26/7/21

EE 605

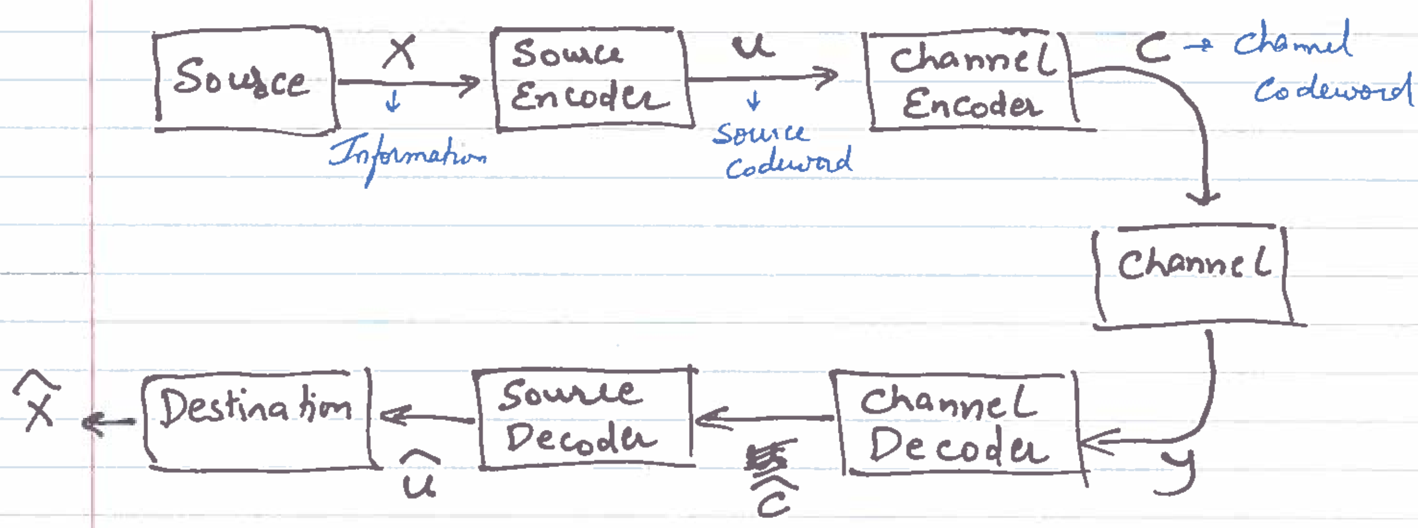


Fig. of a general communication system.

Channel is noisy and can introduce errors. That module is the focus of this course.

We will use a probabilistic (discrete) channel

$F \rightarrow$ input alphabet , $\phi \rightarrow$ output alphabet

Prob.(y received | x transmitted)

$(x, y) \in F^m \times \phi^m \quad , \quad m \in \mathbb{Z}^+$

Want $\hat{c} = c$ & $\hat{u} = u$

Input to channel encoder $\rightarrow$ Message $u \in \{1, 2 .. M\}$

Channel Encoder generates codeword $c \in F^n$ via one-to-one map

Received output of channel $\rightarrow y \in \phi^n$

Decoder generates $\hat{c} \rightarrow$ decoded codeword, $\hat{u} \rightarrow$ decoded message

Rate of communication $\rightarrow$ $R \triangleq \dfrac{\log_{|F|} M}{n}$. $\left(= \dfrac{\log_2 M}{n \log_2 |F|} \right)$

Amount of information communicated per channel use.

Eg $\rightarrow$ $M = \{1, 2, 3, 4\}$ $\quad F = \phi = \{0, 1\}$, $n = 5$

	M		C
(00)	1	$\rightarrow$	10101
(01)	2	$\rightarrow$	10010
(10)	3	$\rightarrow$	01110
(11)	4	$\rightarrow$	11111

$R = \dfrac{\log_2 4}{5} = \dfrac{2}{5}$

Usually, we will have $M = |F|^K$ and so rate $R = \frac{K}{n}$

Note that since message $\rightarrow$ codeword map is one-to-one,

$K \leq n$ & $R \leq 1$.

