

## 15-853: Algorithms in the Real World

Error Correcting Codes III (expander based codes)

- Expander graphs
- Low density parity check (LDPC) codes
- Tornado codes

Thanks to Shuchi Chawla for many of the slides

15-853

Page1

## Why Expander Based Codes?

Linear codes like RS & random linear codes

The other two give nearly optimal rates

But they are slow :

Code	Encoding	Decoding
Random Linear	$O(n^2)$	$O(n^3)$
RS	$O(n \log n)$	$O(n^2)$
LDPC	$O(n^2)$ or better	$O(n)$
Tornado	$O(n \log 1/\epsilon)$	$O(n \log 1/\epsilon)$

Assuming an  $(n, (1-p)n, (1-\epsilon)pn+1)_2$  tornado code

15-853

Page2

## Error Correcting Codes Outline

Introduction

Linear codes

Read Solomon Codes

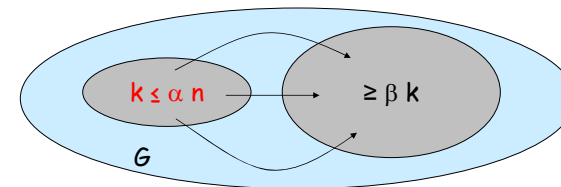
Expander Based Codes

- ➔ - Expander Graphs
- Low Density Parity Check (LDPC) codes
- Tornado Codes

15-853

Page3

## Expander Graphs (non-bipartite)



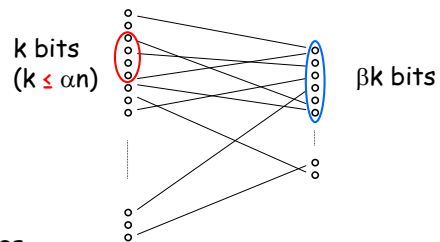
**Properties**

- **Expansion:** every small subset ( $k \leq \alpha n$ ) has many ( $\geq \beta k$ ) neighbors
- **Low degree** - not technically part of the definition, but typically assumed

15-853

Page4

## Expander Graphs (bipartite)



### Properties

- **Expansion:** every small subset ( $k \leq \alpha n$ ) on left has many ( $\geq \beta k$ ) neighbors on right
- **Low degree** - not technically part of the definition, but typically assumed

15-853

Page5

## Expander Graphs

Useful properties:

- Every set of vertices has many neighbors
- Every balanced cut has many edges crossing it
- A random walk will quickly converge to the stationary distribution (rapid mixing)
- The graph has "high dimension"
- Expansion is related to the eigenvalues of the adjacency matrix

15-853

Page6

## Expander Graphs: Applications

**Pseudo-randomness:** implement randomized algorithms with few random bits

**Cryptography:** strong one-way functions from weak ones.

**Hashing:** efficient n-wise independent hash functions

**Random walks:** quickly spreading probability as you walk through a graph

**Error Correcting Codes:** several constructions

**Communication networks:** fault tolerance, gossip-based protocols, peer-to-peer networks

15-853

Page7

## d-regular graphs

An undirected graph is **d-regular** if every vertex has d neighbors.

A bipartite graph is **d-regular** if every vertex on the left has d neighbors on the right.

The constructions we will be looking at are all d-regular.

15-853

Page8

## Expander Graphs: Eigenvalues

Consider the normalized adjacency matrix  $A_{ij}$  for an undirected graph  $G$  (all rows sum to 1)

The  $(x_i, \lambda_i)$  satisfying

$$A x_i = \lambda_i x_i$$

are the **eigenvectors** and **eigenvalues** of  $A$ .

Consider the eigenvalues  $\lambda_0 \geq \lambda_1 \geq \lambda_2 \geq \dots$

For a  $d$ -regular graph,  $\lambda_0 = 1$ . Why?

The separation of the eigenvalues tell you a lot about the graph (we will revisit this several times).

If  $\lambda_1$  is much smaller than  $\lambda_0$  then the graph is an expander.

Expansion  $\beta \geq (1/\lambda_1)^2$

15-853

Page9

## Expander Graphs: Constructions

Important parameters: **size** ( $n$ ), **degree** ( $d$ ), **expansion** ( $\beta$ )

### Randomized constructions

- A random  $d$ -regular graph is an expander with a high probability
- Construct by choosing  $d$  random perfect matchings
- Time consuming and cannot be stored compactly

### Explicit constructions

- Cayley graphs, Ramanujan graphs etc
- Typical technique - start with a small expander, apply operations to increase its size

15-853

Page10

## Expander Graphs: Constructions

Start with a small expander, and apply operations to make it bigger while preserving expansion

