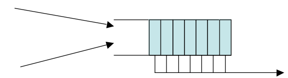
Connections between network coding and stochastic network theory

Bruce Hajek

Abstract: Randomly generated coded information blocks form the basis of novel coding techniques used in communication networks. The most studied case involves linear coding, in which coded blocks are generated as linear combinations of data blocks, with randomly chosen coefficients. Digital fountain codes, including Luby's LT codes, and Shokrollahi 's Raptor codes, involve coding at the source, while network coding involves coding within the network. Recently Maneva and Shokrollahi found a connection between the analysis of digital fountain codes, and fluid and diffusion limit methods, such as in the work of Darling and Norris. A simplified description of the connection is presented, with implications for code design. (Background reading can be found at http://courses.ece.uiuc.edu/ece559/spring06BH/)

Orientation

On Thursday, Ralf Koetter discussed network coding: coding within the network



This talk will be limited to the topic of coding at source nodes, which is much further towards real world use. (See www.digitalfountain.com)

Stochastic network theory can be helpful in learning how to apply such codes in networks (e.g. tradeoff between buffer size and erasure protection) and in designing such codes. This talk focuses on the later.

Outline

- Multicast, and linear codes for erasures
- Luby's LT codes
- Markov performance analysis for Poisson model
- Fluid approximation and an application
- Diffusion approximation and an application

Multicast with lost packets

source message: k symbols: (fixed length binary strings) S1 S2 Sk

Symbols are repeatedly broadcast in random order to many receivers, but can be lost.
Each receiver tries to collect a complete set, for example:

S1 ___ ___ S4 S5 ___ ___ Sk-2 ___ Sk

$$P[\text{collection complete}|\text{receive } k\log k + kc \text{ symbols}]$$
$$\approx P[\text{collection complete}|\text{receive } Poi(k\log k + kc) \text{ symbols}]$$
$$= (1 - \tfrac{e^{-c}}{k})^k \approx \exp(-e^{-c})$$

Multicast with coding at source, and lost packets

Source message: k symbols: S1 S2 Sk
(fixed length binary strings)

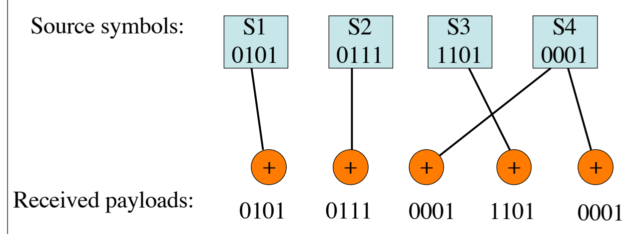
Source forms m coded symbols: S1 S2 Sm

Symbols are repeatedly broadcast in random order
to many receivers, but can be lost.
Each receiver tries to collect enough distinct coded packets:

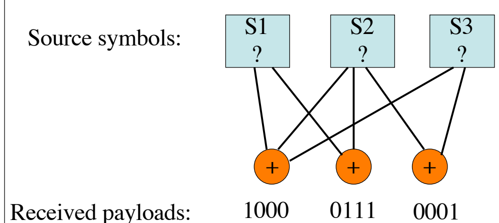
S1 ___ ___ S4 S5 ___ ___ Sm-2 ___ Sm

For a good code, only k or a few more coded packets are enough.
If m > > k, then problem of duplicates at receiver is reduced.

Linear coding and greedy decoding



Greedy decoding can get stuck:



Nevertheless, we will stick to using greedy decoding.

LT codes (M. Luby) random, rateless, linear codes

Given the number of source symbols k, and a probability distribution on {1,2, . . . , k}, a coded symbol is generated as follows:

e.g., k=8

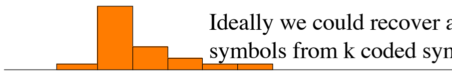
Step one: select a degree d with the given probability distribution. — e.g., d=3.

Step two: select a subset of {1, . . . , k} of size d. — e.g., {3,5,6} code vector 00101100

Step three: form the sum of the source symbols with indices in the random set — e.g., form S3+S5+S6

The resulting coded symbol can be transmitted along with its code vector.

Ideal soliton distribution for degree (Luby)



Ideally we could recover all k source symbols from k coded symbols.

steps complete v	1	2	3	4	$\cdots$	$k-2$	$k-1$	k
0	1	$\frac{k}{2}$	$\frac{k}{6}$	$\frac{k}{12}$	$\cdots$	$\frac{k}{(k-2)(k-3)}$	$\frac{k}{(k-1)(k-2)}$	$\frac{k}{k(k-1)}$
1	1	$\frac{k-1}{2}$	$\frac{k-1}{6}$	$\frac{k-1}{12}$	$\cdots$	$\frac{k-1}{(k-2)(k-3)}$	$\frac{k-1}{k(k-1)}$	0
2	1	$\frac{k-2}{2}$	$\frac{k-2}{6}$	$\frac{k-2}{12}$	$\cdots$	$\frac{k-2}{(k-2)(k-3)}$	0	0
$\vdots$	$\vdots$				$\vdots$		$\vdots$	
$k-2$	1	1	0	0	$\cdots$			
$k-1$	1	0	0	0	$\cdots$			

--The ideal soliton distribution doesn't work well due to stochastic fluctuations. Luby introduced a robust variation (see LT paper)

--Raptor codes use precoding plus LT coding to help at the end.

