ZETA: A Novel Network Coding Based P2P Downloading Protocol

- Jing Wang
- Department of Electronis and Information Engineering
- Huazhong University of Science and Techonology

Background

- P2P Downloading Protocol
  - Napster
  - Gnutella
  - BT
  - Emule
  - Avanlanche
- My work is based on Zeta and Avalanche

Existing P2P protocol

- BT Protocol
  - Accounts For 27-53 % of the total Internet Traffic
  - The most Important Feature is Slicing
- Avalanche
  - Developed by Microsoft Research Cambridge
  - First proposed in 2005, but has little influence
  - The most important Feature is Network Coding

Comparison

	Server-Client Downloading	BT	Avalanche
Cute File into Pieces?	No	Yes	Yes
Transportation Protocol	TCP UDP	TCP	TCP
Users Number VS. Downloading Speed	Negatively Related	Positively Related	Positively Related
Source Number	Single	Multi	Multi
Network Coding	No	No	Yes
Calucation Overhead	No	Small	Large
Application	Widely applied	Widely applied	Rare
PUSH or PULL	either	PULL	PULL

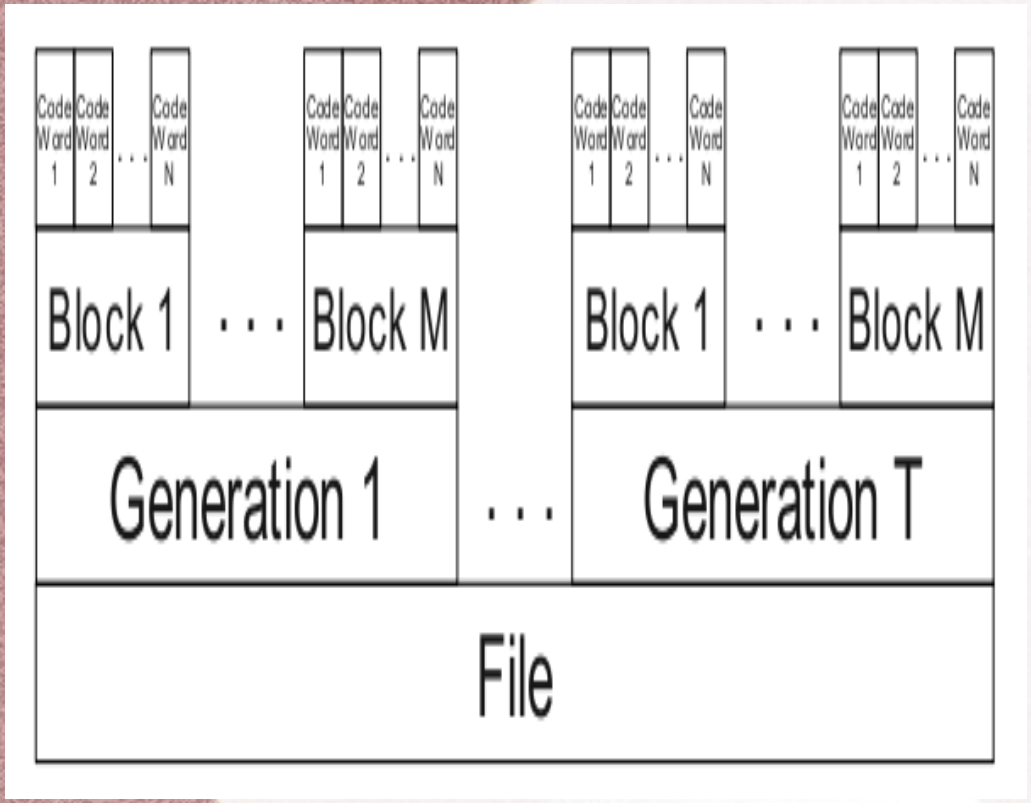
New Features

- Why Avanlanche is not widely applied?
  - Caculation Overhead is two large.
  - Lack mature practical software or simulation tools.
  - Closed Source Policy of Micorsoft, Lack documentation
- New Feature should be added.
  - Reduce Calculation Overhead
  - Reduce Other Overhead
  - Develop a simulation suite to optimize parameters

Zeta Protocol

- PULL----> PUSH
  - Reducing Overhead of message exchange
- Decoding at last ----> Progressive Decoding
  - Reducing the calclation overhead
- Separate Boot Management and Download management
  - Make the network more robust

