{FR}}\) is not NB, but still,
  \[
  \mathbb{E}[T^{\text{FR}}] = \frac{k_p}{1-\epsilon_p} \cdot n_s.
  \]

How do we compare
\[
\mathbb{E}[T^{\text{IIR}}] = \frac{k_s \cdot k_p}{1-\epsilon_s} \quad \text{vs.} \quad \mathbb{E}[T^{\text{FR}}] = \frac{n_s \cdot k_p}{1-\epsilon_p} \quad (n_s \geq k_s \land \epsilon_s \geq \epsilon_p)
\]
\[ \mathbb{E}[T^{\text{IIR}}] \leq \mathbb{E}[T^{\text{FR}}] \]

Proof:
Let \( S \) be the number of received symbols after \( n_s \) transmission.
\( \Rightarrow \) \( S \) is Binomial with expectation is \( n_s \cdot (1 - \epsilon_s) \).
\( \Rightarrow \)

\[ 1 - \epsilon_p = P(S \geq k_s) \leq \frac{n_s \cdot (1 - \epsilon_s)}{k_s}, \]

by the definition of \( \epsilon_p \) and Markov’s inequality.

Ratless coding at the inner rather than at the outer layer results in fewer channel uses on average for chunk download.
Multicast Scenarios

Point-to-Point Like:

Infinite Incremental Redundancy (IIR):

- transmit **coded symbols** until \( k_s \) are received **by all users**.
  \[ \Rightarrow \] no coding or re-transmission at the packet level is needed.

Fixed Redundancy (FR):

- a fixed a rate code \((n_s, k_s)\) is used at the physical layer.
  \[ \Rightarrow \] coding or retransmission at the packet level is required.
- transmit **coded packets** until \( k_p \) are received **by all users**.

⚠️ Rateless coding may sometimes be used at both layers, e.g.,

- transmit **coded symbols** until \( k_s \) are received **by some \( \ell \) users**, and
- transmit **coded packets** until \( k_p \) are received **by all users**.
IIR vs. FR With $u$ Users (normalized average download time)

$k_p = 100, k_s = 100, \epsilon_s \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$.
IIR vs. FR in Multicast (normalized average download time)

\[ k_p = 100, \quad k_s = 1000, \quad \epsilon_s \in \{0.1, 0.2, 0.3, 0.4, 0.5\}. \]
Codes for the Cloudy Season

We want our content to be **reliably stored** & **quickly accessible**, and that requires energy for

- storage
- transmission
- computation

Coding minimizes resources needed to maintain the guaranteed reliability. **But what does it do to accessibility?**
Getting Content from the Cloud(s)
(n, k) Multiple Broadcasts Content Access Model

- Users request the same content (file F), stored in the cloud.
- Server \( s \), \( s = 1, \ldots, n \),
  1. acquires \( F \) from the cloud at the time \( W_s \)
     \((W_s \text{ are iid RVs, exponentially distributed with the mean } W)\)
  2. delivers \( F \) by broadcast to the users in time \( D_s \)
     \((D_s \text{ are iid RVs, highly concentrated around the mean } D)\)
\( \Rightarrow \) File F download time from server \( s \) is \( W_s + D_s \).
- \( F \) is split into \( k \) blocks and encoded into \( n \) blocks s.t.
  any \( k \) out of \( n \) blocks are sufficient for content reconstruction.
- Each user’s request is sent to all \( n \) servers.

When do \( k \) out of \( n \) servers deliver their \( F/k \)-size blocks?
Order Statistics

- For iid RVs \(\{X_1, X_2, \cdots, X_n\}\), the \(k^{th}\) smallest among them is an RV, known as the \(k^{th}\) order statistics \(X_{k,n}\).