### Squaring

- $G^2$  contains edge  $(u,w)$  if  $G$  contains edges  $(u,v)$  and  $(v,w)$  for some node  $v$
- $A' = A^2 - 1/d I$
- $\lambda' = \lambda^2 - 1/d$
- $d' = d^2 - d$

Size	≡
Degree	↑
Expansion	↑

15-853

Page11

## Expander Graphs: Constructions

Start with a small expander, and apply operations to make it bigger while preserving expansion

### Tensor Product (Kronecker product)

- $G = A \times B$  nodes are  $(a,b) \quad \forall a \in A \text{ and } b \in B$
- edge between  $(a,b)$  and  $(a',b')$  if  $A$  contains  $(a,a')$  and  $B$  contains  $(b,b')$
- $n' = n_1 n_2$
- $\lambda' = \max(\lambda_1, \lambda_2)$
- $d' = d_1 d_2$

Size	↑
Degree	↑
Expansion	↓

15-853

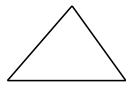
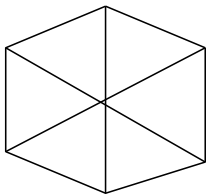
Page12

## Expander Graphs: Constructions

Start with a small expander, and apply operations to make it bigger while preserving expansion

Zig-Zag product

- "Multiply" a big graph with a small graph



$$\begin{aligned} n_2 &= d_1 \\ d_2 &= \sqrt{d_1} \end{aligned}$$

15-853

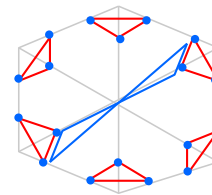
Page13

## Expander Graphs: Constructions

Start with a small expander, and apply operations to make it bigger while preserving expansion

Zig-Zag product

- "Multiply" a big graph with a small graph



Size	↑
Degree	↓
Expansion	↓ (slightly)

15-853

Page14

## Combination: square and zig-zag

For a graph with size  $n$ , degree  $d$ , and eigenvalue  $\lambda$ , define  $G = (n, d, \lambda)$ . We would like to increase  $n$  while holding  $d$  and  $\lambda$  the same.

Squaring and zig-zag have the following effects:

$$(n, d, \lambda)^2 = (n, d^2, \lambda^2) \quad \equiv \uparrow\uparrow$$

$$(n_1, d_1, \lambda_1) \text{ zz } (d_1, d_2, \lambda_2) = (n_1 d_1, d_2^2, \lambda_1 + \lambda_2 + \lambda_2^2) \quad \uparrow\downarrow\downarrow$$

Now given a graph  $H = (d^4, d, 1/5)$  and  $G_1 = (d^4, d^2, 2/5)$

$$- G_i = G_{i-1}^2 \text{ zz } H \quad (\text{square, zig-zag})$$

Giving:  $G_i = (n_i, d^2, 2/5)$  where  $n_i = d^{4i}$  (as desired)

15-853

Page15

## Error Correcting Codes Outline

Introduction

Linear codes

Read Solomon Codes

Expander Based Codes

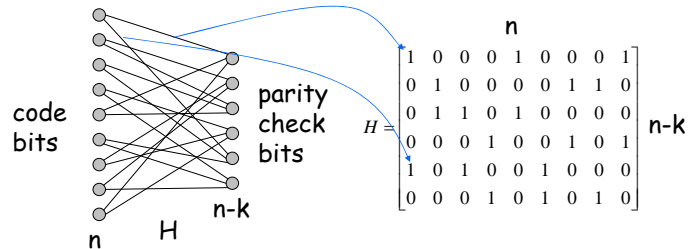
- Expander Graphs
- Low Density Parity Check (LDPC) codes
- Tornado Codes



15-853

Page16

## Low Density Parity Check (LDPC) Codes



Each row is a vertex on the right and each column is a vertex on the left.  
 A codeword on the left is valid if each right "parity check" vertex has parity 0.  
 The graph has  $O(n)$  edges (**low density**)

15-853

Page17

## Applications in the "real world"

10Gbase-T (IEEE 802.3an, 2006)

- Standard for 10 Gbits/sec over copper wire

WiMax (IEEE 802.16e, 2006)

- Standard for medium-distance wireless.  
 Approx 10Mbits/sec over 10 Kilometers.

NASA

- Proposed for all their space data systems

15-853

Page18

## History

Invented by Gallager in 1963 (his PhD thesis)

Generalized by Tanner in 1981 (instead of using parity and binary codes, use other codes for "check" nodes).

Mostly forgotten by community at large until the mid 90s when revisited by Spielman, MacKay and others.

15-853

Page19

## Distance of LDPC codes

Consider a  $d$ -regular LPDC with  $(\alpha, 3d/4)$  expansion.

