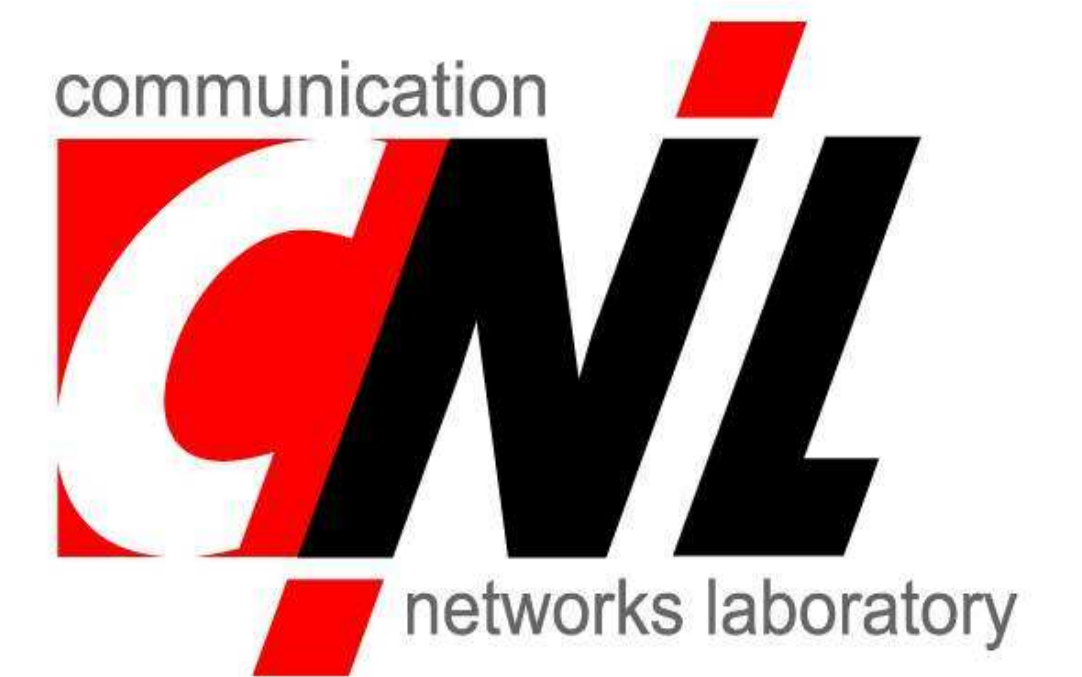


Reliable Data Transmitting Without Feedback

Dömötör Pálvölgyi, PhD student
Eötvös Loránd University, supervisor: Zoltán Király



Scenarios

Data sending possibilities and demands may vary greatly, we distinguish the following cases.

- **Number of Receivers:**
One-to-one/Simple (One person to another one)
One-to-many/Share/Multicast/Broadcast (One person to many other sending same file)
- **Network Topology:**
Connection can be Symmetric (Two-way)
Asymmetric (Ad hoc networks, where sending in different directions can be different)
One-way (Receiver cannot reply)
- **Data Type:**
One file (Static data available on harddrive)
Stream (Data arrives continuously and must be sent ASAP)
- **Chance of Packetloss:**
Known (Approximately some fixed p probability)
Unknown (May vary completely)
- **Decoding:**
Continuous (Decoding starts immediately after first few packets arrive)
After Transmission is over (Decoding starts only after all/almost all packets arrive)

Possible Protocols

Here we compare the known protocols to our *Streaming Protocol*.

TCP/IP	Fountain Codes (like LT code, Raptor)	Streaming Protocol
needs Two-way connection works only in One-to-one case can be used to Stream Packetloss can be arbitrary Continuous	One-way One-to-many not good for Streaming Packetloss can be arbitrary After Transmission	One-way One-to-many good for Streaming bound for Packetloss needed Continuous

Basic Fountain Protocol

This is a very simple example when $N = 5$ Units are sent. Each row of the matrix below to the left represents a Message, the numbers show which elements are contained in the random \mathbb{S} subset.

1 1 1 1 0	$\xrightarrow{\text{Matrix received}}$	1 1 1 1 0	$\xrightarrow{\text{Matrix after Gaussian elimination}}$	1 0 0 0 0
0 1 0 0 0		0 1 0 0 0		0 1 0 0 0
0 1 0 1 0		0 1 0 0 0		0 0 1 0 0
0 1 0 0 0		1 0 1 1 1		0 0 0 1 0
1 0 1 1 1		0 1 1 0 0		0 0 0 0 1
0 1 0 0 1		0 1 1 0 0		
0 1 1 0 0		0 0 1 1 0		
0 0 1 1 0				

So in this special case 8 Messages were sent, 2 were lost, and the remaining 6 were enough to recover all 5 Units.

