Ramon + Alberto minutes 18/10/23

Timescale - simple version
↳ 3-4 weeks

simple case → 3VN - 1CN
↓
good idea

---

* Dry Run

$^nC_r = \frac{n!}{r!\,(n-r)!}$

4C2
n=4
r=2

$\frac{4!}{2!\,(2!)} = \frac{4 \times 2 \times 3}{2} = 6$

0 1 2 3

⊙⊙ • •	0	Symbols	(0,1)	5 < 4C2
⊙ • ⊙ •	1		(0,2)	
⊙ • • ⊙	2		(0,3)	Set of symbols
• ⊙⊙ •	3		(1,2)	
• ⊙ • ⊙	4		(1,3)	
• • ⊙⊙	5		(2,3) ✗	

* How we choose our mapping will be imp → some more susceptible to errors

↳ we're going to be restricted to a Finite Field → fraction of combinations you are not using is 1-p → not a problem

↳ is there a better choice for decoding?

* Decoder will need to exclude combinations

→ Non-Symmetrical channel if we don't use some combinations

↳ also as we get larger, gap increases → how does this affect decoding

* Prime density tends to zero for increasing (nk)
↳ pretty dense for the values we care about
↓
could work on Extension groups
↳ Addition modulo..
↳ Linear Mapping

Non Binary LDPC code on Group
↓
How are Field Operations replaced?

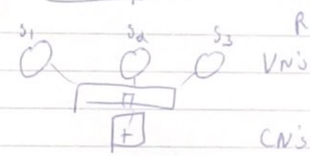
* We don't know much about it

FIELD OR GROUP?

Field
GF(5)
↳ subset of symbols
↓
for the simplest case does not matter which we choose
↓
Exclude (2,3)

Tanner Graph

R = 2/3



→ Single parity check code → * Easiest trivial case

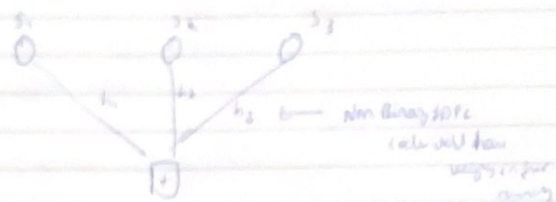
3
2

* Figure out Decoder Eqs

Field → Field mapping
Group → Linear mapping

$H = [h_1 \; h_2 \; h_3]$

GF(Q)



← Non Binary LDPC
each will have
weights in field
[illegible]

$s_1h_1 + s_2h_2 + s_3h_3 = 0$ (PCM step)

GF(Q)

Assume $h_1 = h_2 = h_3 = 1$ → Might match for G but not in

$s_1 + s_2 + s_3 = 0$

G Gx = y
Hy = 0

(n, (n-k)) x = y
(1, n-k)(n, (n-k))
k → n
n → n·O₀

Transmit

$s_1 = 1$ (0,2)
$s_2 = 3$ (1,2) ← Input to channel
$s_3 = 2$ (0,2)

4
2,3
2/3

+ MGF

G → (k, n)
H → (n-k, n)

Read + Receive

For each symbol → R reads

R=3 → 3 sample from corresp. matrix, uniform at sampled
→ (2,2,0)
→ (1,1,1)
→ ~~(2,0,2)~~ (2,2,2)

3,2 2,1
(3,1)
(1,3)(3)
(n,n) (n,1) → (n)
(n,n,n) (n,n)
→ (n-1)

* Implementation → how to implement channel R?

Remove repeated elements

(2,0) {0,2} → (0,2) → {1}
(2) {1} → (1,2) (0,1) (3,0) → {0,3,4}
(0) {0} → (~~2~~,1) (2,1) ~~(2,1)~~ → {~~2~~,3,~~2~~}

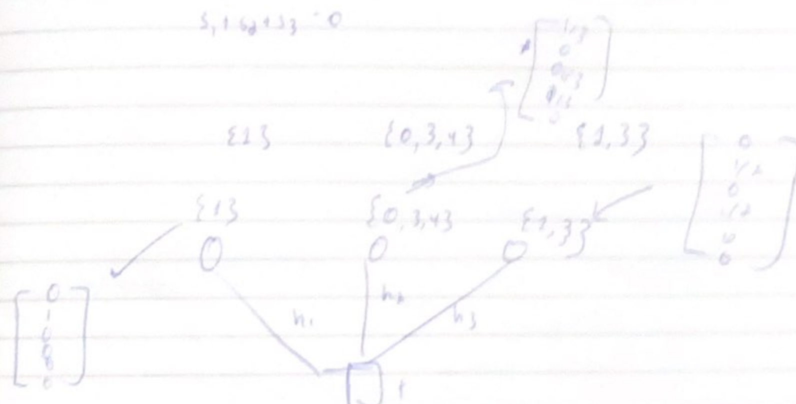
↓
Probability of each symbol being transmitted

Asymmetric
↓
some symbols are better protected

* Isn't there something we can do at encoding side to reduce this problem?

Possible Cleanup!

$s_1 + s_2 + s_3 = 0$



Q) For normal subset where → decoding always possible?

CN update to S_0

in: $\{0, 3, 4\}$ from S_2
$\{1, 3\}$ from S_3 ← might be faster to work w/ probabilities & convolutions

out: possible values of S_1
compatible w/ S_2 & S_3 & the SPC eqn

$\{4, 2, 1, 4, 0, 3\}$
↳ $\{4, 2, 1, 4, 0, 3\}$
$\{0, 1, 2, 3, 4, 5\}$ ← multiplicity equivalent to probab.
1 2 1 1 2

CN update to S_2
in $\{1\}$ from S_1
$\{1, 3\}$ from S_3

out. $\{3, 1\}$

CN update to S_3 ← multiplicity to guide decoder

in. $\{1\}$ from S_1
$\{0, 3, 4\}$ from S_2

→ may need to preserve multiplicity

out. $\{4, 1, 0\}$

VN update

~~$\{0, 1, 2\}$~~	$\{3, 1\}$	$\{4, 1, 0\}$
$\{4, 2, 1, 4, 0, 3\}$	$\{3\}$	$\{1\}$

{what does multiplicity of 4 do?}

$\{1\}$ $\{3\}$ $\{2\}$

↳ Shuffling through pairs blows up for a large system as opposed to Belief Propagation!! decoding

✓ Equivalent to BEC decoding

↓ Implementation is equivalent

↳ Repeat BP for BEC decoding

① Pick uniformly at random from each alphabet

(0,2)

R=3

0/2 0/2 0/2
↓ ↓ ↓
2 0 0

0/2 0/2 0/2
↓ ↓ ↓
0 0 0

Belief Propagation Turned Viewed