**Theorem:** Distance of code is greater than  $\alpha n$ .

**Proof.** (by contradiction)

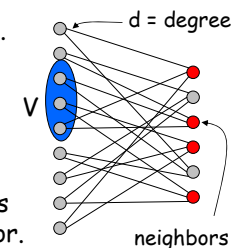
Assume a codeword with weight  $v \leq \alpha n$ .

Let  $V$  be the set of 1 bits in codeword

It has  $>3/4dv$  neighbors on the right

Average # of 1s per such neighbor is  $< 4/3$ .

To make average work, at least one has only 1 bit...which would cause an error.



15-853

Page20

## Correcting Errors in LPDC codes

We say a vertex is *unsatisfied* if parity  $\neq 0$

### Algorithm:

While there are unsatisfied check bits

1. Find a bit on the left for which more than  $d/2$  neighbors are unsatisfied
2. Flip that bit

Converges since every step reduces unsatisfied nodes by at least 1.

Runs in linear time.

Why must there be a node with more than  $d/2$  unsatisfied neighbors?

15-853

Page21

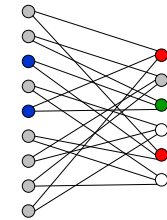
## Coverges to closest codeword

**Theorem:** If # of error bits is less than  $\alpha n/4$  with  $3d/4$  expansion then the simple decoding algorithm will coverge to the closest codeword.

**Proof:** let:

- $u_i$  = # of unsatisfied check bits on step  $i$
- $r_i$  = # corrupt code bits on step  $i$
- $s_i$  = # satisfied check bits with corrupt neighbors on step  $i$

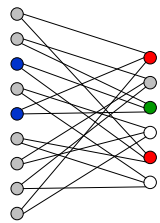
We know that  $u_i$  decrements on each step, but what about  $r_i$ ?



15-853

Page22

## Proof continued:



- $u_i$  = unsatisfied
- $r_i$  = corrupt
- $s_i$  = satisfied with corrupt neighbors

$$u_i + s_i > \frac{3}{4} dr_i \quad (\text{by expansion})$$

$$2s_i + u_i \leq dr_i \quad (\text{by counting edges})$$

$$\frac{1}{2} dr_i \leq u_i \quad (\text{by substitution})$$

$$u_i < u_0 \quad (\text{steps decrease } u) \quad u_0 \leq dr_0 \quad (\text{by counting edges})$$

**Therefore:**  $r_i < 2r_0$  i.e. number of corrupt bits cannot more than double

If we start with at most  $\alpha n/4$  corrupt bits we will never get  $\alpha n/2$  corrupt bits but the distance is  $\alpha n$

15-853

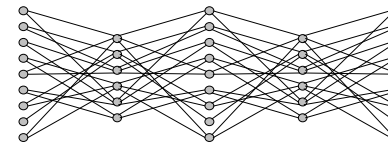
Page23

## More on decoding LDPC

Simple algorithm is only guaranteed to fix half as many errors as could be fixed but in practice can do better.

Fixing  $(d-1)/2$  errors is NP hard

Soft "decoding" as originally specified by Gallager is based on belief propagation---determine probability of each code bit being 1 and 0 and propagate probs. back and forth to check bits.



15-853

Page24

## Encoding LPDC

Encoding can be done by generating  $G$  from  $H$  and using matrix multiply.

What is the problem with this?

Various more efficient methods have been studied

15-853

Page25

## Error Correcting Codes Outline

**Introduction**

**Linear codes**

**Read Solomon Codes**

**Expander Based Codes**

- Expander Graphs
- Low Density Parity Check (LDPC) codes
- Tornado Codes



15-853

Page26

## The loss model

Random Erasure Model:

- Each bit is lost independently with some probability  $\mu$
- We know the positions of the lost bits

For a **rate** of  $(1-p)$  can correct  $(1-\epsilon)p$  fraction of the errors.

Seems to imply a

$$(n, (1-p)n, (1-\epsilon)pn+1)_2$$

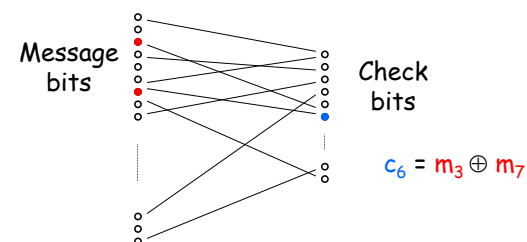
code, but not quite because of random errors assumption.

We will assume  $p = .5$ .