File Structure



S_p Is the maximum packet size of a Zeta Packet

S_g^{max} Is the maximum size of a generation

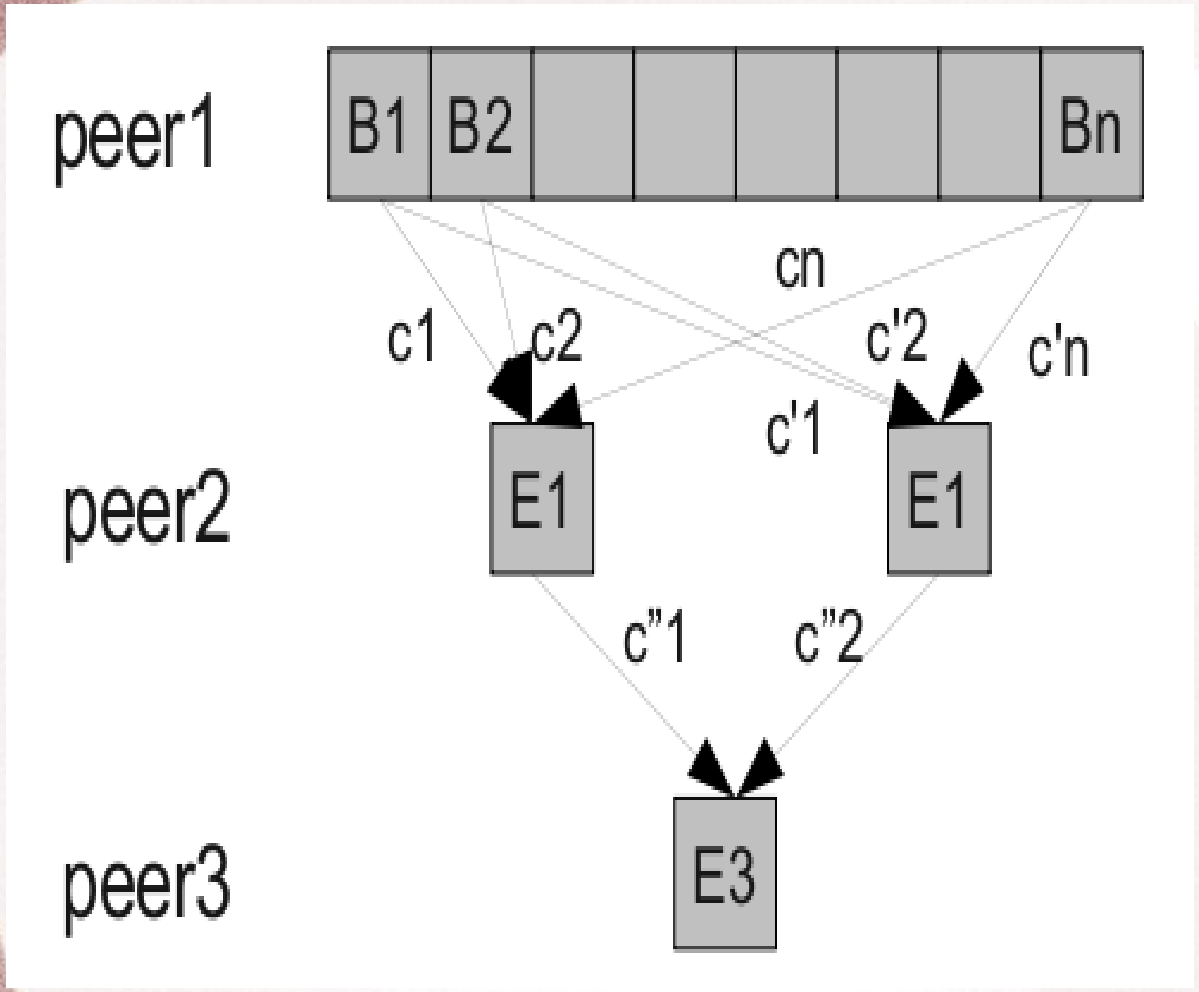
S_w Is the size of a code word

$$S_g^{max} = \frac{S_p^2}{2S_w}$$

To avoid slicing, S_p had better to be a little smaller than MTU
MTU in Ethernet is 1500 Byte. Let S_w be 1

