Router Caching -aided VoD Systems

Xin Wang

School of Computer Science

Fudan University

October 3, 2012

Streaming System for Video on Demand Applications

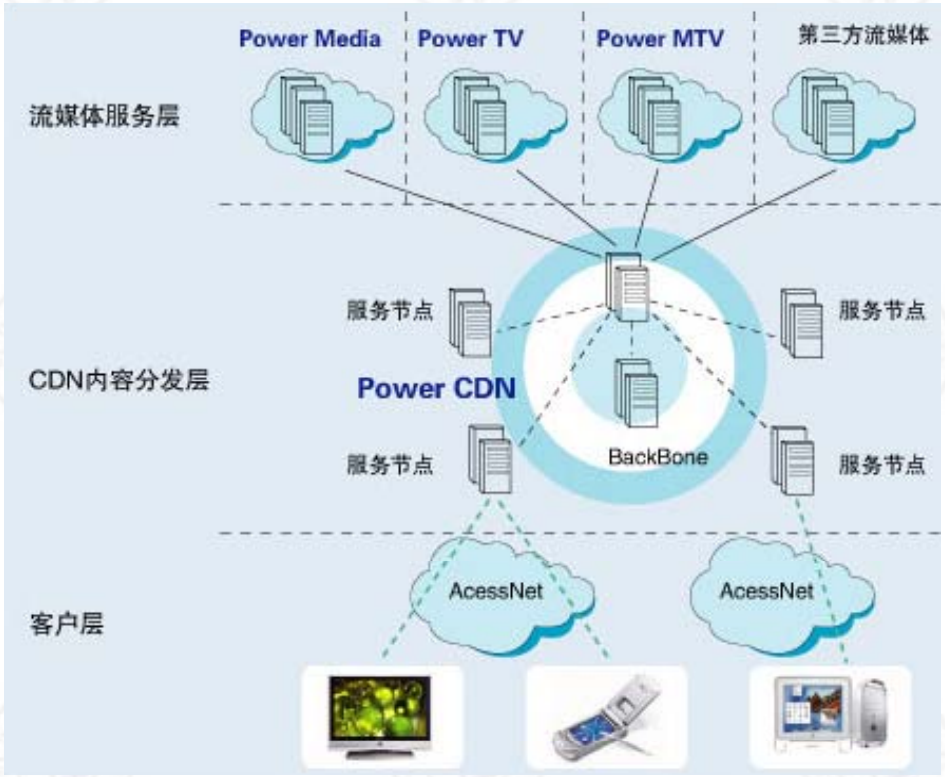
Features

- Large data
- Long duration
- Delay
- Jitter
- Packet loss
- Bandwidth

Requirements

- Large capability
- Reliability
- Security
- Economy

VoD Streaming Systems with CDN

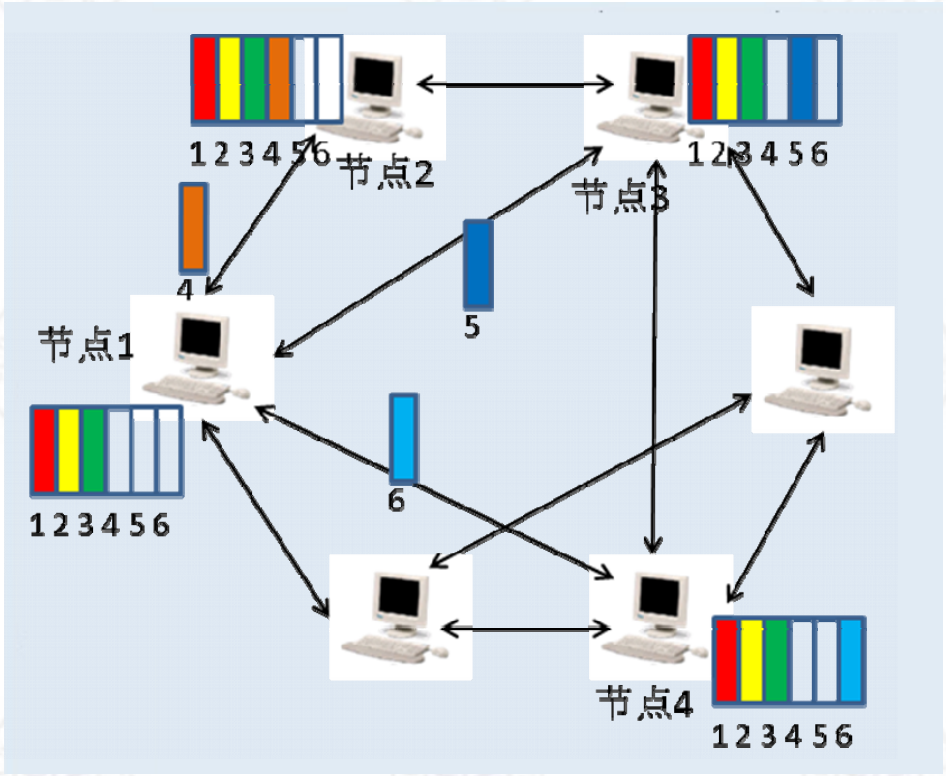


- Push hot content to edge cache nodes near to the user
- Key idea: avoid potential jams or delays

