On connections between network coding and stochastic network theory -- an introduction

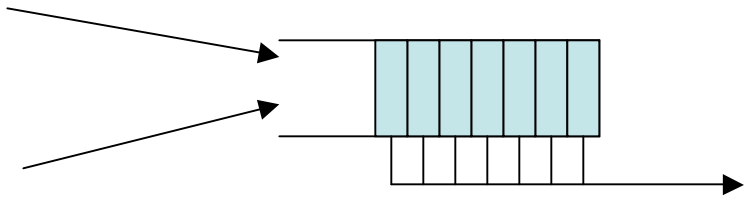
Bruce Hajek

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, including fluid and diffusion approximations, with implications for code design.

INFORMS, November 6, 2006

Orientation

Network coding is aimed at coding within a 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)

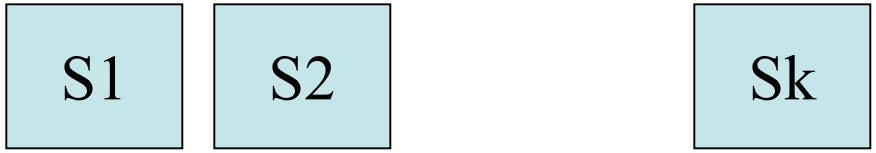
Stochastic network theory can be helpful in learning how to apply such codes in networks (e.g. tradeoff between buffer size and erasure protection) and in designing such codes. This talk focuses on the later.

Outline

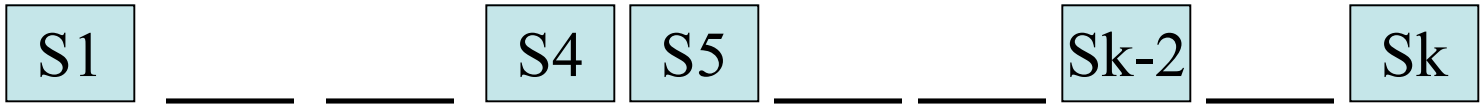
- Multicast, and linear codes for erasures
- Luby's LT codes
- Markov performance analysis for Poisson model
- Fluid approximation
- Diffusion approximation and an application
- Tree approach

Multicast with lost packets

source message: k symbols:
(fixed length binary strings)

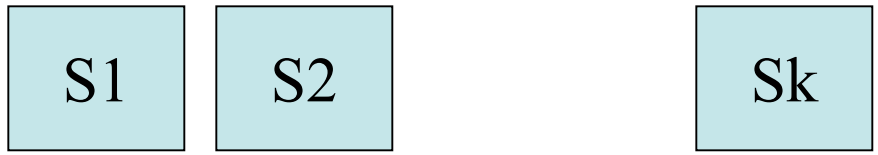


Symbols are repeatedly broadcast in random order
to many receivers, but can be lost.
Each receiver tries to collect a complete set, for example:

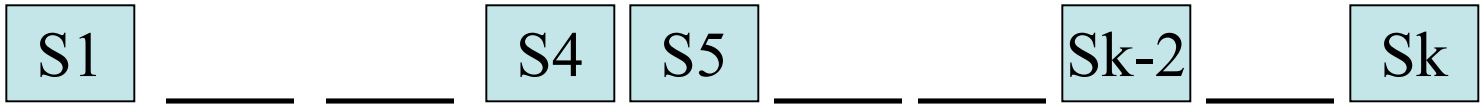


Multicast with lost packets

source message: k symbols:
(fixed length binary strings)



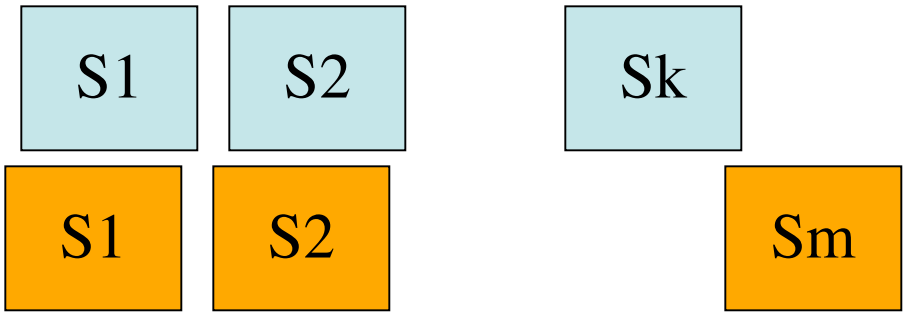
Symbols are repeatedly broadcast in random order
to many receivers, but can be lost.
Each receiver tries to collect a complete set, for example:



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

Multicast with coding at source, and lost packets

Source message: k symbols:
(fixed length binary strings)



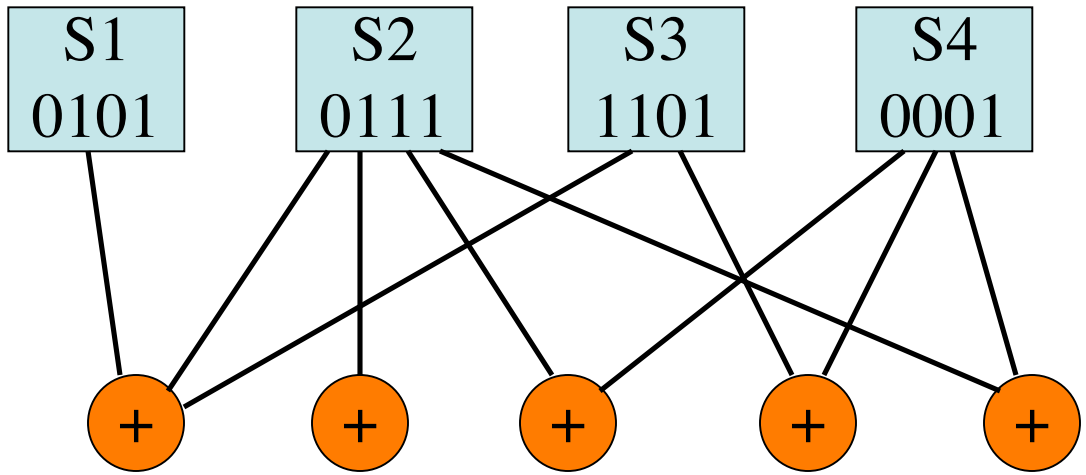
Source forms m coded symbols:

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

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

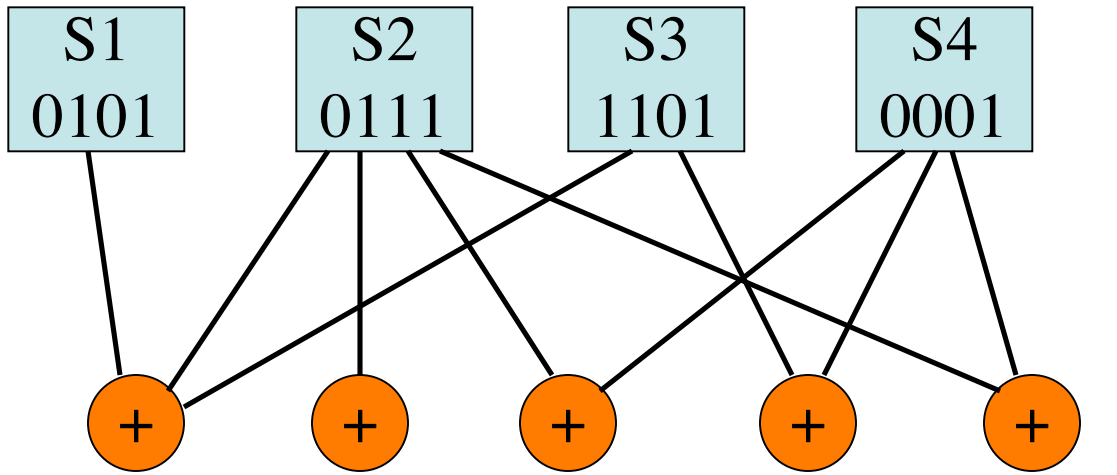
Source symbols:



Received payloads:

Linear coding and greedy decoding

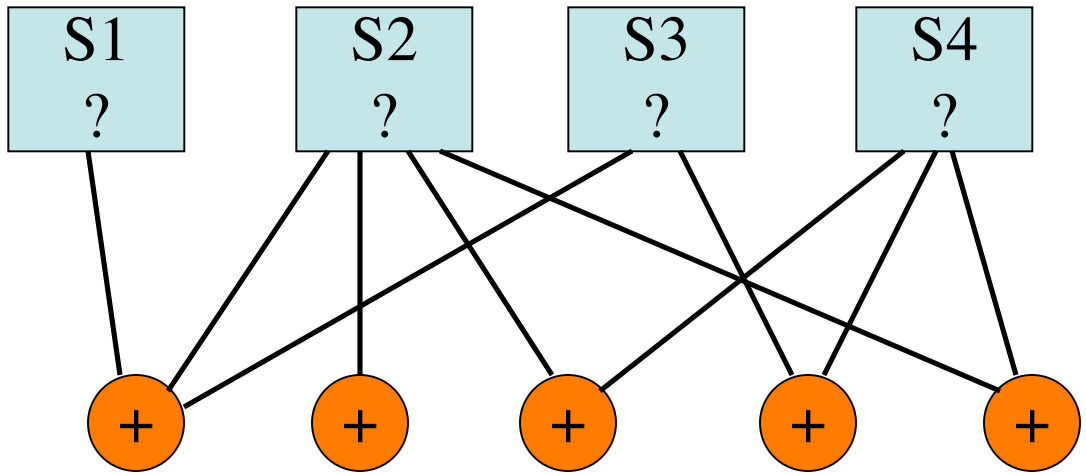
Source symbols:



Received payloads: 1111 0111 0110 1100 0110

Linear coding and greedy decoding

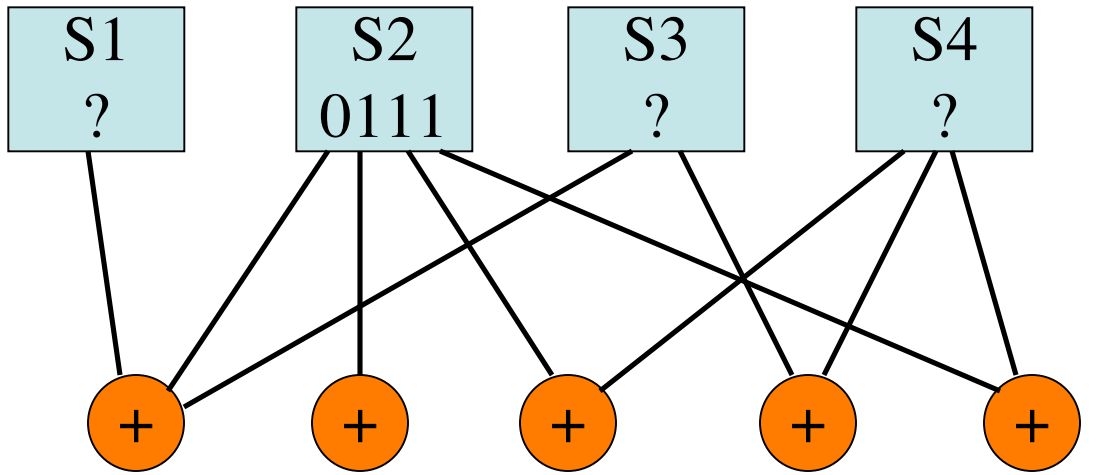
Source symbols:



Received payloads: 1111 0111 0110 1100 0110

Linear coding and greedy decoding

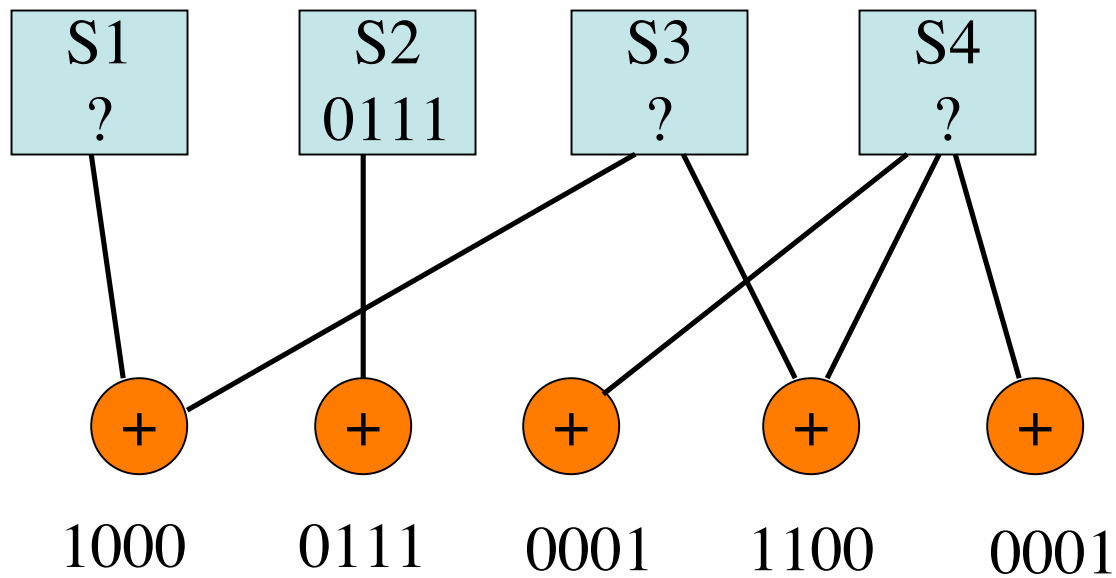
Source symbols:



Received payloads: 1111 0111 0110 1100 0110

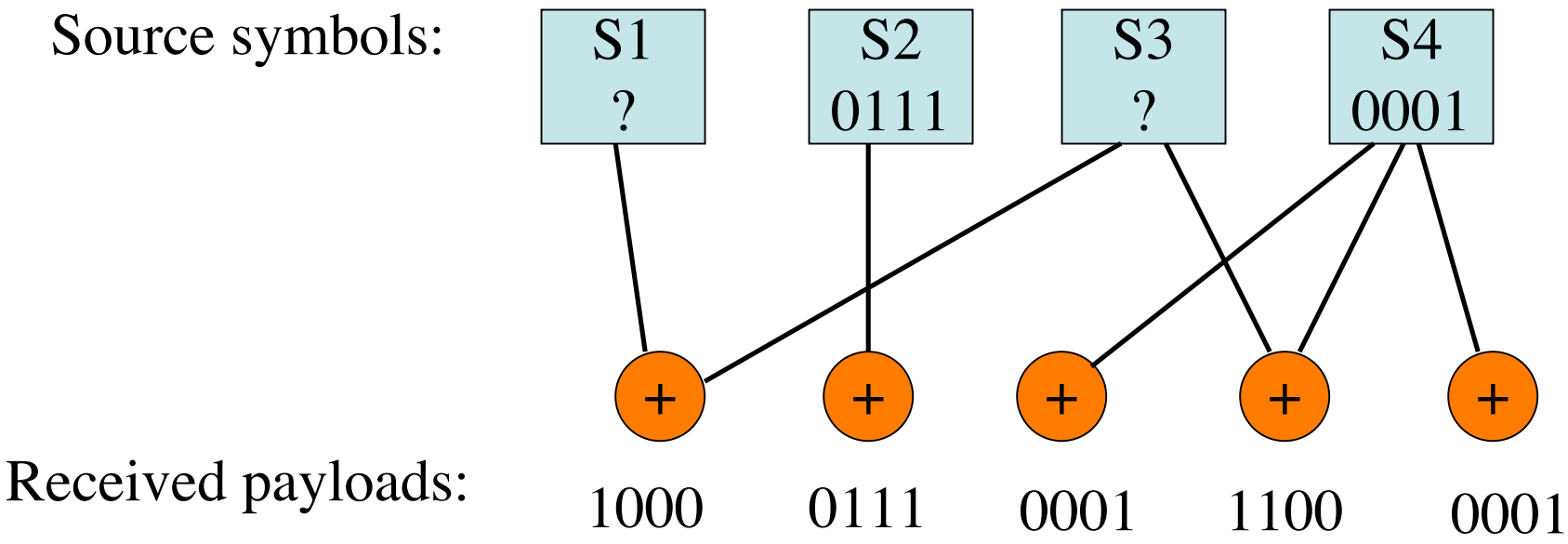
Linear coding and greedy decoding

Source symbols:

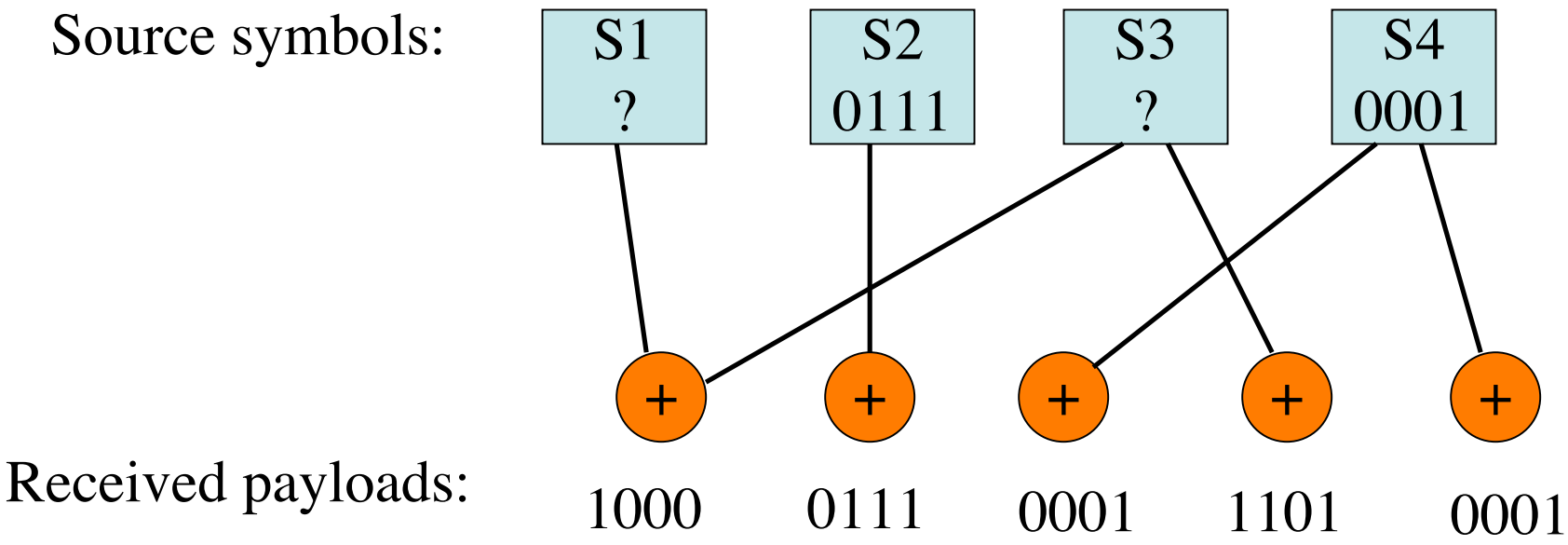


Received payloads:

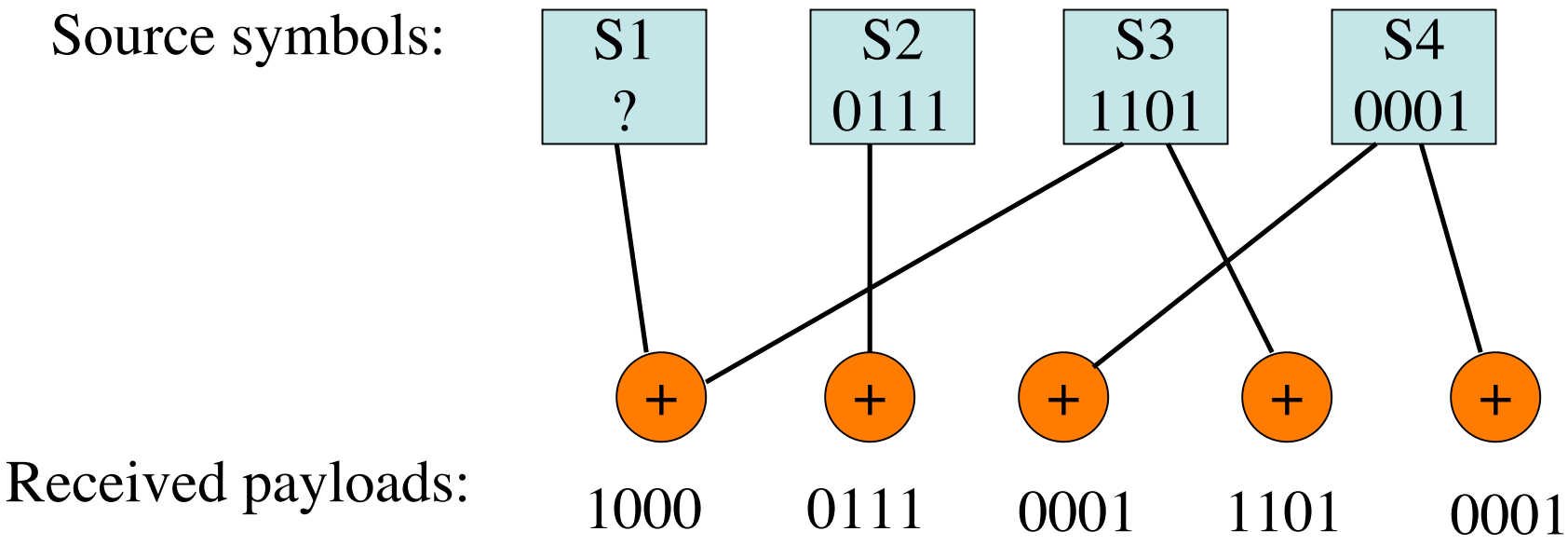
Linear coding and greedy decoding



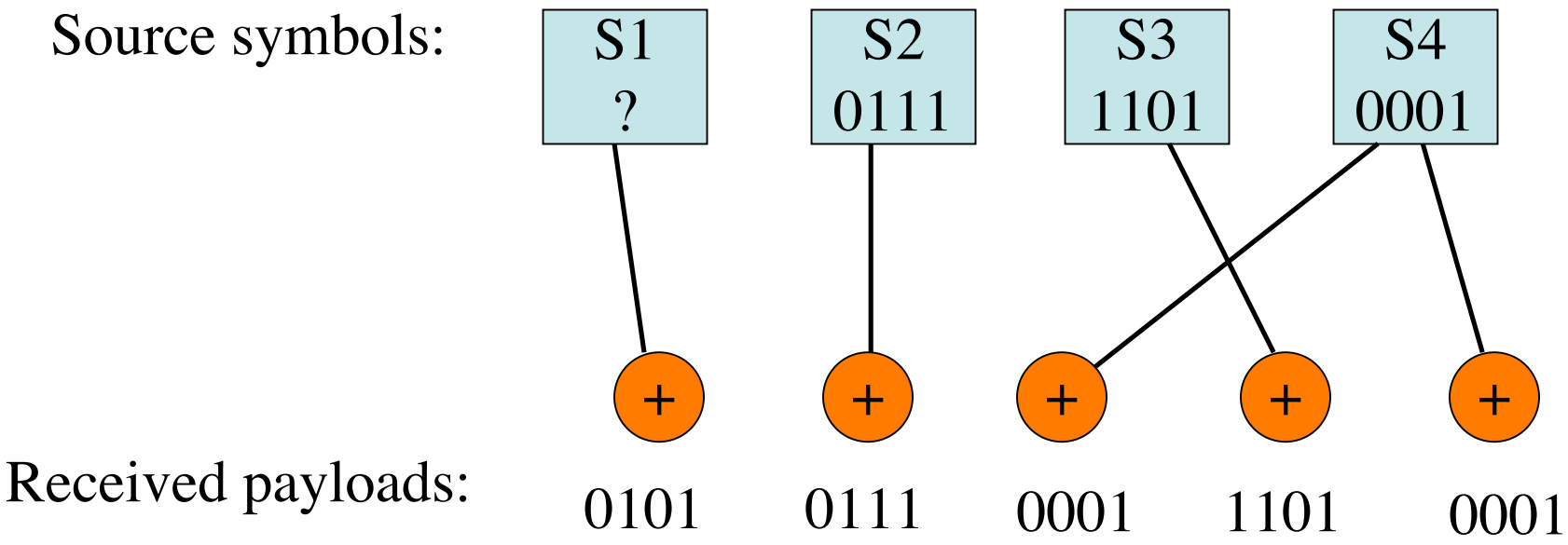
Linear coding and greedy decoding



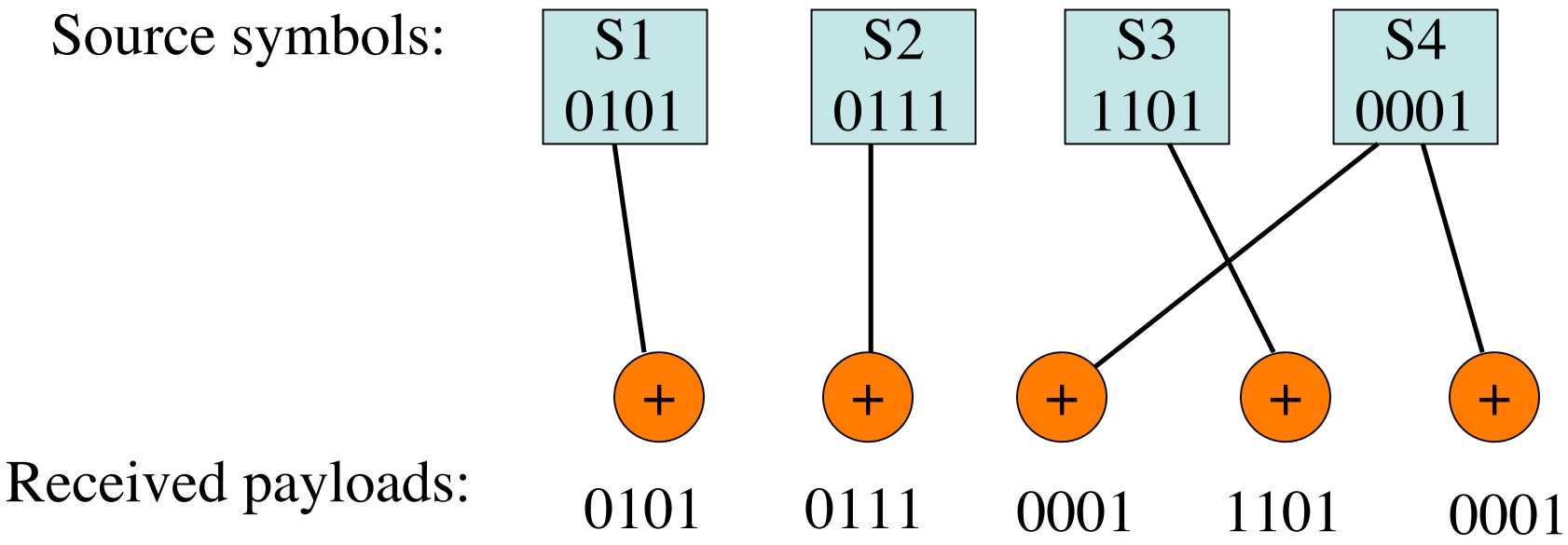
Linear coding and greedy decoding



Linear coding and greedy decoding

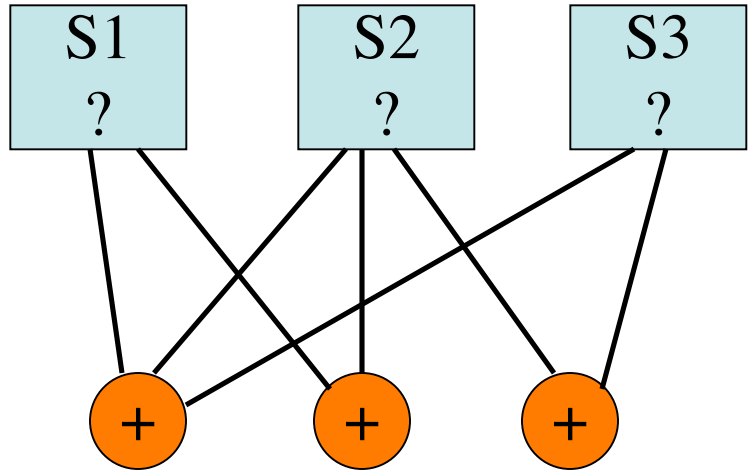


Linear coding and greedy decoding



Greedy decoding can get stuck:

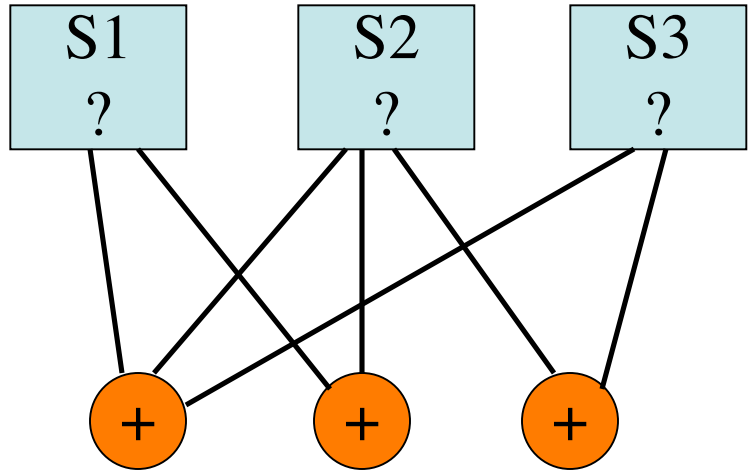
Source symbols:



Received payloads: 1000 0111 0001

Greedy decoding can get stuck:

Source symbols:



Received payloads: 1000 0111 0001

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

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.

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.

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.

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

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

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

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.

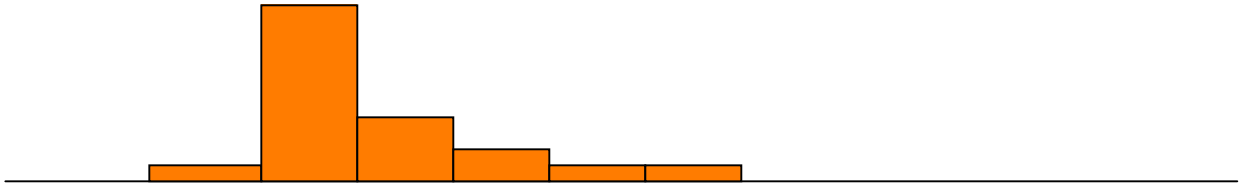
e.g., {3,5,6} code vector 00101100

Step three: form the sum of the source symbols with indices in the random set

e.g., form S3+S5+S6

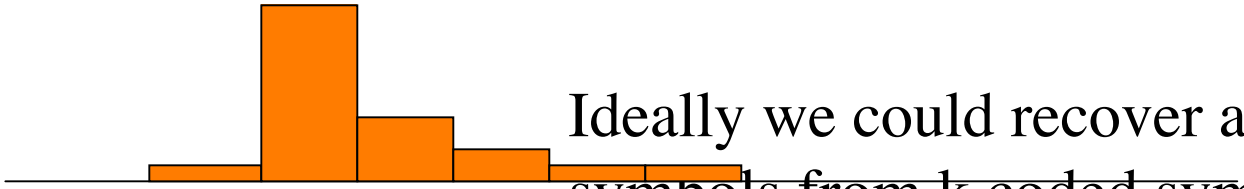
The resulting coded symbol can be transmitted along with its code vector.