$$S_g^{max} = 1.125MB$$

Reduce Overhead

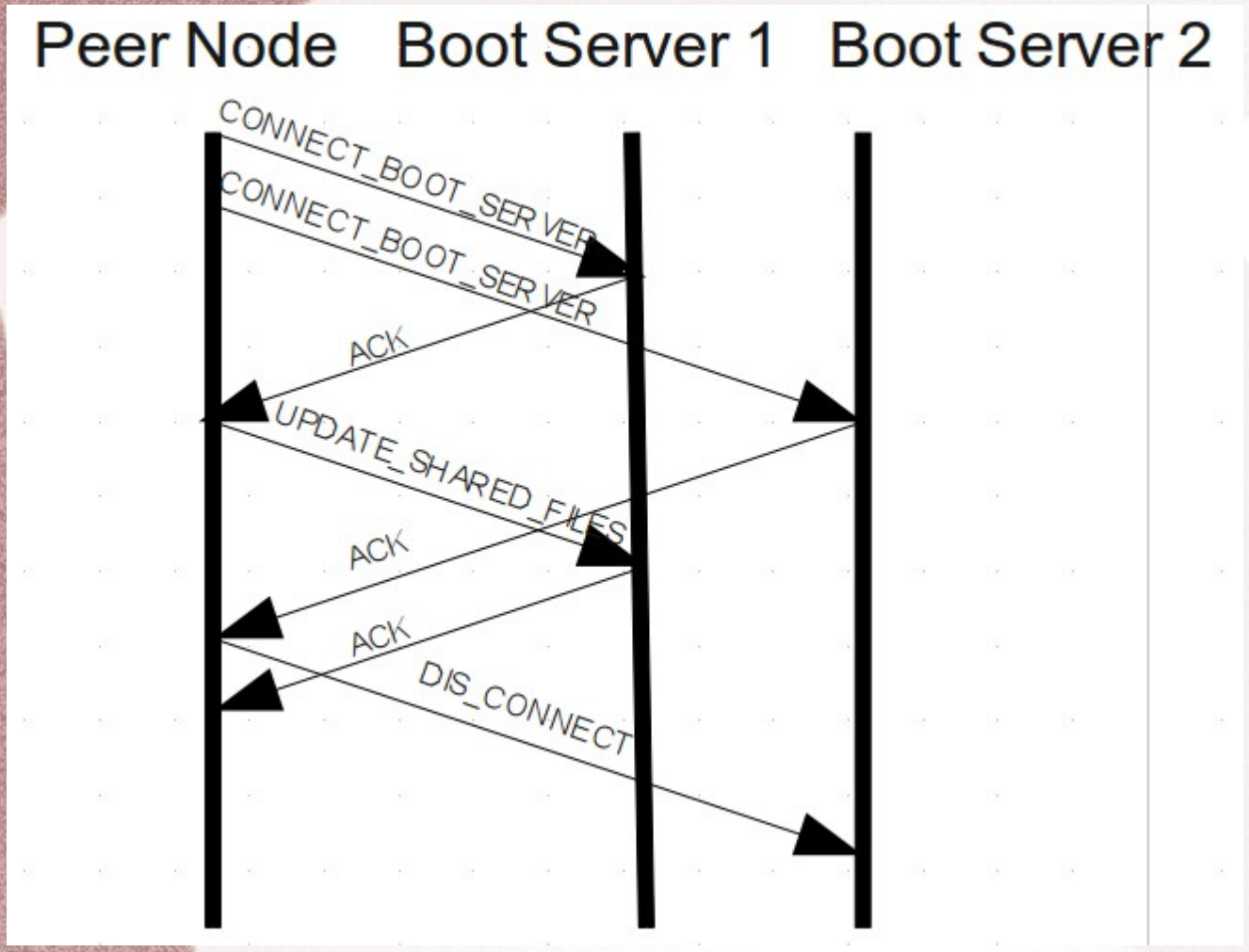

peer1
B1
B2
Bn
cn
c1
c2
c'2
c'n
c'1
peer2
E1
E1
c"1
c"2
peer3
E3

Node Type

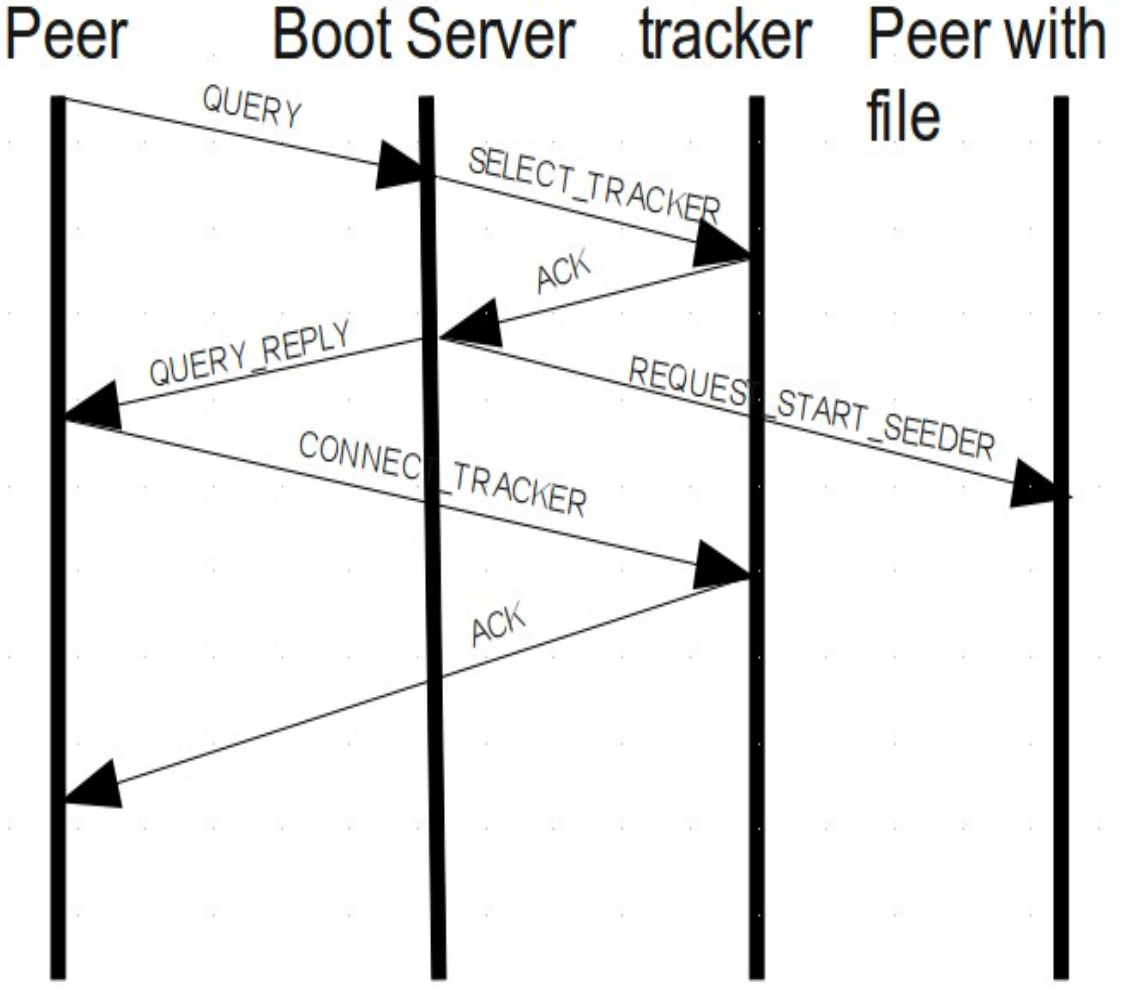
- Boot Server
  - Boot Management
  - Store the corresponding relation between tracker server and file
- Tracker
  - In charge of the download of a specific file
  - Return Peer Set
- Peer Node
  - Downloader
  - Seeder & Downloader
  - Seeder

Comparison With .torrent file

Boot Process



Query Process



Comparison with BT

1. Dynamicly Assign the tracker. Avoid invalid torrent file