If one of the four codewords in C is received, we take the corresponding M to be the true message.

If some other vector y is received, then try to find most likely transmitted codeword. For eg. if received word is 11101, perhaps assume 10101 is true codeword & $M=1$ was transmitted.

In general, goal is to have high rate and high error correction/detection capability. In general, a tradeoff.

Uncoded $\rightarrow$ $|M| = |F|^n$ $\rightarrow$ $R = 1$ (no error handling)

Repetition $\rightarrow$ $M = \{1, 2\}$, $\quad 1 \rightarrow 00\ldots0$ (n times) $\quad R = 1/n$

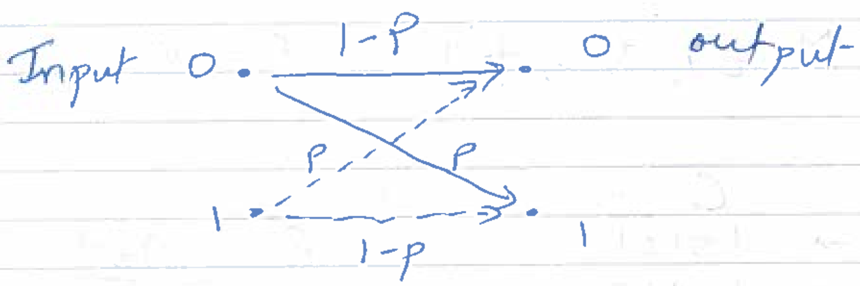
$2 \rightarrow 11\ldots1$ (n times) $\quad \frac{n-1}{2}$ errors corrected

n odd

Channels

Eg1 – Memoryless binary symmetric channel. (BSC)

$F = \phi = \{0, 1\}$



For $x = (x_1, x_2 \ldots x_m) \in \{0,1\}^m$

$y = (y_1, y_2 \ldots y_m) \in \{0,1\}^m$

$$\mathbb{P}(y \text{ received} \mid x \text{ transmitted}) = \prod_{i=1}^{m} \underbrace{\mathbb{P}(y_i \text{ received} \mid x_i \text{ tran})}_{\substack{1-p \quad y_i = x_i \\ p \quad y_i \neq x_i}}$$

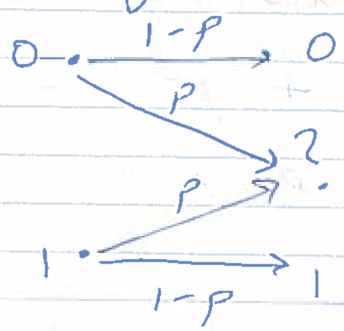
Eg2 → q-ary symmetric channel: a generalization of BSC

$F = \phi = \{0, 1, \ldots q-1\}$

$y = x$ w.p. $1-p$,

$y = j$ w.p. $(1-p)/q$ for each $j \neq x$.

Eg3 → Binary Erasure Channel (BEC)



$F = \{0, 1\}$

$\phi = \{0, 1, ?\}$

Input symbol erased with prob. p.

~~Most of the course will focus on~~

Decoding — A decoder is a fn.

$D : \phi^n \to C$ $\quad C$ = set of codewords [codebook]

Takes as input received word and outputs an estimate $\hat{c}$ of transmitted codeword c.

Error prob. → $P_e = \max_{c \in C} P_e(c)$

where

$$P_e(c) = \sum_{y : D(y) \neq c} \Pr(y \text{ received} \mid c \text{ transmitted})$$

Goal → Decoders with low P_e.

Eg → BSC(p). For uncoded transmission, $P_e^{un} = p$.