Ideal soliton distribution for degree (Luby)



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	$\cdots$			

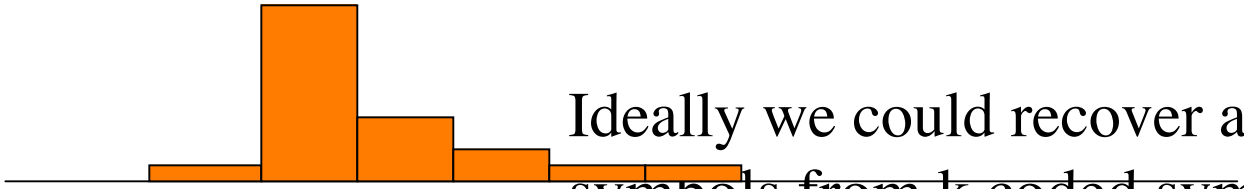
Ideal soliton distribution for degree (Luby)



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	$\cdots$			

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.

Analysis

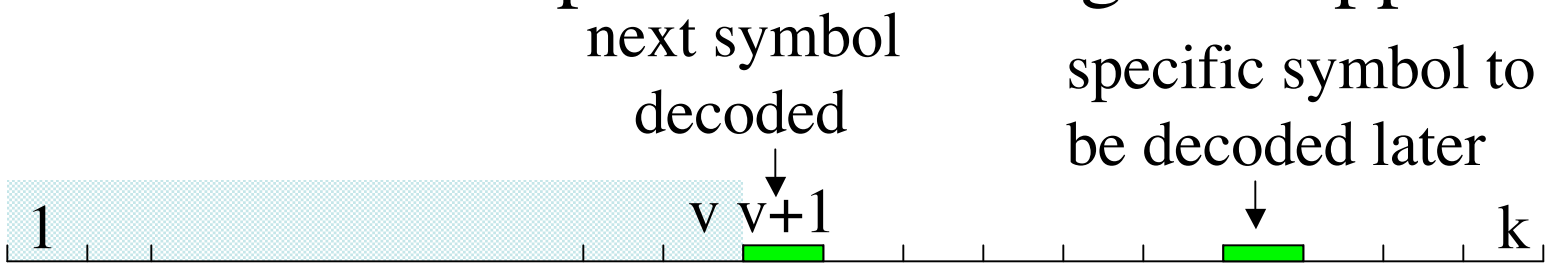
A coded symbol with reduced degree one (which has not been processed yet) is said to be in the <u>gross ripple</u>. Let X_v denote the number of symbols in the gross ripple after v symbols have been decoded.

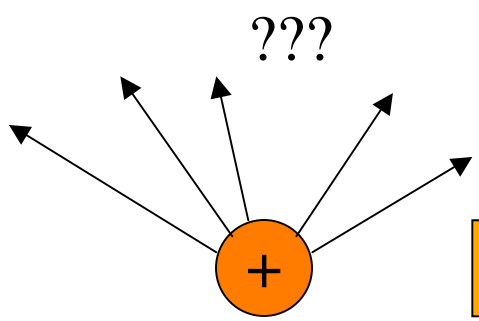
Decoding successful if and only if ripple is nonempty through step k-1.

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 $Poi\left((\sum \beta_j)k\right)$ distributed.

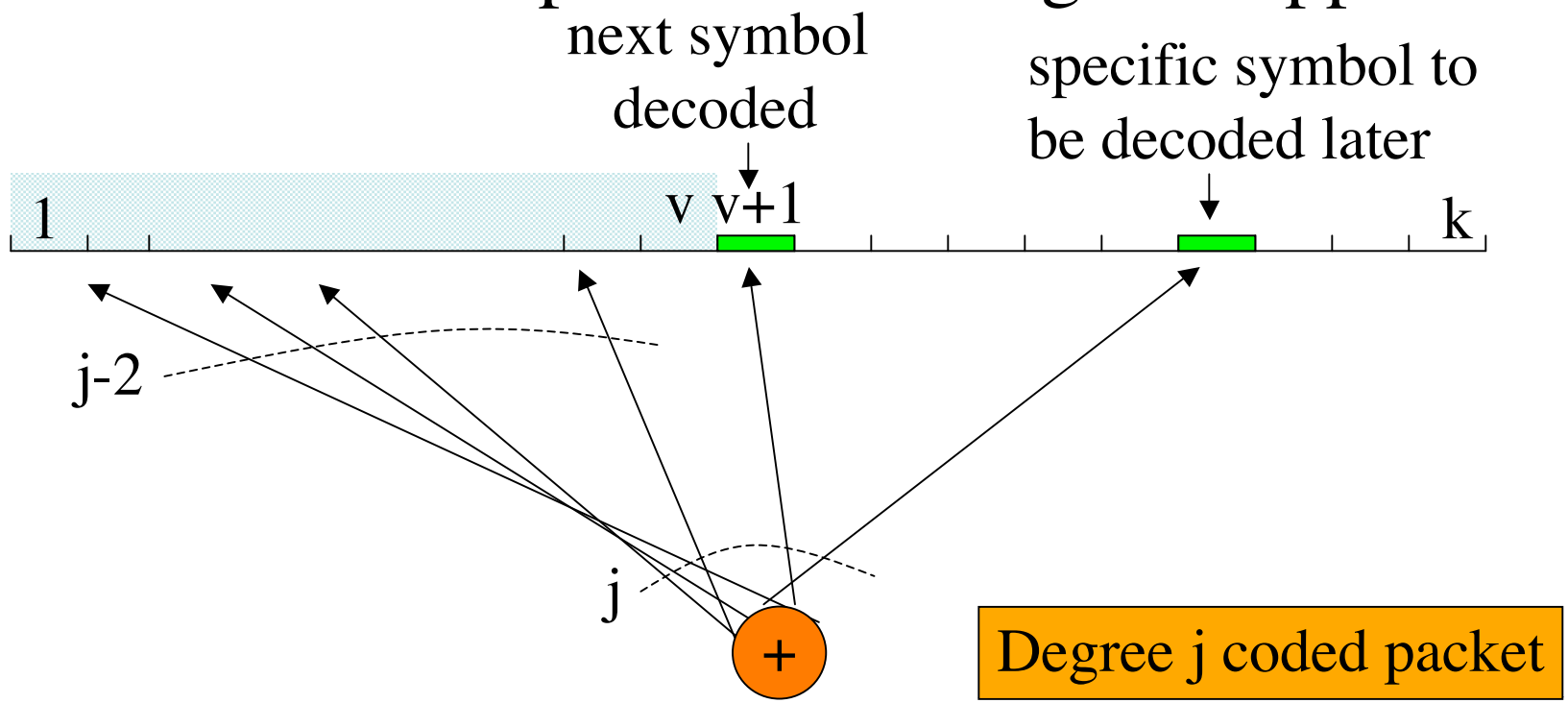
Examine arrival process for the gross ripple