✓ For user
  - Fast response

✓ For service provider
  - Less pressure on sever
  - Less bandwidth on the backbone

P2P VoD Streaming Systems



- P2P v.s. C/S
- Each peer is the same to each other
- More user involved in the content distribution

✓ For service Provider
- Reduce server load sharply

✓ For ISP
- Less bandwidth on backbone

Challenges on CDN and P2P

For CDN

1）Limitation on the CDN architecture

2）High cost to scale up and maintain

3）No insurance on user experience

For P2P

1）Reliability: peers can leave any time

2）Security：peers may overheard and defraud

3）Manageability：can not control peers

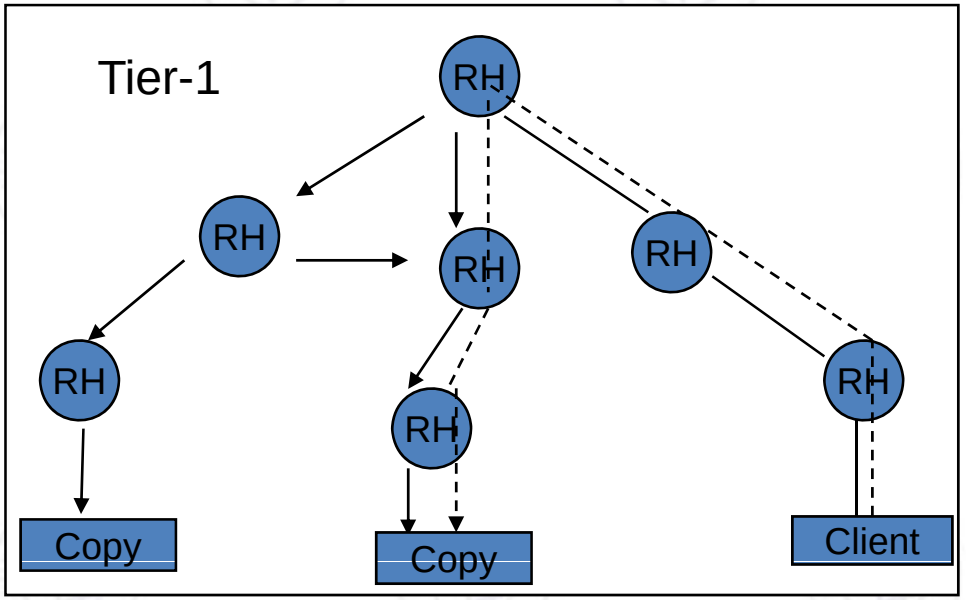
Content Centric Network

Name of Content：**P:L**

Content： **triplet<data, public key, signature>**

Publish and Search：**REGISTER（P:L） FIND（P:L）**

- ✓ Be compatible to existing networks and interfaces
- ✓ Content centric
- ✓ Content can be cached around the network, accessible any time and anywhere
- ✓ Content can be encrypted, searched by name or description