2. Tracker need not to open for all the time. User can request boot server to select new tracker

2. Make uploading easier

File Distribution

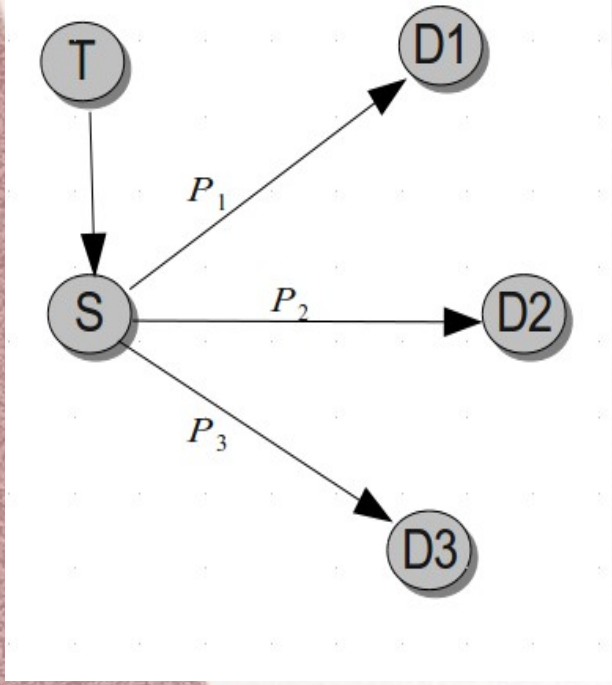
- Seeder Periodically Send UPDATE_PEERS message to tracker to request information of its neighboring downloaders.
- Downloader periodically update its own information in tracker.
- Use PUSH scheme instead of PULL scheme
- Seeders deduce the needs of downloaders and send data actively

Why it is impossible in BT?
The benefis of Netwok Coding

Sending And Receiving Data

When?