- When \(X_i\)'s are \(\exp(W)\), the mean and variance of \(X_{k,n}\) are

\[
E[X_{k,n}] = W(H_n - H_{n-k}) \quad \text{and} \quad V[X_{k,n}] = W^2(H_{n^2} - H_{(n-k)^2}),
\]

where \(H_n\) and \(H_{n^2}\) are (generalized) harmonic numbers

\[
H_n = \sum_{j=1}^{n} \frac{1}{j} \quad \text{and} \quad H_{n^2} = \sum_{j=1}^{n} \frac{1}{j^2}.
\]

When \(k\) decreases, for fixed \(n\)

\(E[X_{k,n}]\) decreases; good for diversity but what about parallelism?
Theorem:

1. The mean download completion time is given by

\[ T_{n,k} = WN_n - H_{n-k} + \frac{D}{k}, \]

where \( H_n = 1 + \frac{1}{2} + \cdots + \frac{1}{n}. \)

2. \( T_{n,k} \) depends on \( W \) vs. \( D \), and is minimized when

\[ k^* \approx \frac{-D + \sqrt{D^2 + 4nWD}}{2W}. \]
What is Really the Optimal $k$?

![Graph showing expected response time vs. $k$ for two different scenarios: $W=2, D=1$ and $W=1, D=2$.]

VS.
You know a lot of geniuses, y’know.

You should meet some stupid people once in a while, y’know.  
**You could learn something!**
Queueing for Content
Two Queueing Models

Single M/M/1 Queue:

- Requests arrive at rate $\lambda$ according to a Poisson process.
- Job service times have an exponential distribution with rate $\mu$.
- Many metrics of interest are well understood for this model, e.g. the response time is exponential with rate $\mu - \lambda$.

Fork-Join FHW Queue:

- Jobs are split on arrival and must be joined before departure.
- In the FHW (Flatto, Hahn, Wright) model, at each queue,
  - Requests arrive at rate $\lambda$ according to a Poisson process.
  - Job service times have an exponential distribution with rate $\mu$.
- It is seen as a key model for parallel/distributed systems, e.g., RAID.
- There is a renewed interest in the problem (e.g., map-reduce).
- Few analytical results exist, but various approximations are known.
The \((n, k)\) Fork-Join System

**Architecture:**

- \(F\) is split into \(k\) blocks and encoded into \(n\) blocks s.t. any \(k\) out of \(n\) blocks are sufficient for content reconstruction.
- The \(n\) coded blocks are stored on \(n\) disks.

**Operation:**

- User request for \(F\) are forked to all \(n\) disks.
- Downloads from any \(k\) disks jointly enables reconstruction of \(F\).

\[\Rightarrow \text{Arrival rate at each of the } n \text{ queues is } \lambda \text{ and service rate is } k\mu.\]
(3, 2) Fork-Join System Architecture

- Content $F$ is split into equal parts $a$ and $b$, and stored on 3 disks as $a$, $b$, and $a + b \Rightarrow$ each disk stores half the size of $F$.
- User request for $F$ are forked to all 3 disks.
- Downloads from any 2 disks jointly enables reconstruction of $F$.

Storage is 50% higher, but download time (per disk & overall) is reduced.
(3, 2) Fork-Join System Operation
Stability of \((n, k)\) Fork-Join FHW System

The rate of arrivals \(\lambda\) and the service rate \(k\mu\) per node must satisfy

\[
\lambda - \frac{\lambda(n - k)}{n} < k\mu
\]

\(\Rightarrow \lambda < n\mu\)
The \((n, 1)\) Fork-Join FHW System

- The effective arrival rate per node is \(\lambda/n\).
- The system behaves as \(n\) independent \(M/M/1\) queues.
  \(\Rightarrow\) the node response time is \(\exp(1/(\mu - \lambda/n))\)

\(\Rightarrow\) the system response time is \(\exp(1/(n\mu - \lambda))\).
Response Time Histogram for $10^4$ Downloads

$(10, k)$ Fork-Join Queue, FHW Model, $\lambda = 1$, $\mu = 3$
Storage Space vs. Download Time in (10, k) Systems

\[ M/M/1 \]
request rate \( \lambda = 1 \)
\( \mu = 3 \) per unit-download

\[ \text{response time} \]
\[ \text{fraction of completed downloads} \]

- single disk
- \( k=1 \)
- \( k=5 \)
- \( k=10 \)

\[ \text{single disk baseline – unit storage} \]
\[ \text{the same total storage} \]
\[ \text{double total storage} \]
\[ 10 \times \text{increase in storage} \]
Doubling Storage Space Shortens Download Time

![Graph showing the impact of doubling storage space on download time and fraction of completed downloads for different values of k.](image)
FHW \((n, k)\) Fork-Join System

- \(k = n\) means there is no redundancy \(\Rightarrow\) fork-join FHW queue.
- \(k = 1\) means replication \(\Rightarrow\) \(n\) independent M/M/1 queues.
- \(1 < k < n\) means coding \(\Rightarrow\)
  1. there is no independence and the system is not memoryless \(\Rightarrow\) hard to derive analytical results,
  2. but there is enough independence to benefit from diversity.