Our Contribution

CDN from the server aspects

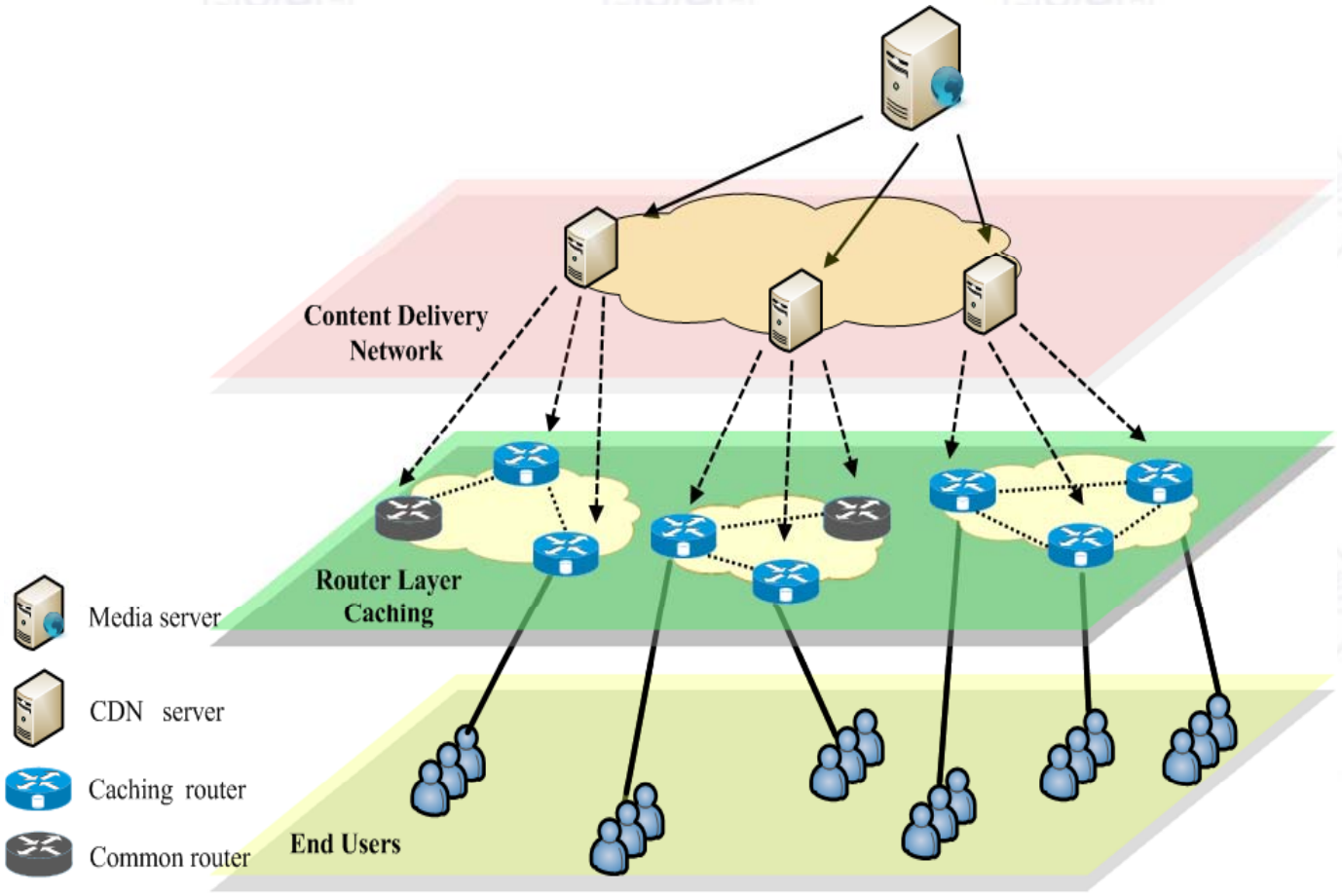
P2P from the client aspects

Router Ignored

CCN uses the whole network

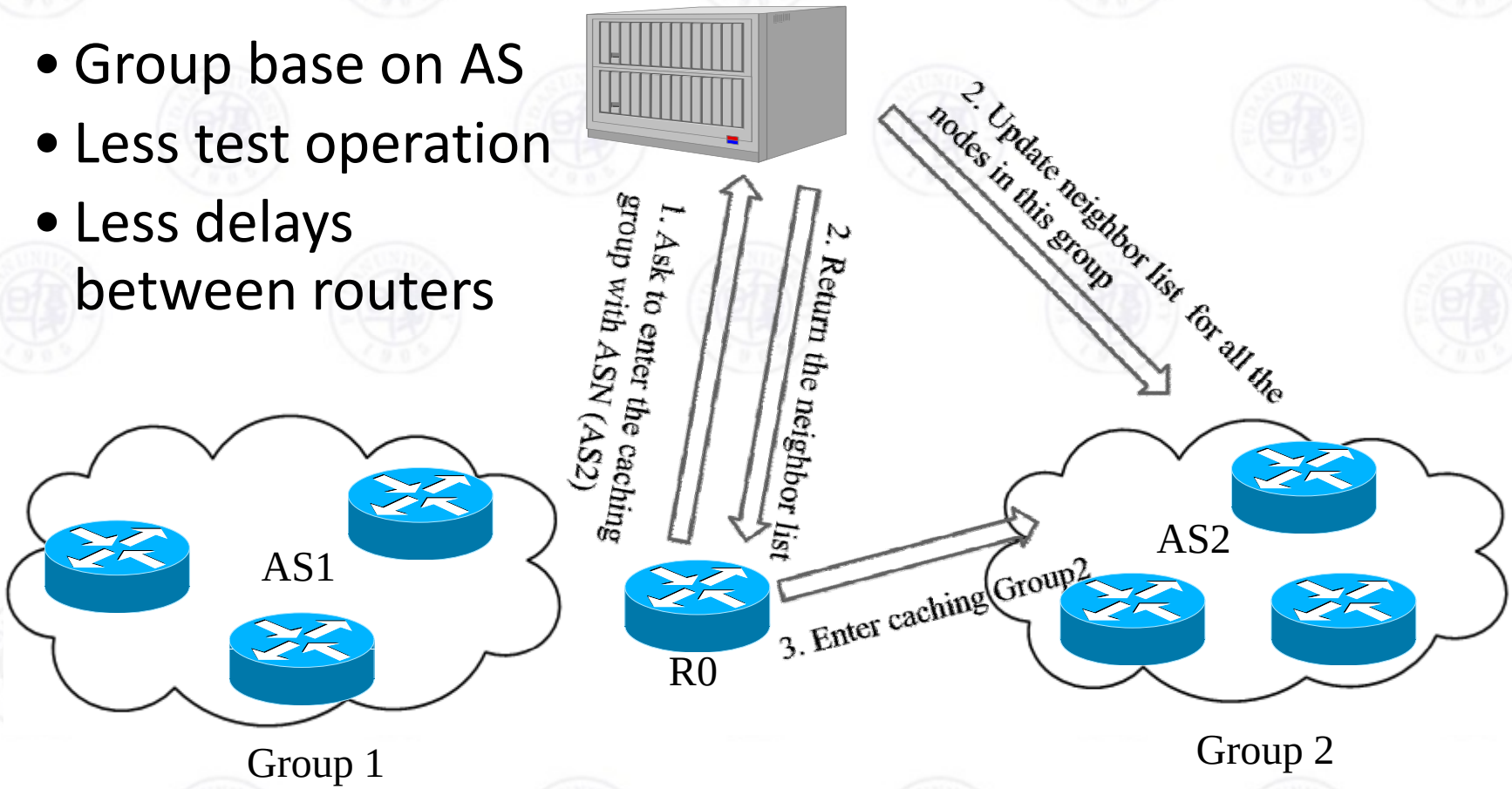
Router Aided VoD Streaming Systems

Architecture



Topology Structure of the Overlay Network

- Group base on AS
- Less test operation
- Less delays between routers



Efficiency of Router Aided Cache

- Requests distribution according to content's popularity
- Router can store *m* unit
- *s* : state of router

Hitting probability on one router

$$h = \sum_{s} (P_s \sum_{i=1}^{m} p_i)$$

Total hitting probability on *n* routers:

$$H = h_1 + (1-\alpha_2)h_2 + \ldots + (1-\alpha_i)h_i + \ldots + (1-\alpha_n)h_n$$