$$S_p = S_{cv} + S_b \qquad T_i = \frac{S_p}{B_i}$$



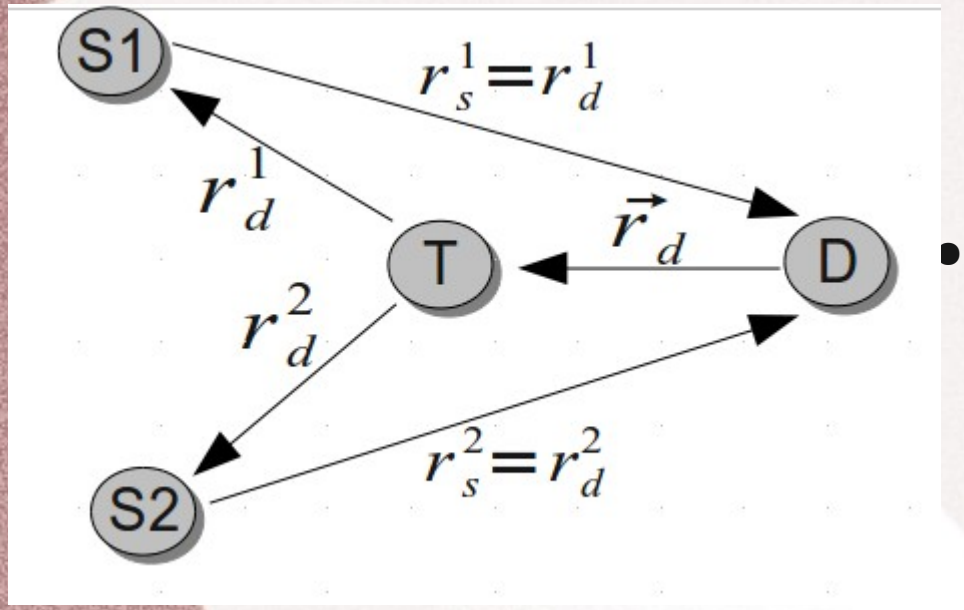
To Whom

- 1. Historical Record H_i^j
- 2. Instant Rate B_j
- 3. Distance $D_i^j = |IP_i - IP_j|$

Reason

$$P_i^j = \frac{H_i^j \times B_j / D_i^j}{\sum_{j \in S_d} P_j^i}$$

Flow Control in Zeta



- Zeta Protocol runs on UDP protocol, so it must contains a flow control scheme.
- The flow control schems is simple but enough
  - Zeta is not sensitive with packet drop
  - Zeta use PUSH scheme

Random Network Coding

- Encoding
  - In Source Node

$$x_j = \sum_{i=1}^{m} c_{ji} \cdot b_i$$

  - In Average Seeder

$$x'_j = \sum_{i=1}^{t} c'_{ji} \cdot x_i$$

$$c_j'' = \sum_{i=1}^{t} c_{ji} \cdot c_i$$

- Progressive Decoding
  - Store the Coefficient Matrix as a Upper Triangle Matrix. Add a block means add a row

$$\left(\begin{array}{ccc|cc} 1 & 2 & 3 & 4 & 3 \\ 0 & 5 & 6 & 5 & 4 \\ 0 & 0 & 9 & 8 & 6 \end{array}\right)$$