Analysis

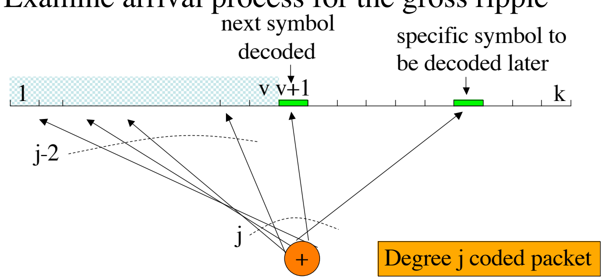
A coded symbol with reduced degree one (which has not been processed yet) is said to be in the <u>gross ripple</u>. Let X_v denote the number of symbols in the gross ripple after v symbols have been decoded.

Decoding successful if and only if ripple is nonempty through step k-1.

Study Poisson case:

Let the number of coded symbols of each degree j have the $Poi(k\beta_j)$ distribution, where $\beta_1, \cdots, \beta_k$ are nonnegative. The total number of symbols is thus $Poi\left((\sum \beta_i)k\right)$ distributed.

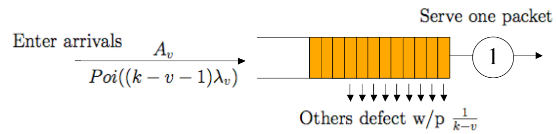
Examine arrival process for the gross ripple



$$\lambda_v = \sum_{j=2}^{v+2} \frac{k\beta_j \binom{v}{j-2}}{\binom{k}{j}} \approx \sum_{j=2}^{\infty} k\beta_j j(j-1)t^{j-2} = k\beta''(t)$$

where $t = \frac{v}{k}$ is scaled time, and β is the function $\beta(t) = \sum_{j=2}^{\infty} \beta_j t^j$.

Evolution of gross ripple

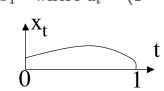


$$X_{v+1} = X_v - 1 - L_v + A_v$$

- X_0 is $Poi(k\beta_1)$
- A_v is $Poi((k-v-1)\lambda_v)$
- L_v given $X_v = j$ is $Binom(j-1, \frac{1}{k-v})$.

Fluid limit of $\frac{X_{kt}}{k}$: $\dot{x}_t = a_t - 1 - \frac{x_t}{1-t}$; $x_0 = \beta_1$ where $a_t = (1-t)\beta''(t)$.

Solution: $x_t = (1-t)(\beta'(t) + \ln(1-t))$

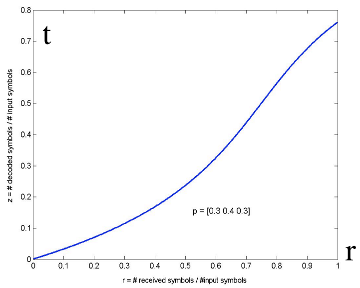


Intermediate Performance

An application of fluid limit (S. Sanghavi's poster, this conference)

Let K = # input symbols. If # received coded symbols is rK, for r<1, then the number of inputs recovered is tK where

$$t = \inf\{z : r\beta'(z) + \log(1-z) < 0\}$$

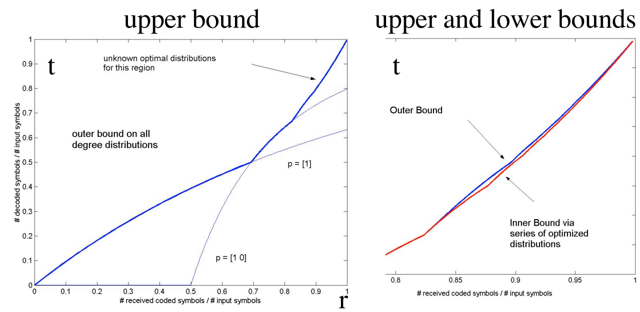


Example: for

$$\beta = (0.3\ 0.4\ 0.3)$$

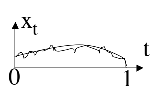
S. Sanghavi investigated maximizing t with respect to the degree distribution, for r fixed.

Sujay's bounds



Next: diffusion analysis

Solution: $x_t = (1-t)(\beta'(t) + \ln(1-t))$



Diffusion limit of gross ripple

Recall: $X_{v+1} = X_v - 1 - L_v + A_v$

$\dot{x}_t = a_t - 1 - \frac{x_t}{1-t};$

- X_0 is $Poi(k\beta_1)$
- A_v is $Poi((k-v-1)\lambda_v)$
- L_v given $X_v = j$ is $Binom(j-1, \frac{1}{k-v})$.