$$H = \sum_{i=1}^{n} h_i - \sum_{i=2}^{n} (\alpha_i h_i)$$

α_i for the *i*th router's overlapping ratio with $1 \sim i-1$ routers, $0 \le \alpha_i \le 1$

Random Linear Network Coding

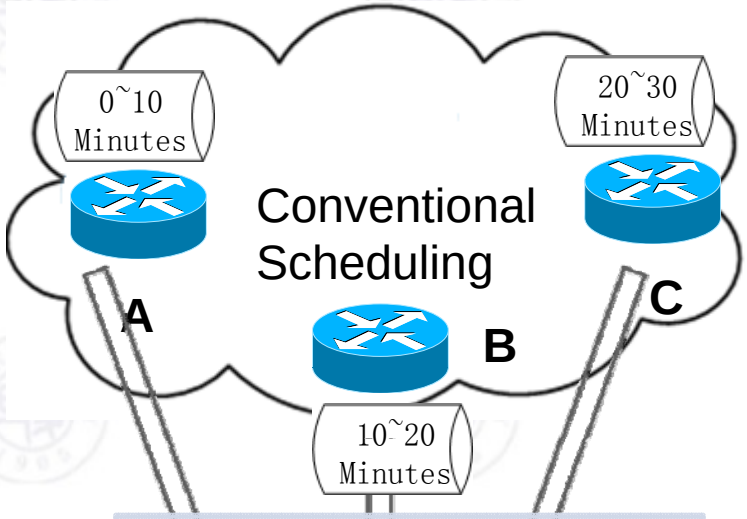
- Data are divided in to *k blocks* $[b_1, b_2, \ldots, b_k]$
- Randomly choose a coefficient $[c_1, c_2, \ldots, c_k]$ from field *GF(t), coded as one block*

$$x = \sum_{i=1}^{k} c_i \cdot b_i$$

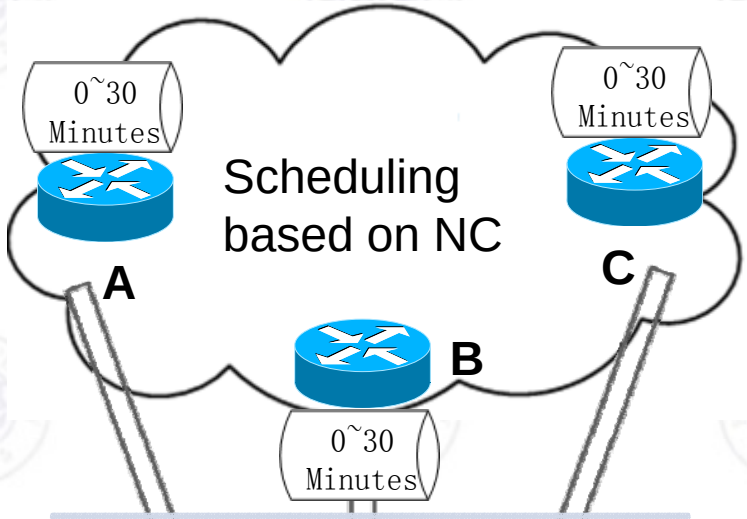
Decoding: $[b_1, b_2, \ldots, b_k] = A^{-1} X^T$

where $X^T = [x_1, x_2, \ldots, x_k]^T$, $x_1, x_2, \ldots, x_k$ for any *k* coded blocks, A^{-1} is the matrix composed of the coefficients of the *k* coded blocks