Existing matrix
BlockNumInData = 5
Have received 3 Linearly Independent blocks

$$\begin{pmatrix} 1 & 2 & 3 & 0 & 0 \end{pmatrix}$$

Coefficient of dependent Blocks

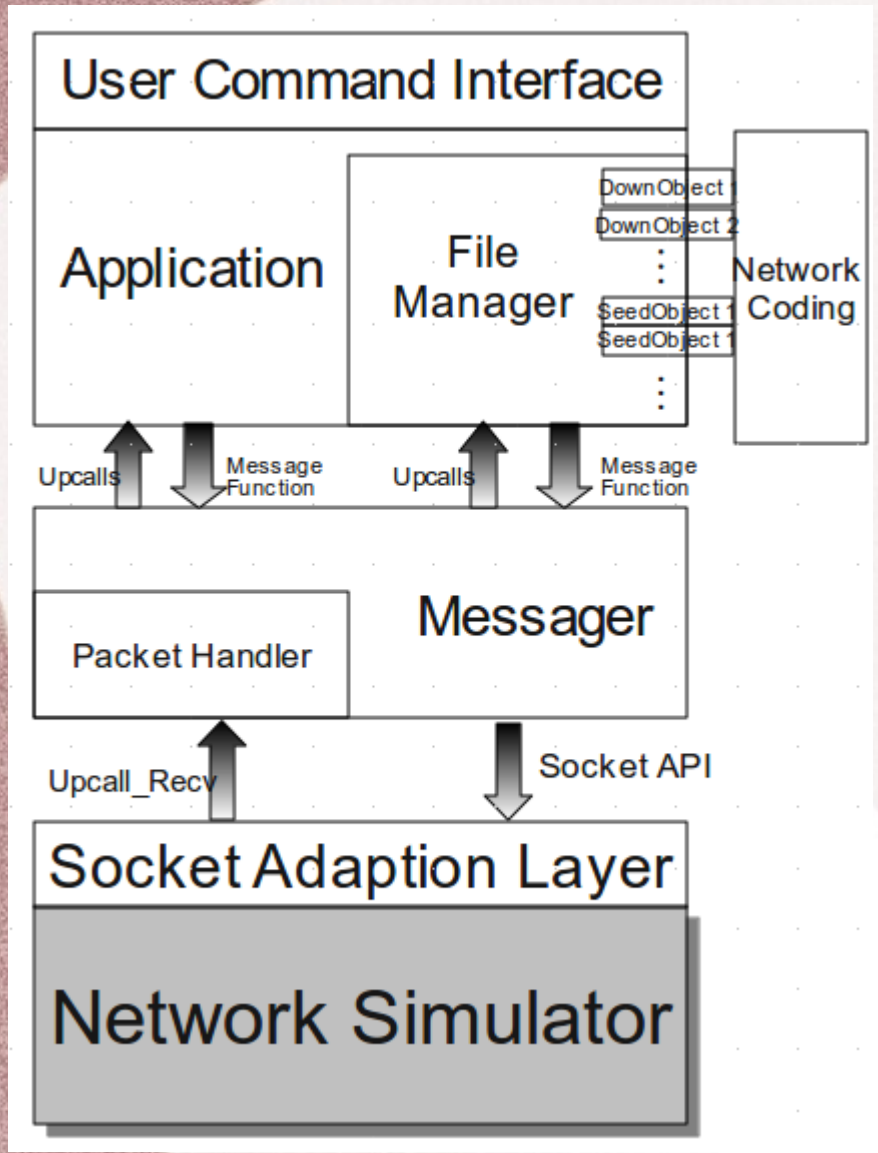
Only 2 Comparisons

$$\begin{pmatrix} 1 & 2 & 3 & 1 & 2 \end{pmatrix}$$

Coefficient of dependent Blocks

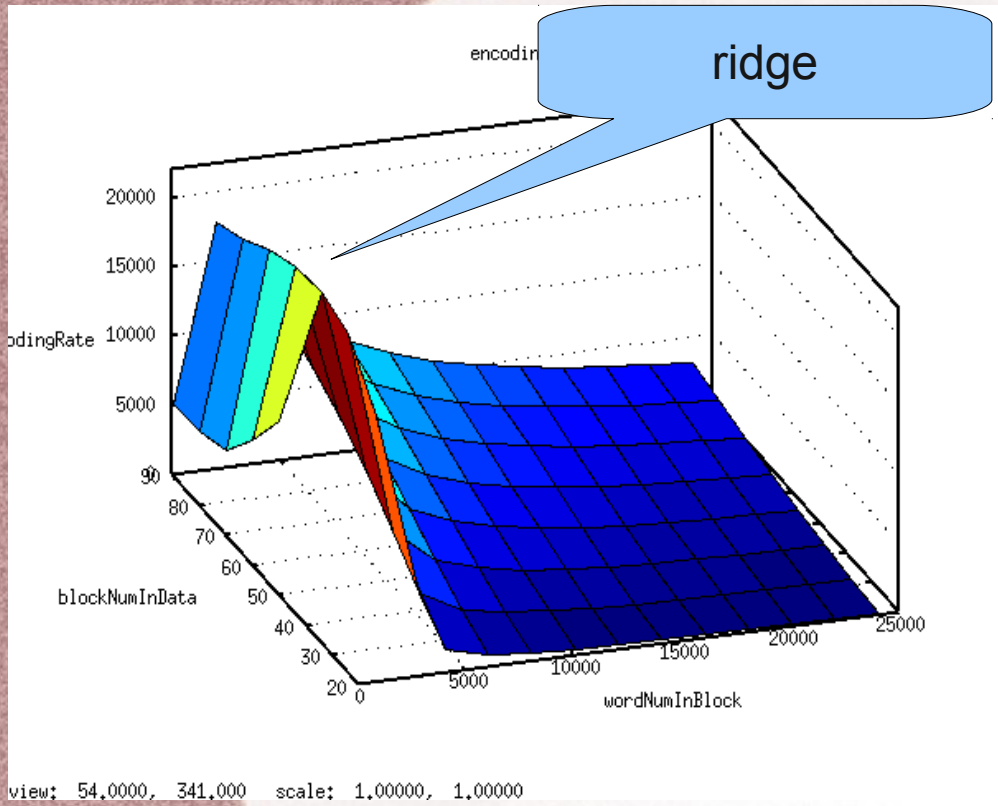
If blockNumInData is n, existing linearly independent blocks is k, then
Check dependenence requires only n-k comparisons
Progressive decoding requires only $\frac{(2n-k)\times k}{2}$ muliplications and addistions