$C = \{000, 111\}$ Take repetition code with $n = 3$, and say majority decoder

$D(000) = D(010) = D(001) = D(100) = 000$

$D(011) = D(110) = D(101) = D(111) = 111$

$$\text{Prob(Error)} = P_e = \text{Prob}(\geq 2 \text{ errors})$$

$$P_e^{rep} = \binom{3}{2} p^2(1-p) + p^3 = 3p^2 - 3p^3 + p^3$$

$$= p(2p-1)(1-p) + p$$

Note $P_e^{rep} < P_e^{un}$ for $p < \frac{1}{2}$

Coding has improved prob. of error

Price is in rate $R^{rep} = \frac{\log_2 |M|}{n} = \frac{1}{3}$.

Applications

Beyond simple repetition ⇒ Combining of codeword symbols to introduce redundancy

(1) Coding for Communication / Storage :

To deal with noise / fading in communication channel.

To deal with errors / erasures in storage ranging from CDs, hard drives to flash memory & DNA storage

Eg. Simple parity check code. Parity helps to detect errors

0 0	→	0 0 0
0 1	→	0 1 1
1 0	→	1 0 1
1 1	→	1 1 0

Parity bit (XOR of message bits)

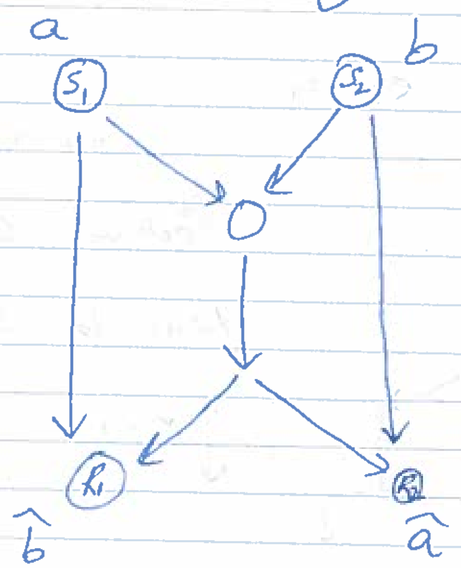
If 001 received, unsure if tx is 000 or 011

Rate $= \frac{\log_2 4}{3} = \frac{2}{3}$

Can detect one error. Since all valid outputs have even 1's.

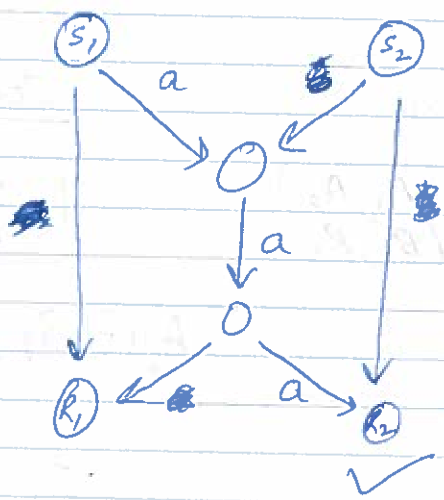
Cannot correct any errors – Can with more parity bits added.

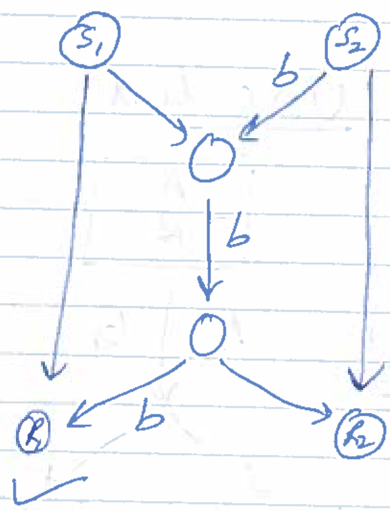
(2) Network coding — Increasing network throughput



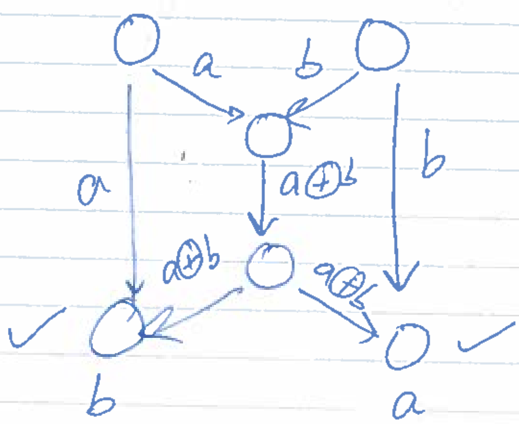
Each link can carry one symbol $\in \{0,1\}$