Scheduling Policy



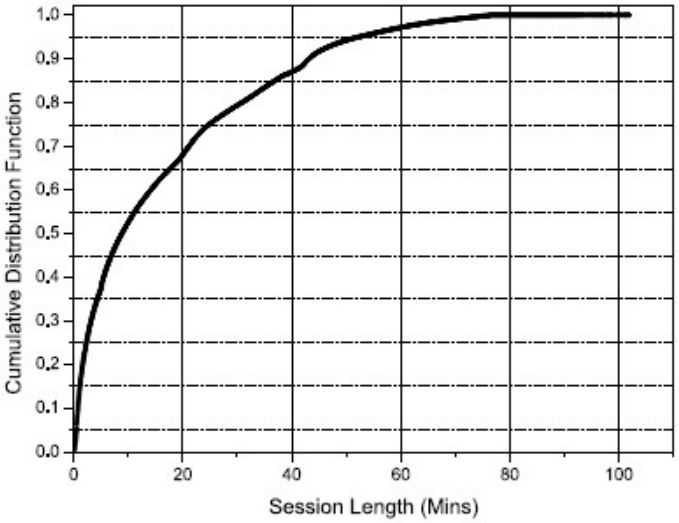
- Different routers for different intervals
- Needs coordination or centered control

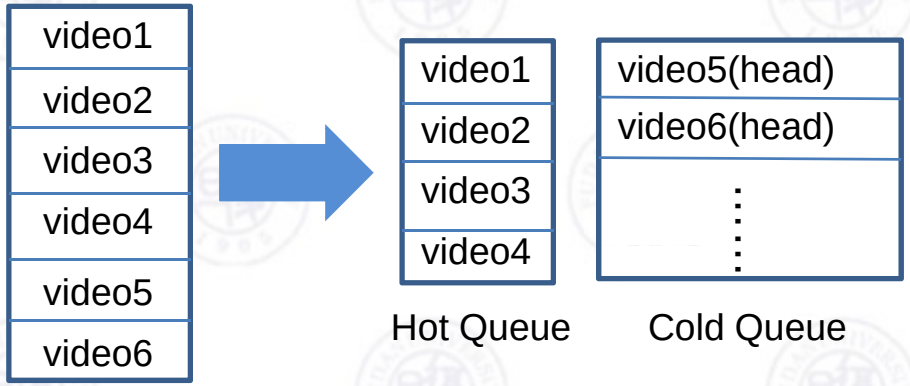


- Routers have the same amount of information
- Concurrent transmission
- Simple schedule policy

Replacement Policy

Session length	percentage
5 min	37.44%
10 min	52.55%
25 min	75.25%
50 min	94.23%





Conventional Caching policy

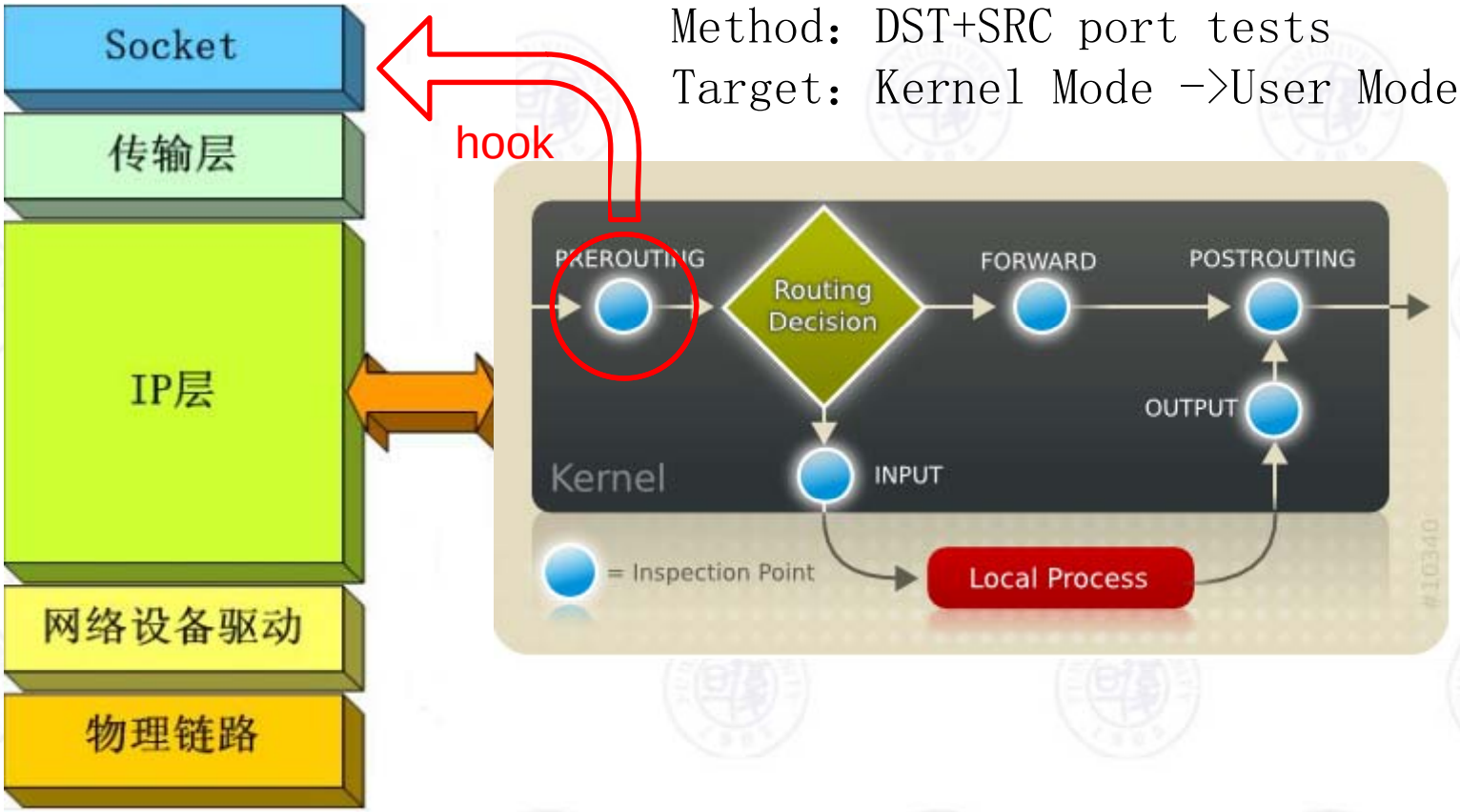
- Based on popularity
- Cache the whole video with high popularity
- Simple