Let $\widetilde{x}_t = \text{weak}\lim_{k\to\infty} \dfrac{X_{tk} - kx_t}{\sqrt{k}}$

Then, for a Brownian motion $\widetilde{W}$,

$$d\widetilde{x}_t = -\frac{\widetilde{x}_t}{1-t}dt + b_t d\widetilde{W}_t; \qquad \widetilde{x}_0 \sim N(0, \beta_1)$$

where $b_t^2 = a_t + \frac{x}{1-t}$

Yields $\widetilde{x}_t = (1-t)(\frac{\widetilde{x}_t}{1-t}) = (1-t)W(\frac{\beta'(t)+\log(1-t)+t}{1-t})$

In particular, $\text{Var}(\widetilde{x}_t) = (1-t)(\beta'(t) + \log(1-t) + t) = x_t + t(1-t)$

Design based on diffusion limit

The diffusion limit result suggests the representation:

$$X_v \overset{d.}{\approx} kx(t) + \sqrt{k}\sigma_t N(0,1)$$

which in turn suggests guidelines for the degree distribution:

$$\begin{aligned} kx(t) &\geq c\sqrt{k}\sigma_t \qquad \text{for } 0 \leq t \leq 1-\delta \\ \text{or } k(1-t)(\beta'(t) + \ln(1-t)) &\geq c\sqrt{k}\sqrt{t(1-t)+x(t)} \quad (\text{drop } x(t)) \end{aligned}$$

or

$$\beta'(t) \geq -\ln(1-t) + \frac{c}{\sqrt{k}}\sqrt{\frac{t}{1-t}} \qquad \text{for } 0 \leq t \leq 1-\delta$$

Nearly same as suggested in [Sokrollahi 2006] based on Luby heuristic:

$$\beta'(t) \geq -\ln\left(1 - t - c\sqrt{\frac{1-t}{k}}\right)$$

The net ripple (used by Luby in LT paper)

The net ripple is the set of input symbols represented in the gross ripple. Let Y_v denote the number of symbols in the gross ripple after v symbols have been decoded. Decoding successful if and only if the net ripple is nonempty through step k-1.

$$Y_{v+1} = Y_v - 1 + \widehat{A}_v$$

where

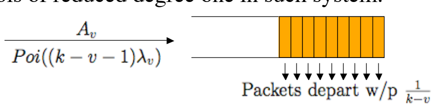
- Given $Y_v = j$, $\widehat{A}_v$ is $Binom(k - j - Y_v, 1 - e^{-\lambda_t})$
- Y_0 is $Binom(\mathrm{k}, 1 - e^{-\beta_1})$

Notes:

- Net ripple has finite state space $\{0, 1, \ldots, k\}$.
- Net ripple satisfies ode: $\dot{y} = a_t\left(\frac{1-t-y_t}{1-t}\right) - 1$
- $E[Y_v|X_v] = (n-v)(1 - e^{\frac{X_v}{n-v}})$
- Gross and net ripple fluid limits are similarly related by $y_t = (1-t)(1 - \exp(\frac{-x_t}{1-t}))$

The free ripple (understanding fluid approximation)

Suppose that input symbols are revealed one at a time by a genie, independently of the packets received. The degrees of coded symbols are decreased as the genie reveals symbols. The free ripple is the set of coded symbols of reduced degree one in such system.



$$Z_{v+1} = Z_v - D_v + A_v$$

- Z_0 is $Poi(k\beta_1)$
- A_v is $Poi((k-v-1)\lambda_v)$
- D_v given $Z_v = j$ is $Binom(j, \frac{1}{k-v})$.

Fluid limit of $\frac{Z_{kt}}{k}$: $\dot{z}_t = a_t - \frac{z_t}{1-t}$; $x_0 = \beta_1$ where $a_t = (1-t)\beta''(t)$.

Solution: $z_t = (1-t)\beta'(t)$

References

[1] M. Luby, "LT Codes," *Proc. IEEE Symposium on Theory of Computing*, 2002. (Presents LT codes.)

[2] R. W. R. Darling, J. R. Norris, "Structure of large random hypergraphs," *Annals of Applied Probability* Vol. 15, No. 1A, 125-152, March 2005. (Establishes ode and diffusion limits for equivalent model.)

[3] E. N. Maneva and A. Shokrollahi, "New model for rigorous analysis of LT-codes," ArXiv, December 8, 2005. (Describes Markov structure, and points to connection with Darling and Norris.)

[4] A. Shokrollahi, "Raptor Codes," *IEEE Transactions on Information Theory*, June 2006. (Raptor codes use a combination of precoding and LT coding. Contains heuristic for choice of degree distribution, based on private communication from Luby)

Conclusions

•Coding at the source, network coding, and peer-to-peer networking (gossip type algorithms) provide a rich source of network design and analysis problems.
•Iterating design and analysis can lead to interesting tractable problems (as in LT code design)

Thanks!