$a, b \in \{0, 1\}$





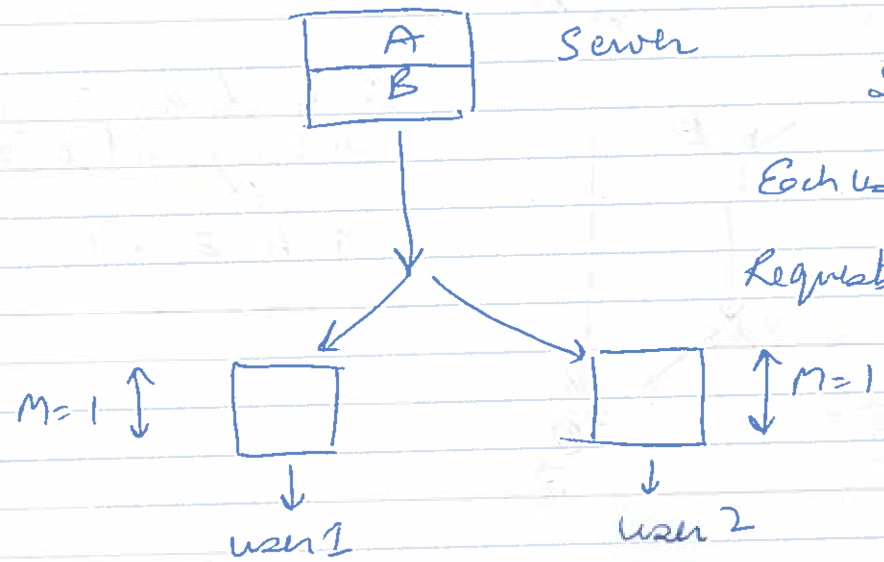
Uncoded transmission. Need two uses of network $\rightarrow R = 1/2$



Coded transmission in bottleneck link

$R = 1$

(3) Coded Caching (dual to network coding).



Server → 2 files

Each user can store upto 1 file

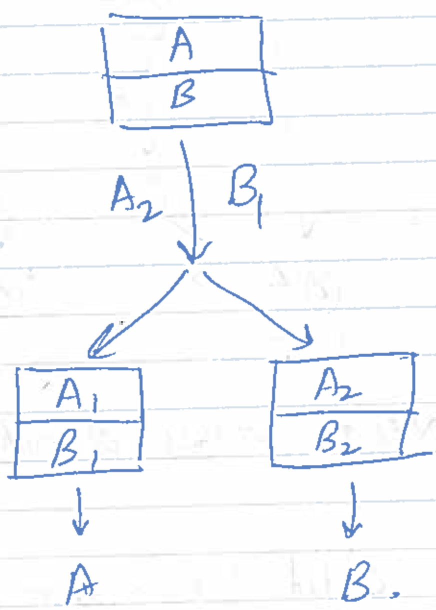
Requests one file at a time

Server broadcasts message.

Uncoded

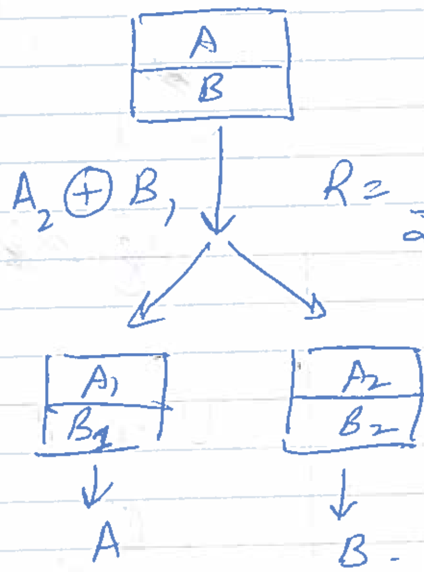
$R = \frac{1}{2} + \frac{1}{2}$

$= 1$



$A = (A_1, A_2)$
$B = (B_1, B_2)$

Coded



$R = \frac{1}{2}$

4 **Gradient coding:**

Problems of the form $\beta^* = \underset{\beta \in \mathbb{R}^p}{\arg\min} \sum_{i=1}^{d} L(x_i, y_i, \beta)$