Hot Queue、cold Queue

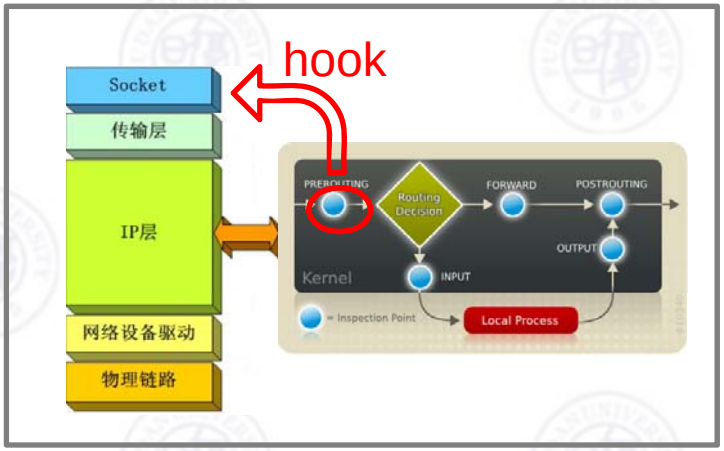
- Hot Queue to cache the whole video
- Cold Queue to cache the beginning parts of the video
- Reduce response time

Router Aided Caching Policy: Implementation

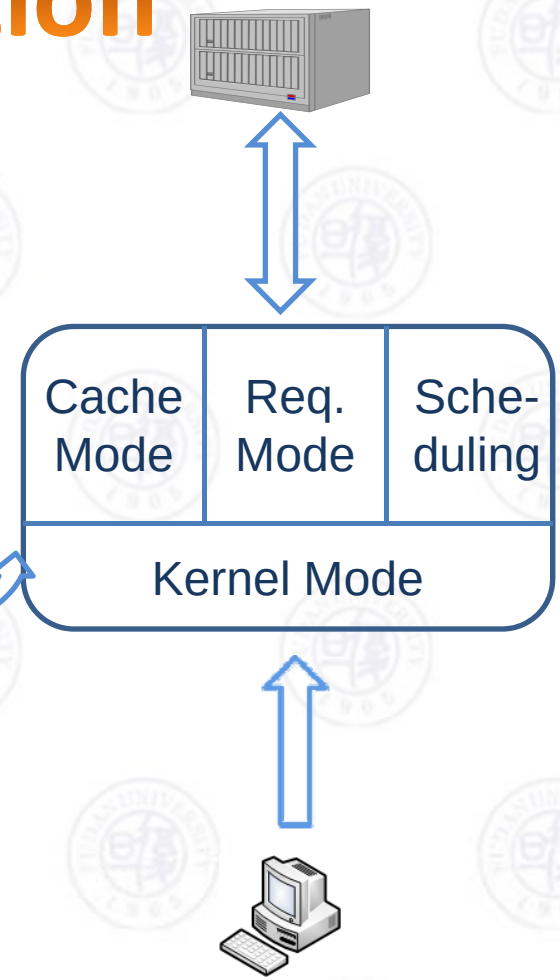
Method: DST+SRC port tests
Target: Kernel Mode ->User Mode