We are interested in the mean response time \(T_{n,k}\).

Previous work has attempted finding \(T_{n,n}\), but only bounds are known.
Upper Bound on Response Time $T_{n,k}$

Consider a modified $(n, k)$ fork-join system in which a completed task does not exit its queue until $k$ tasks of the same job are completed. (cf. split-merge system)

The $(n, k)$ split-merge system

- has response time greater than its fork-join counterpart, and
- is equivalent to an $M/G/1$ queue with service time $S_k$, the $k^{th}$ order statistics of $\exp(k\mu)$, with the mean and variance

$$E[S_k] = \frac{H_n - H_{n-k}}{k\mu}, \quad V[S_k] = \frac{H_n^2 - H_{(n-k)^2}}{k\mu^2}.$$ 

⇒ An upper bound on $T_{n,k}$ is given by the Pollaczek-Khinchin formula:

$$T_{n,k} \leq E[S_k] + \frac{\lambda (V[S_k] + E[S_k]^2)}{2(1 - \lambda E[S_k])}$$
Lower Bound on Response Time $T_{n,k}$

Stages of Job Processing:

(Varki et al. approach)

- A job goes through $k$ stages of processing, one for each task.
- At stage $j$, $0 \leq j \leq k - 1$, the job has completed $j$ tasks.
- The service rate of a job in stage $j$ stage is at most $(n - j)k\mu$.

\[
T_{n,k} \geq \sum_{j=0}^{k-1} \frac{1}{(n - j)k\mu - \lambda} \quad \leftarrow \text{sum of response times of } k \text{ stages}
\]

\[
= \frac{1}{k\mu} \left[ H_n - H_{n-k} + \rho \cdot (H_{n-(n-\rho)} - H_{(n-k)(n-k-\rho)}) \right] \quad (\rho = \frac{\lambda}{\mu})
\]
Tightness of the Bounds

(10, k) Fork-Join Queue, FHW Model, $\lambda = 1$

For $\mu = 3$

For $\mu = 1.001$
Diversity – the Power of Choosing All

Average time to download one unit of content ($1/\mu$)

Mean Response Time

(10, 1) fork-join system
(20, 2) fork-join system
Power-of-2
Power-of-10 (LWL job assignment)
When is this Bound Valid, Tight, Applicable?

STOP!
REALITY CHECK!
What is a Good Model for Service Time?
$\alpha$-Job-Dependent Service Time

Mean response time

$\alpha = 1$

$\alpha = 1/2$

$\alpha = 0$
Heavy Tailed Service Time

Mean response time

- Pareto, $\alpha = \infty$
- Exponential
- Pareto, $\alpha = 1.8$
- Pareto, $\alpha = 4$
What is "coding gain" beyond the physical layer?

BER vs. SNR

SNR necessary for BER of $10^{-12}$ vs. code rate for RS(255,k)
A Coding Tale of a Tail at Scale

Splitting jobs into smaller tasks allows parallel task execution, but increases randomness in the system, **hence the tail.**

Coding cuts the tail.

Which jobs permit cutting the tail?
What (and Who) Should be Noted Down?

Hybrid ARQ may not be enough.
* w. M. Heindelmaier, TUM.

Ratelessness may be unbearable.
*** w. I. Andriyanova, ENSEA, & S. Kokalj, ONR.
***** EPFL, CRNS and NSF

Redundancy may reduce delay.
**** w. G. Joshi, MIT, & Y. Liu, U-W Madison.
****** Bell Labs and DIMACS
Thomas, I chose the right man for chancellor!

I should in fairness add that my taste in music is reputedly deplorable.

Your taste in music is excellent. It exactly coincides with my own!
What is the Future for Coding Theorists?