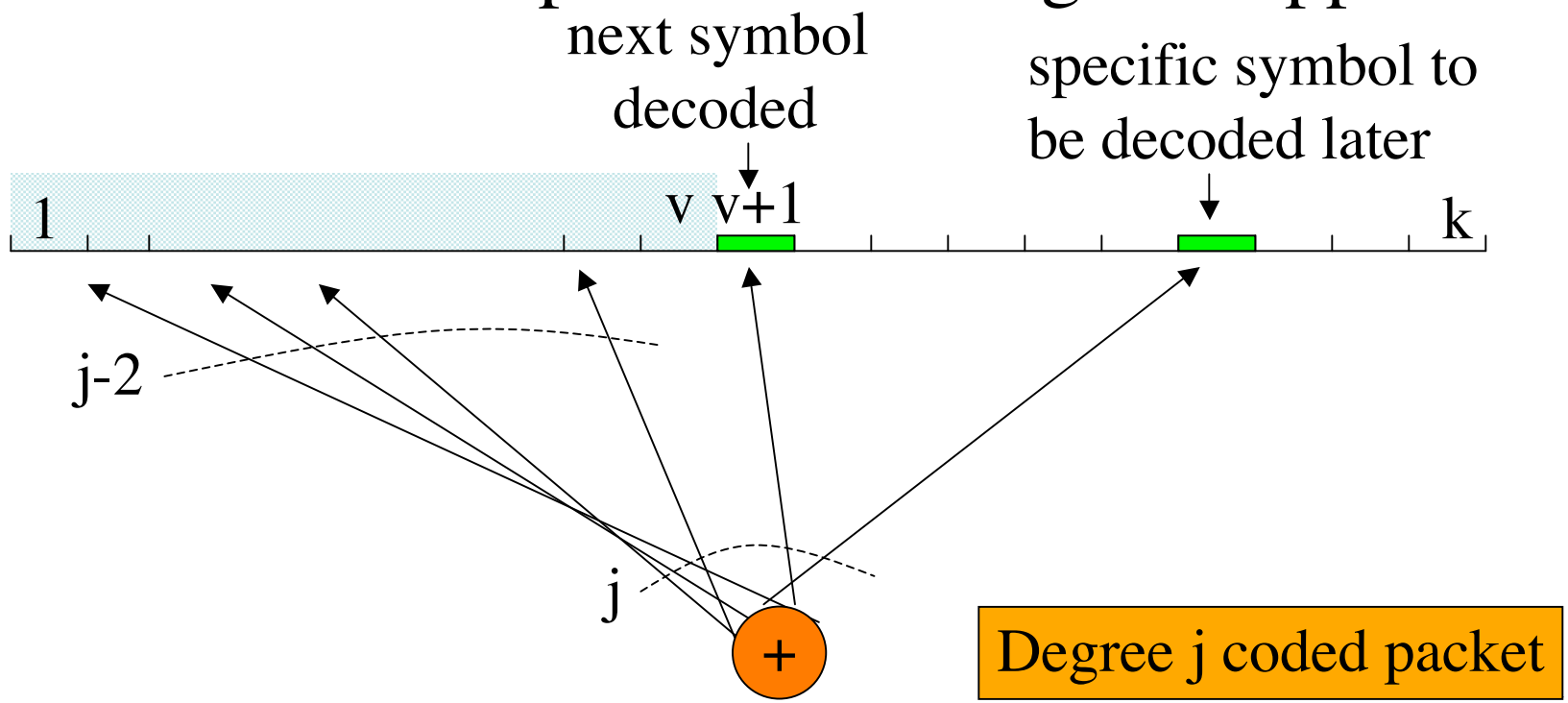


Degree j coded packet

Examine arrival process for the gross ripple



Examine arrival process for the gross ripple



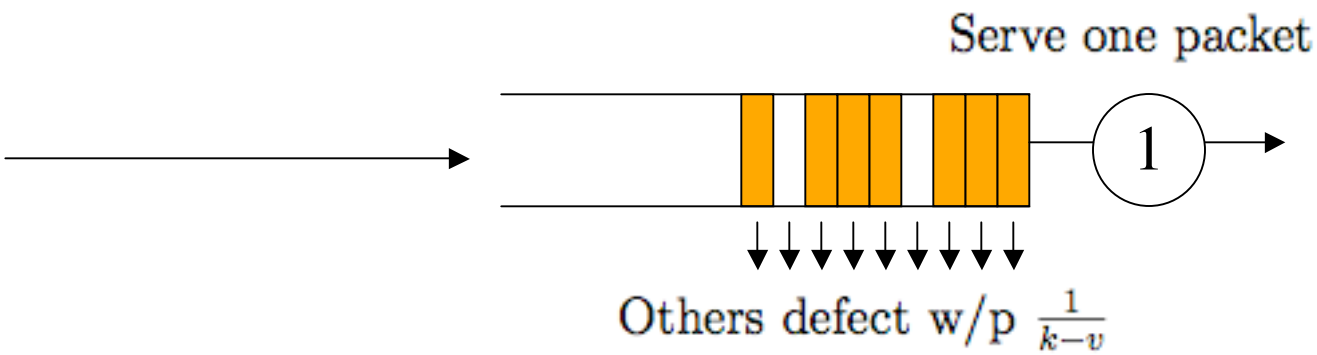
$$\begin{aligned}\lambda_v &= \sum_{j=2}^{v+2} \frac{k\beta_j \binom{v}{j-2}}{\binom{k}{j}} \\ &\approx \sum_{j=2}^{\infty} k\beta_j j(j-1)t^{j-2} \\ &= k\beta''(t)\end{aligned}$$