Router Aided Caching Policy: Implementation



Transparent Caching on Routers

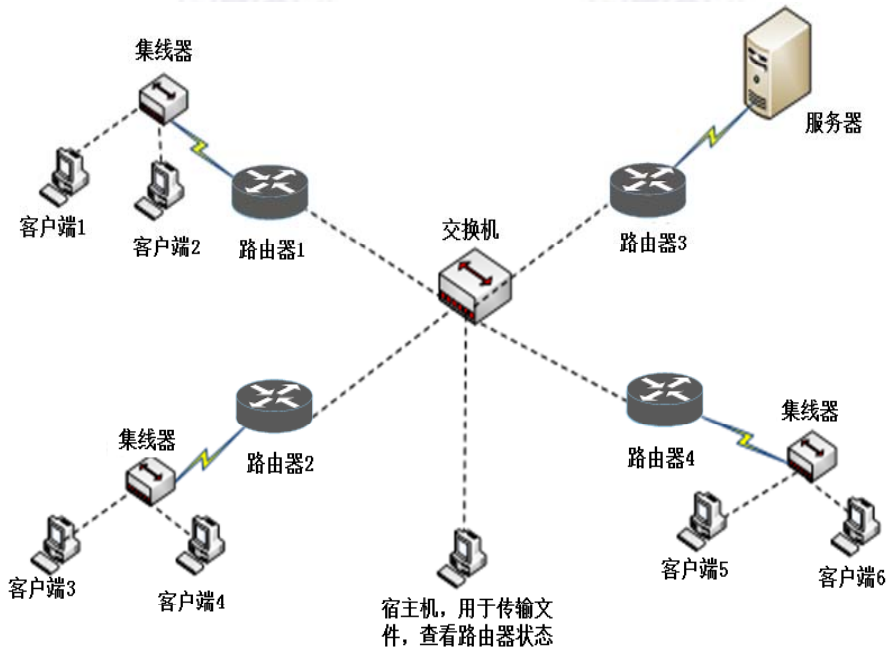


In overlay network: router forwards the requests to its neighbors

Router-->Server: Change source address, forward requests, cache

Performance Evaluation

Prototype system based on routers from Shanghai-bell

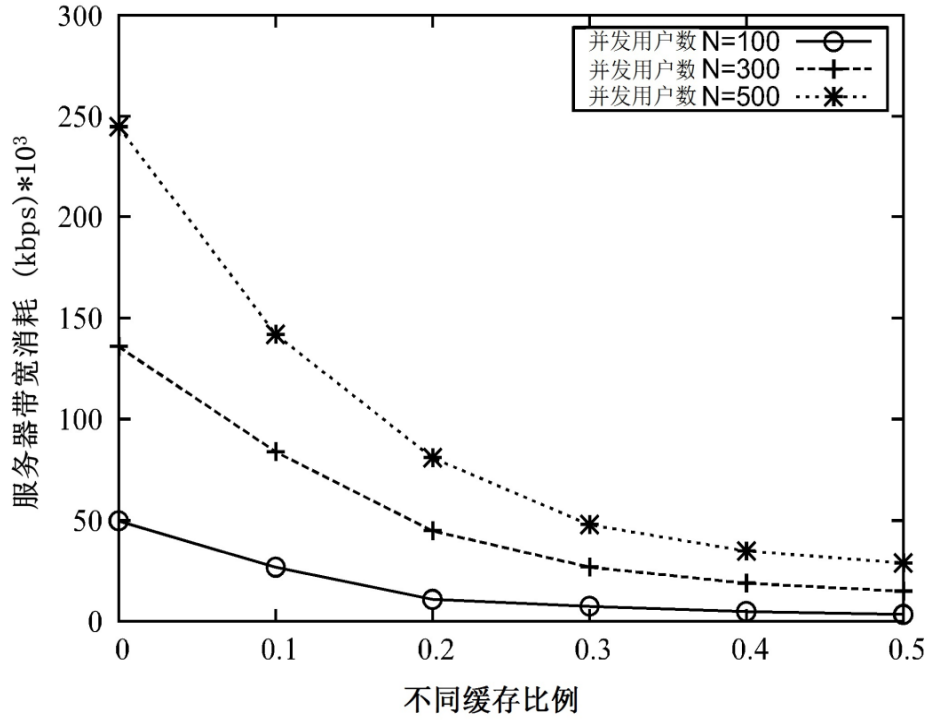


Parameters

Ave. Neighbor	3
Cache Size	*R*： 0~0.5
# videos	20
Rate	500kbps
Distribution	Zipf
# users	*N*： 100~500
User arrival	Poisson