Error Correction can be done with some more effort

15-853

Page27



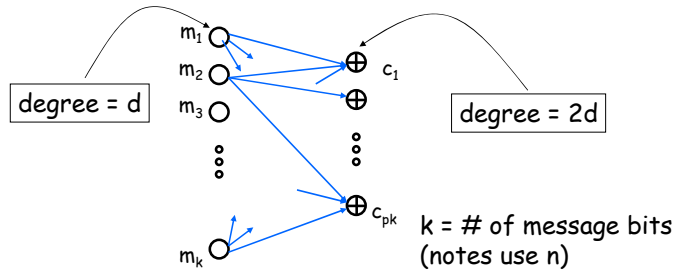
Similar to LDPC codes but check bits are not required to equal zero (i.e the graph does not represent  $H$ ).

15-853

Page28

## Tornado codes

Will use  $d$ -regular bipartite graphs with  $n$  nodes on the left and  $pn$  on the right (notes assume  $p = .5$ )  
 Will need  $\beta > d/2$  expansion.

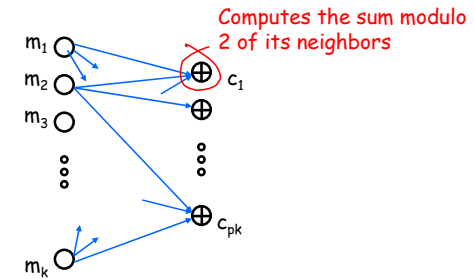


15-853

Page29

## Tornado codes: Encoding

Why is it linear time?

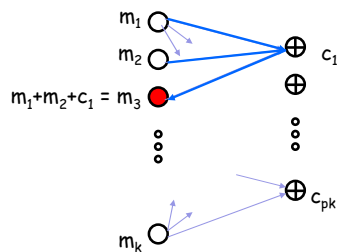


15-853

Page30

## Tornado codes: Decoding

Assume that all the check bits are intact  
 Find a check bit such that only one of its neighbors  
 is erased (an *unshared neighbor*)  
 Fix the erased code, and repeat.

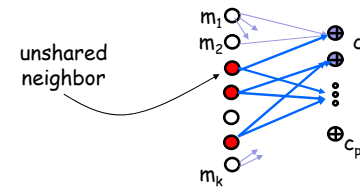


15-853

Page31

## Tornado codes: Decoding

Need to ensure that we can always find such a check bit  
 "Unshared neighbors" property  
 Consider the set of corrupted message bit and their  
 neighbors. Suppose this set is small.  
 $\Rightarrow$  at least one message bit has an unshared neighbor.



15-853

Page32

## Tornado codes: Decoding

Can we always find unshared neighbors?

Expander graphs give us this property if  $\beta > d/2$   
(see notes)

Also, [Luby et al] show that if we construct the graph from a specific kind of degree sequence, then we can always find unshared neighbors.

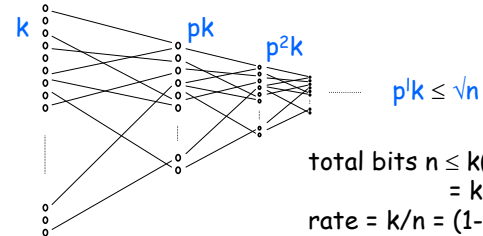
15-853

Page33

## What if check bits are lost?

### Cascading

- Use another bipartite graph to construct another level of check bits for the check bits
- Final level is encoded using RS or some other code



$$\begin{aligned} \text{total bits } n &\leq k(1 + p + p^2 + \dots) \\ &= k/(1-p) \\ \text{rate} &= k/n = (1-p) \end{aligned}$$

15-853

Page34

## Cascading

### Encoding time

- for the first  $k$  stages :  $|E| = d \times |V| = O(k)$
- for the last stage:  $\sqrt{k} \times \sqrt{k} = O(k)$

### Decoding time

- start from the last stage and move left
- again proportional to  $|E|$
- also proportional to  $d$ , which must be at least  $1/\epsilon$  to make the decoding work

Can fix  $kp(1-\epsilon)$  random erasures

15-853

Page35

## Some extra slides

15-853

Page36

## Expander Graphs: Properties

Prob. Dist. -  $\pi$  ; Uniform dist. -  $u$

Small  $|\pi - u|$  indicates a large amount of "randomness"

Show that  $|A\pi - u| \cdot \lambda_2 |\pi - u|$

Therefore small  $\lambda_2 \Rightarrow$  fast convergence to uniform

Expansion  $\beta \frac{1}{4} (1/\lambda_2)^2$

15-853

Page37

## Expander Graphs: Properties

To show that  $|A\pi - u| \cdot \lambda_2 |\pi - u|$

Let  $\pi = u + \pi'$

$u$  is the principle eigenvector

$$Au = u$$

$\pi'$  is perpendicular to  $u$

$$A\pi' \cdot \lambda_2 \pi'$$

So,  $A\pi = u + \lambda_2 \pi'$

Thus,  $|A\pi - u| \cdot \lambda_2 |\pi'|$

15-853

Page38