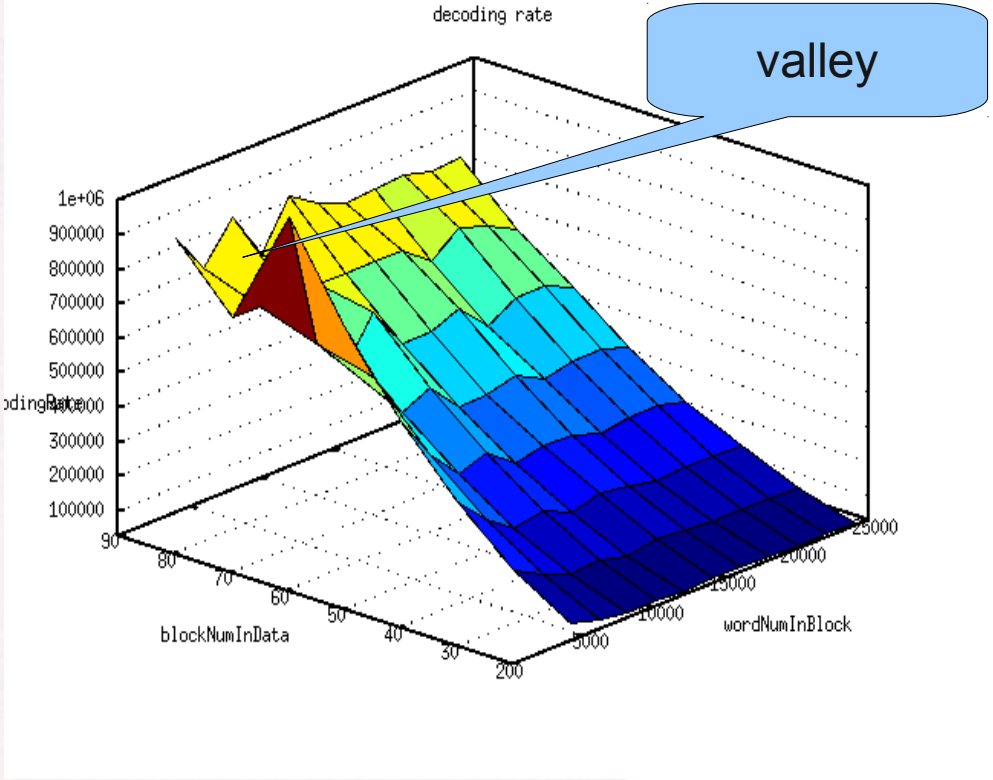
ZetaSim



- Three layer architerture
- Well structured and scalable
- Portable. Can run on any protocol that can provide Socket API. With little code revision, it can be transformed into a practical software

Simulation of Network Coding Parameters





Explanation

Three Conclusions:

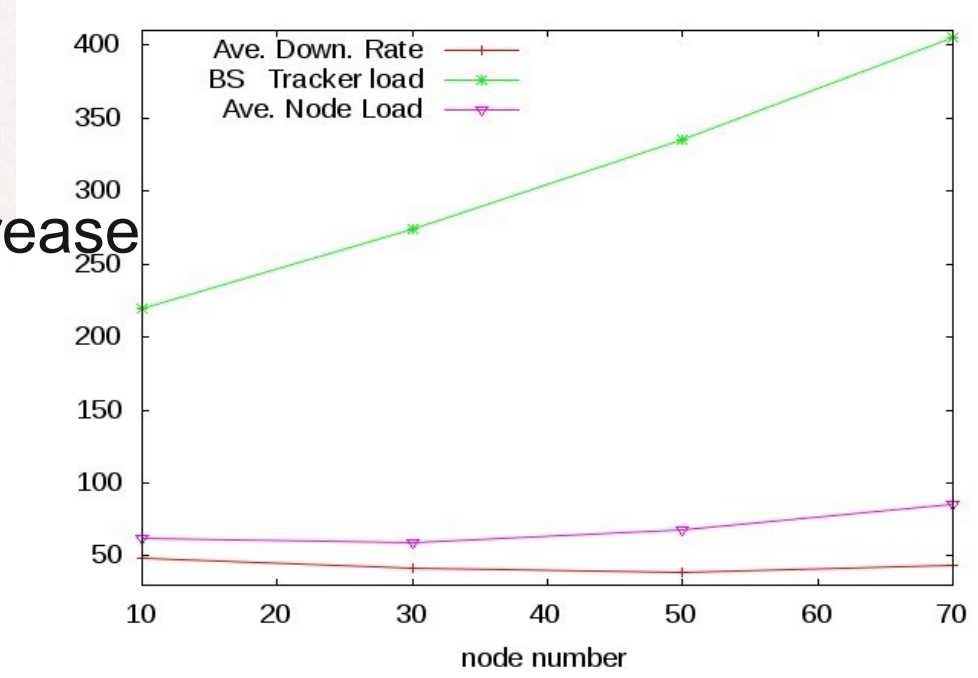
- Encoding Rate is not sensitve to block Num in Data
- Decoding Rate is not sensitive to word Num in Block
- We must find a tradeoff between encoding rate and decoding rate