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

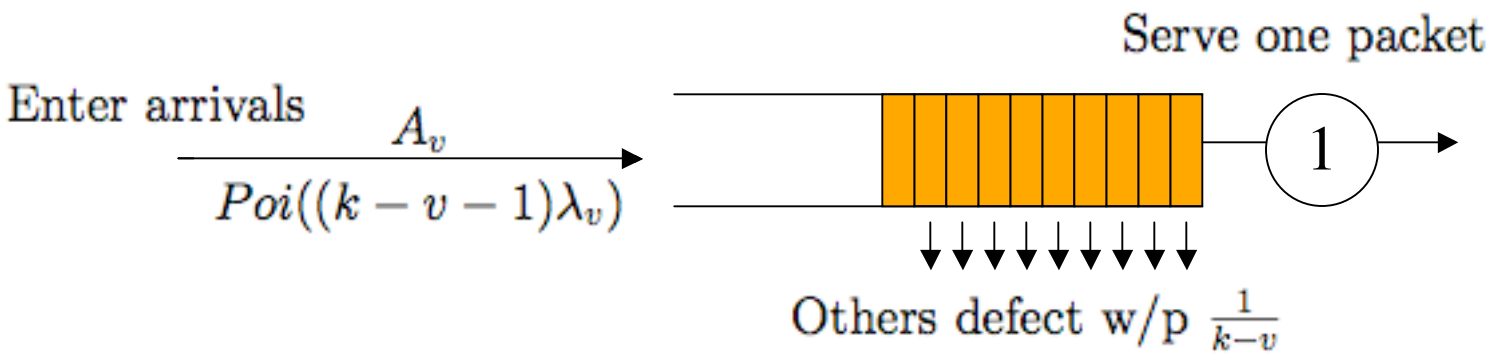
Evolution of gross ripple

Evolution of gross ripple

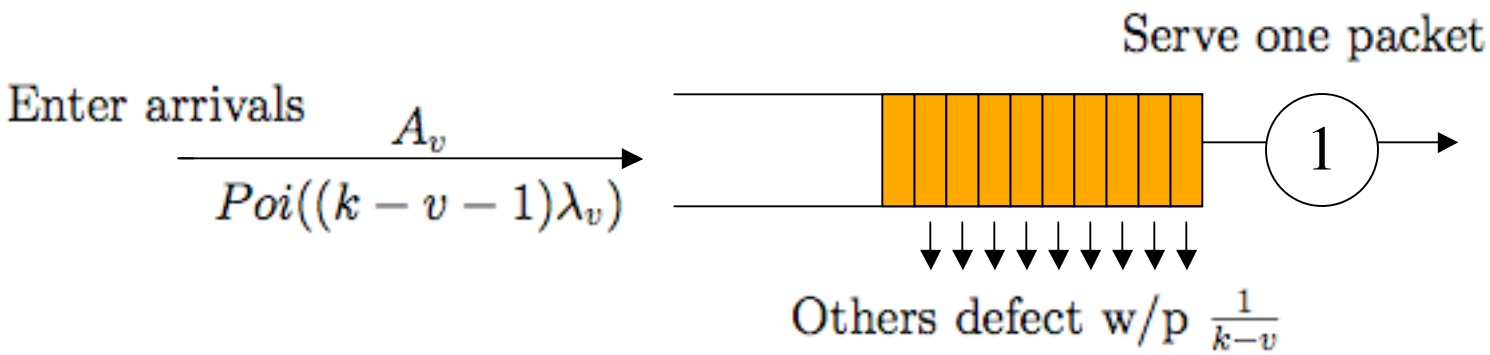
Evolution of gross ripple



Evolution of gross ripple



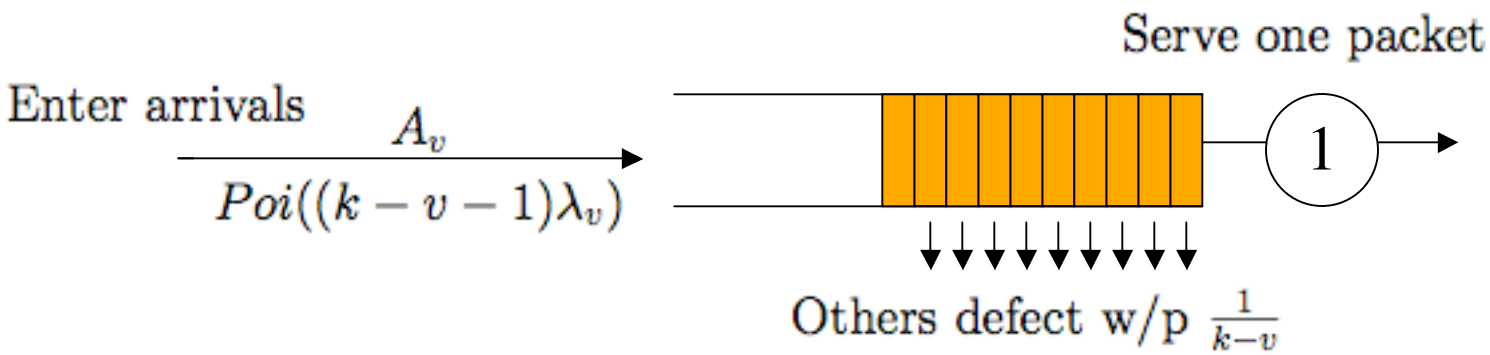
Evolution of gross ripple



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

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

Evolution of gross ripple

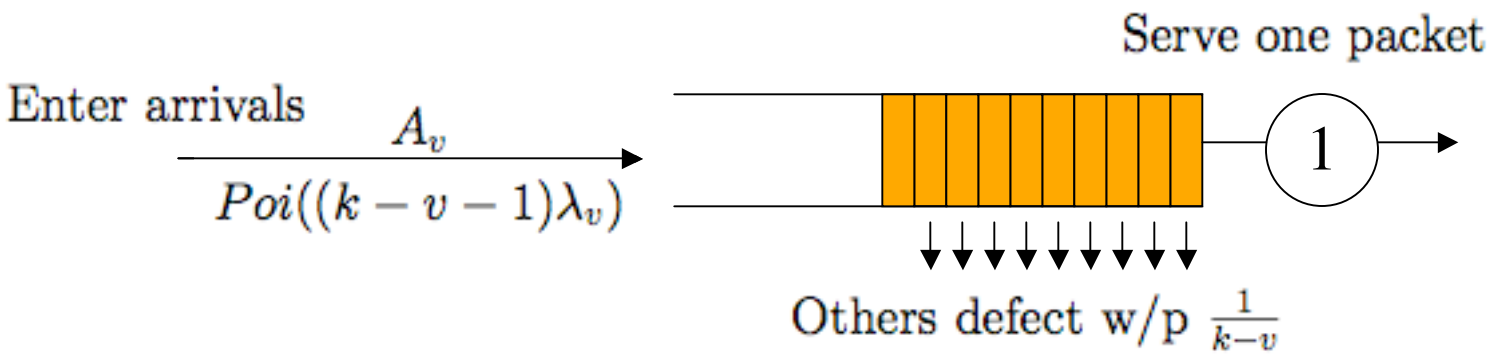


$$X_{v+1} = X_v - 1 - L_v + A_v$$

- 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 $a_t = (1-t)\beta''(t)$.

Evolution of gross ripple

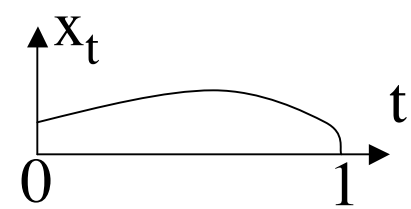


$$X_{v+1} = X_v - 1 - L_v + A_v$$

- 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 $a_t = (1-t)\beta''(t)$.

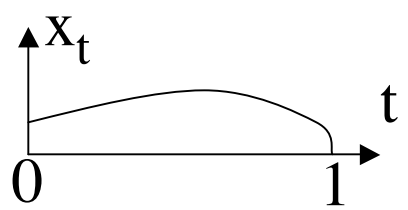
Solution: $x_t = (1-t)(\beta'(t) + \ln(1-t))$



Next: diffusion analysis

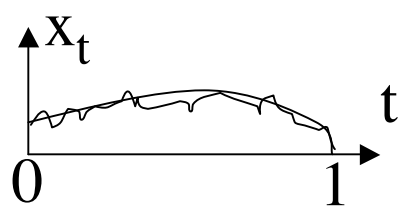
Next: diffusion analysis

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



Next: diffusion analysis

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



Diffusion limit of gross ripple

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

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

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

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

Diffusion limit of gross ripple

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

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

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

Diffusion limit of gross ripple

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

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

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

Diffusion limit of gross ripple

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

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

- 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:

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

Design based on diffusion limit

The diffusion limit result suggests the representation:

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

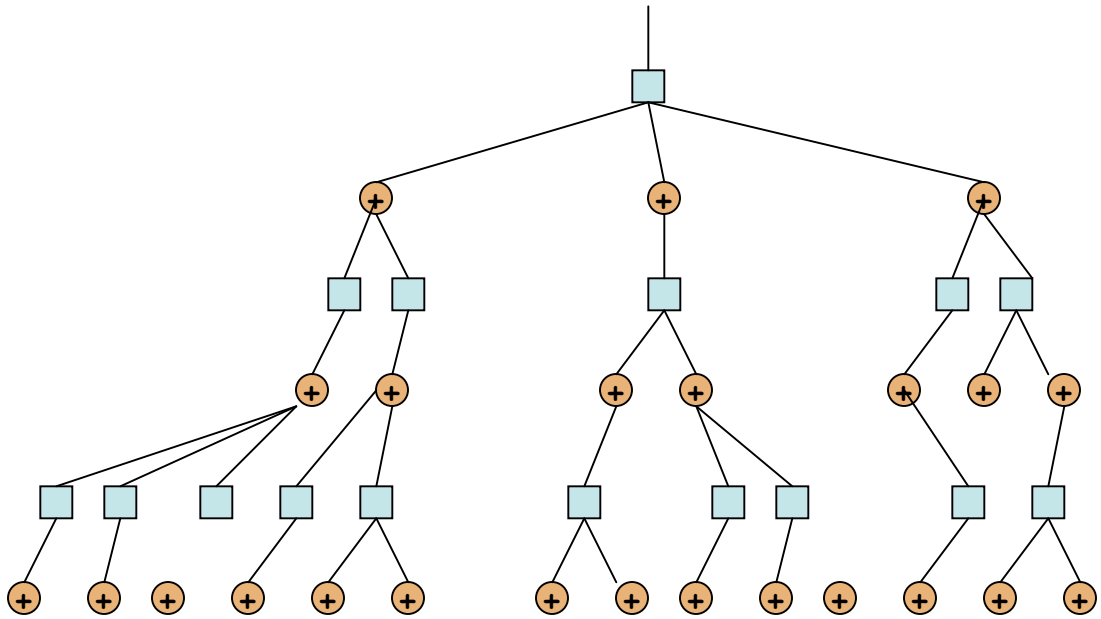
which in turn suggests guidelines for the degree distribution:

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

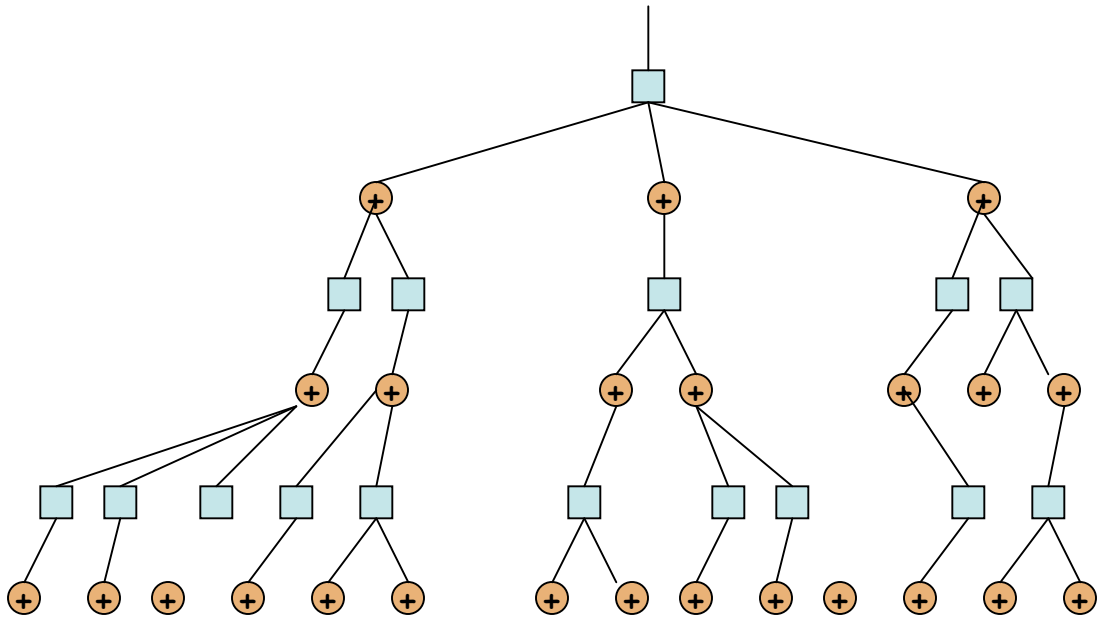
Nearly same as suggested in [Sokrollahi 2006] based on Luby heuristic:

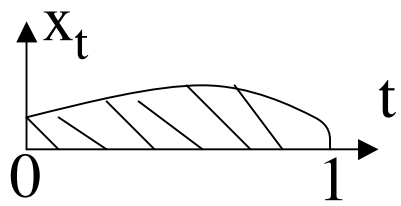
$$\beta'(t) \geq -\ln\left(1 - t - c\sqrt{\frac{1-t}{k}}\right)$$

Alternative analysis approach-- “tree method” (as in branching processes)



Alternative analysis approach-- “tree method” (as in branching processes)





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)
- ODE and related diffusion and large deviation analysis appear better able to provide error vs. block size estimates than tree based methods.
- For error correction, order of iterations matters. There doesn't seem to be an ODE analysis approach for errors.

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)
- ODE and related diffusion and large deviation analysis appear better able to provide error vs. block size estimates than tree based methods.
- For error correction, order of iterations matters. There doesn't seem to be an ODE analysis approach for errors.

Thanks!

Some key papers on codes for iterative decoding

[1] **R.G. Gallager**, *Low density parity check codes*, Cambridge, MA: MIT Press, 1963. (Introduced use of random codes for error correction, and iterative message passing decoding.)

[2] **C. Berrou, A. Glavieux, and P. Thitimajshima**, "Near Shannon limit error-correcting coding and decoding," in *Proc. Int. Conf. Communication* (ICC '93). Geneva, Switzerland, May 1993, pp. 1064-1070. (Introduced first high-performance code implementation based on iterative methods. Spawned much interest in iterative decoding methods.)

[3] **M. Sipser and D.A. Spielman**, "Expander codes," *IEEE Trans. on Information Theory*, vol. 42, pp. 1710 - 1722, 1996. (One of several papers focusing on expansion properties of graphs and fast decoding algorithms for error correction. Important tool for understanding "end game" for iterative decoding methods.)

[4] **M.G. Luby, M. Mitzenmacher, M. Amin Shokrollahi, D.A. Spielman, and V. Stemann** "Practical Loss Resillient Codes," *Proc. 29th Annu. ACM Symp. Theory of Computing*, 1997, pp. 150-159. (Introduced and demonstrated utility of irregular node degrees. First application of differential equation method to analyze simple greedy decoding technique. Context is erasure recovery.)

[5] **M.G. Luby, M. Mitzenmacher, M. Amin Shokrollahi, and D.A. Spielman,** "Efficient erasure correcting codes," *IEEE Trans. Information Theory*, vol. 47, pp. 569-584, February 2001. (Journal version of [4].)

[6] **M.G. Luby, M. Mitzenmacher, M.A. Shokrollahi, and D.A. Spielman,** "Improved low-density parity-check codes using irregular graphs," *IEEE Transactions on Information Theory*, 585- 598, 2001. (Discusses codes for error correction, combining irregular degree ideas from [4, 5] and tree analysis from [13].)

[7] **T.J. Richardson and R.L. Urbanke,** "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Transactions on Information Theory*, vol. 47, pp. 598-618, 2001. (Shows that tree analysis of message-passing decoding, now known at density evolu- tion, termed "density evolution", including a concentration bound, as in [13], works for error correction with general discrete or continuous output alphabets.)

[8] **M. Luby, "LT Codes,"** *Proc. IEEE Symposium on Theory of Computing*, 2002. (Presents LT codes, the first discovered digital fountain, a.k.a. rateless erasure recovery, codes. Similar to earlier erasure codes [4], but LT codes are nonsystematic, and the degree distribution of input symbols is Poisson, rather than designed. Soliton distribution is introduced, and robust soliton distribution is shown to yield complete recovery of all input symbols with high probability.)

[9] **A. Shokrollahi, "Raptor Codes,"** IEEE Transactions on Information Theory, June 2006. (Rap- tor codes use a combination of precoding and LT coding. The paper contains a heuristic for choice of degree distribution for finite blocklength, suggested by M. Luby.)

Some key papers on related analysis and design

[10] **R. Karp and M. Sipser**, “Maximum matchings in sparse random graphs,” in *Proc. 22nd FOCS*, pp. 364-375, 1981. (An early, perhaps the first, application of the differential equation method to random graph problems.)

[11] **N.C. Wormald,** “Differential equations for random processes and random graphs,” *Ann. Appl. Probab.*, vol. 5, pp. 1217-1235, 1995. (Convergence theorem and examples for differential equa- tion analysis approach. Applied in [4]).

[12] **R. W. R. Darling and J. R. Norris,** “Structure of large random hypergraphs,” *Annals of Applied Probability* Vol. 15, No. 1A, 125-152, March 2005. (Provides convergence theorem for differential equation analysis approach, including second order diffusion limit analysis, for a graph problem that is equivalent to the LT decoding problem, and another graph problem.)

[13] **M.G. Luby, M. Mitzenmacher, M.A. Shokrollahi, and D.A. Spielman,** “Analysis of random processes via and-or trees,” in *Proc. 9th Annu. ACM-SIAM Symp. Discrete Algorithms*, pp. 354-373, 1998. (Shows how a martingale concentration argument can be used to provide bounds associated with tree type analysis of iterative decoding algorithms.)

[14] **E. N. Maneva and A. Shokrollahi,** “New model for rigorous analysis of LT-codes,” *Proc. IEEE International Symposium on Information Theory*, July 2006. (Describes Markov structure, points to relevance of [12], and gives an O(k2 log K) numerical algorithm for determining success probability for an LT code with k inputs.)

[15] **S.-Y.Chung, T.J. Richardson, and R.L. Urbanke,** “Analysis of sum-product decoding of low- density parity-check codes using a Gaussian approximation,” *IEEE Trans, Information Theory*, vol. 47, pp. 657-670, Feb. 2001. (Use of Gaussian approximation for density evolution analysis method. Effective heuristic method for analyzing infinite block length limit of low density parity check codes for error correction by belief propagation/iterative decoding.)