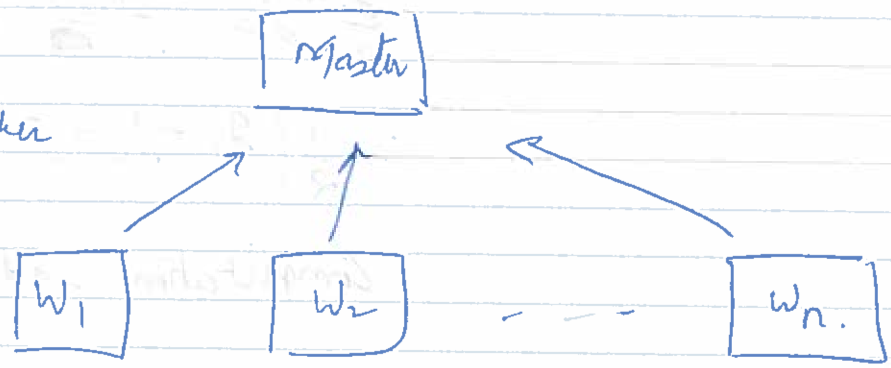
where L is some loss function over a dataset $D = \{x_i, y_i\}_{i=1}^{d}$

Gradient descent a popular method to solve this problem

$$g^{(t)} = \sum_{i=1}^{d} \nabla L(x_i, y_i, \beta^{(t)})$$

Hard to run this on a single machine. Distribute task to multiple workers.

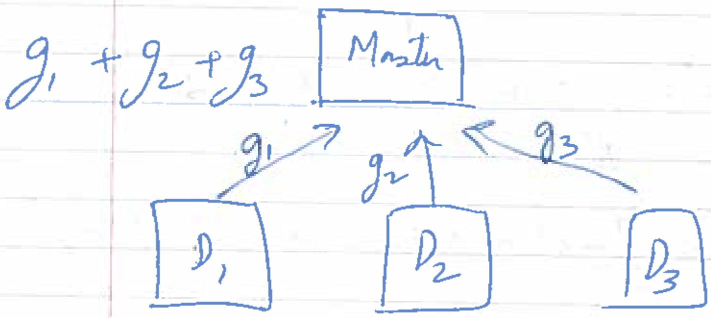
Each worker assigned some subset of D



Some workers may struggle, don't respond. Want to design schemes that work even if s out of n workers fail.

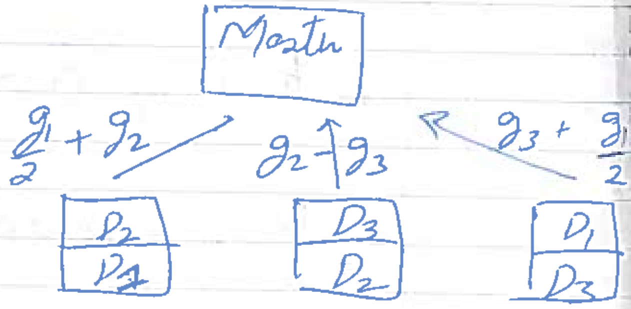
$$D = D_1 \cup D_2 \cup D_3$$

Uncoded



No resilience
Computation load 1/3.

Coded



Master can recover $g_1 + g_2 + g_3$ from any two transmissions.

Say node 3 fails

$$\frac{g_1}{2} + g_2 - \frac{1}{2}(g_2 - g_3)$$

$$= \frac{1}{2}(g_1 + g_2 + g_3)$$

Computation load $\rightarrow$ 2/3