Simulation In Full Connected Topology

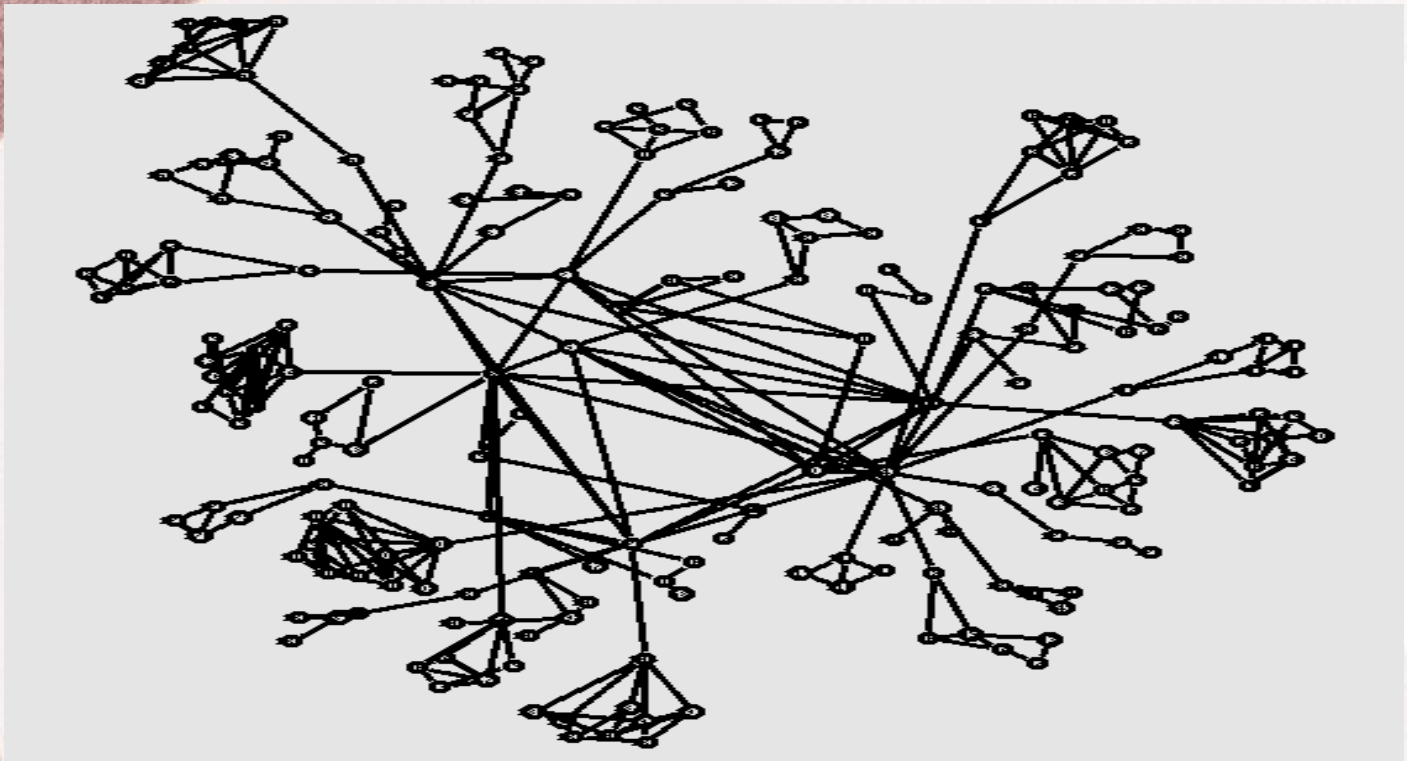
Node Number	Ave. Down Rate(KB/s)	Ave. Query Time(s)	Boot Server & Tracker Load(KB/s)	Average Node Load(KB/s)
10	48.503	0.03400	219.3	61.98
30	41.198	0.03408	274.06	59
50	39.13	0.03654	335.2	68
70	43.5823	0.03815	405	85

With the increase of node size:

- The Average downloading rate increase
- The load of peer node keep stable
- The load of boot server increase



Simulation In Transit-Stub Network



Node Number	Ave. Down Rate(KB/s)	Ave. Query Time(s)	Boot Server & Tracker Load(KB/s)	Average Node Load(KB/s)
100	245.3269	5.4141	1325.85	320.45

Future Work

- Improve Flow Control Scheme
- Implement a practical sofware and test the performance in real Internet structure

Thank You

Q & A