Description of Basic Fountain Protocol

- Cut data into smaller Units (eg. 1kB each)
- Assume that there are N Units to send
- For every Message, generate a random \mathbb{S} subset of the N Units
- This \mathbb{S} is included in the header of the Message (requires N bits, this can be a significant overhead but can be much smaller if LT code is used)
- The i th bit of the data part of the Message is the bitwise sum of the i th bit of the Units that are in \mathbb{S}
- After N such Messages arrive for which the characteristic vectors of the \mathbb{S} subsets have full rank, data can be decoded using Gaussian elimination (this might be slow but can be linear if Raptor code is used)
- If any $N + \log 1/\delta$ Messages are received, the original data can be decoded with probability $\geq 1 - \delta$
- If chance of Packetloss is p , then $\sim \frac{N}{1-p}$ Messages are needed on average to decode, which is optimal

Suggested Streaming Protocol

- Cut incoming data into Units and assign numbers to each Unit (numbers from 0 through 65535 can be used periodically)
- Depending on the p chance of Packetloss, fix two parameters, L (how much packets are mixed in one Message) and $r > 1$ (redundancy rate)
- In the i th step of the protocol $R := \lfloor \frac{i}{r} \rfloor \bmod 65536$, Unit with number R is called Reference Unit
- Choose a random \mathbb{S} subset of the L next Units after the Reference Unit
- This \mathbb{S} along with R is included in the header of the Message (if $L = 80$, then the whole header of the Message is of the same size as a TCP packet header)
- The i th bit of the data part of the Message is the bitwise sum of the i th bit of the Units that are in \mathbb{S} and the i th bit of the Reference Unit
- After the Reference Unit number is bigger than Q , we can decode the Q th Unit using Gaussian elimination with probability at least $\sim 1 - ((\frac{1}{2})^{(1-p)r-1})^L$
- For sending N Units of data, the chance of failing to recover *all* of them is $\leq N((\frac{1}{2})^{(1-p)r-1})^L$
- If $r = \frac{1.27}{1-p}$, $L = 80$, $N = 2^{16}$ (meaning 64MB data) then this probability is $\leq \frac{1}{1000000}$

Message Header

The structure of a Message sent during the Streaming Protocol with $L = 80$.

Bits	0-15	16-31
0	Source	Destination
32	Length	Checksum
64	Ref Unit #	↓
80	Characteristic vector of \mathbb{S}	
160	Data	

Example for Streaming Protocol

This is a simple example when we want to decode the 5th Unit (column) assuming that the previous Units have already been decoded, $L = 3$, $r = 1.5$. Each row of the matrix below to the left represents a Message, the numbers show which elements are contained in the random \mathbb{S} subset. Yellow indicates the Units that might be in the message (depending on \mathbb{S}), blue indicates the lost Messages.

0 1 0 1 0 0 0 0	\rightarrow	1 0 0 0 0 0 0 0	\rightarrow	1 0 0 0 0 0 0 0
0 0 1 0 1 1 0 0		0 1 0 0 0 0 0 0		0 1 0 0 0 0 0 0
0 0 1 1 0 1 0 0		0 0 1 0 0 0 0 0		0 0 1 0 0 0 0 0
0 0 0 1 0 1 1 0		0 0 0 1 0 0 0 0		0 0 0 1 0 0 0 0
0 0 0 0 1 1 0 1		0 0 1 0 1 1 0 0		0 0 0 0 1 0 0 0
0 0 0 0 1 1 0 1		0 0 0 1 0 1 1 0		0 0 0 0 0 1 0 0
0 0 0 0 1 0 0 1		0 0 0 0 1 1 0 1		0 0 0 0 0 0 1 0
		0 0 0 0 1 0 0 1		0 0 0 0 0 0 0 1

Since the previous values are supposed to be known, we can attach their identity matrix to the received Messages and then perform Gaussian elimination. Note that only a matrix of sidelength $2L \times 2L$ must be inverted, this makes the process faster. So in this special case 6 Messages were sent, 2 were lost, and the remaining 4 were enough to recover the 5. Unit. Note that the first 3 of the 4 messages would not have been sufficient to recover the 5. Unit.

Further Improvements

- Whenever the Reference Unit is changed, the data of the next Message can be only the new Reference Unit
This saves L bits from the header for these Messages
It makes decoding easier and safer
- If users might start downloading the Stream any time
Then r should be bigger *or*
Sometimes „catch-up” Messages should be sent
These are also useful if some users might temporarily lose their connection
- The header can be reduced if \mathbb{S} is chosen from the d element subsets of L
This way the header is reduced by $L - d \log L$ bits
Chance of successful transmitting might decrease
- In One-to-one, Asymmetric cases, Receiver can send a warning when lost
This makes the correction easy, so erring a few times is not that bad