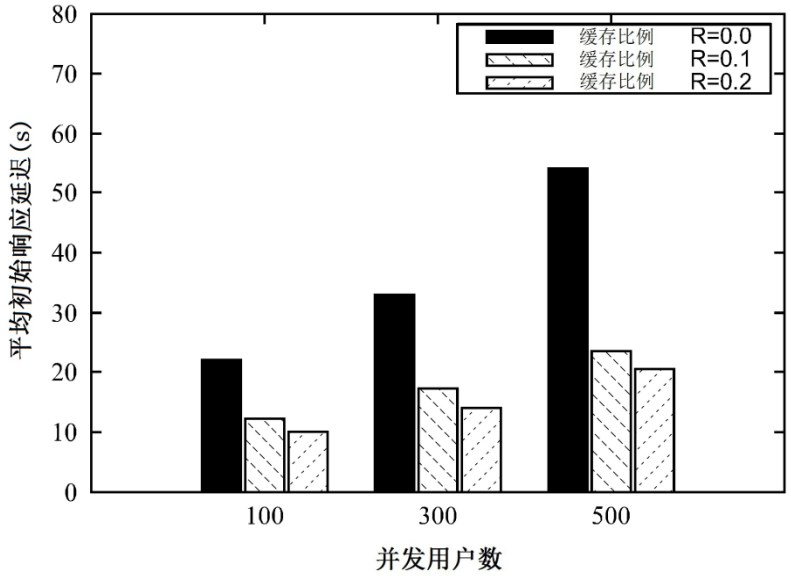
Performance Evaluation（1）

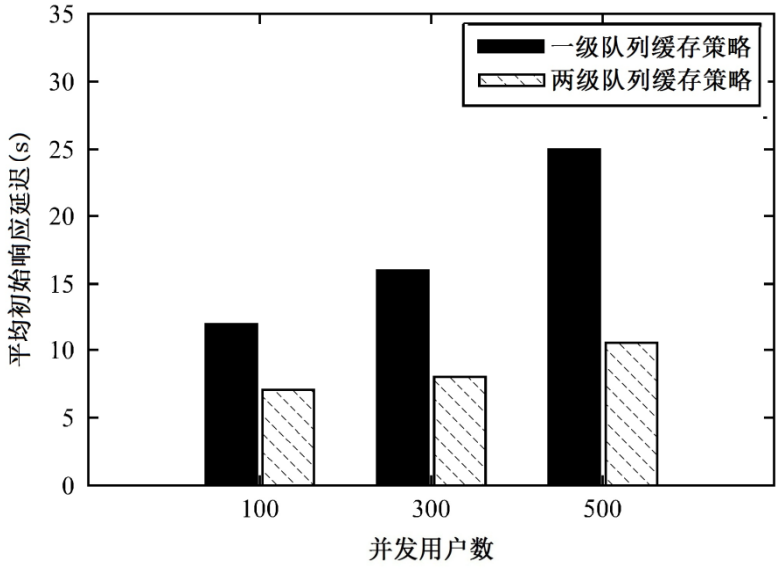
- Reduce bandwidth consumption
- Cache 30% data is enough



Performance Evaluation（2）

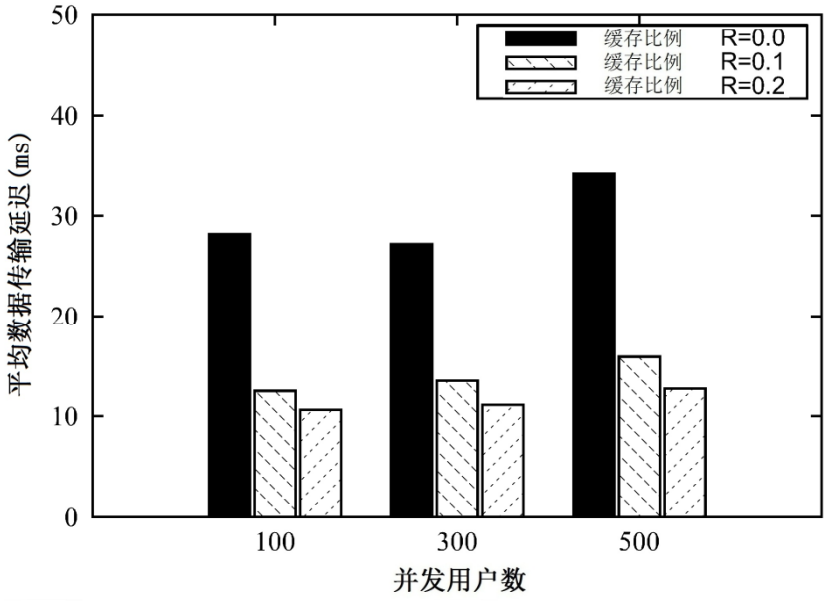
- Reduce initial delay, better user experience
- Two-level cache

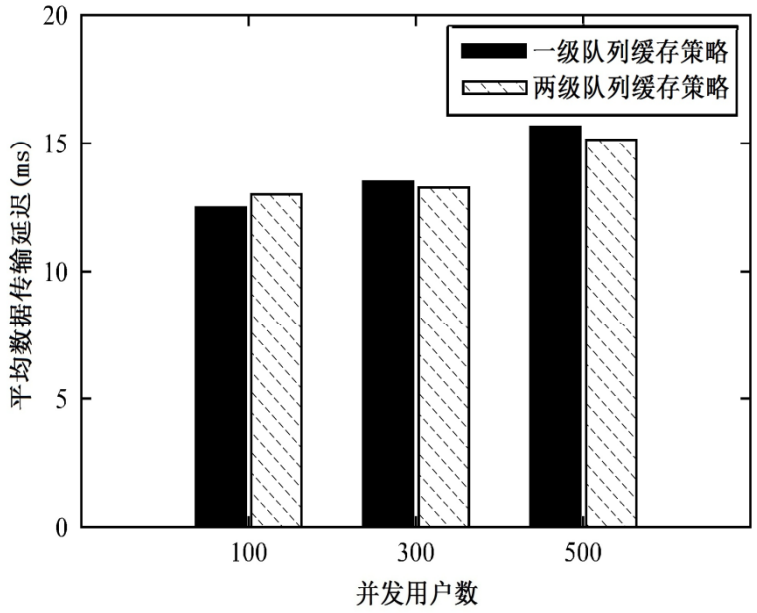




Performance Evaluation（3）

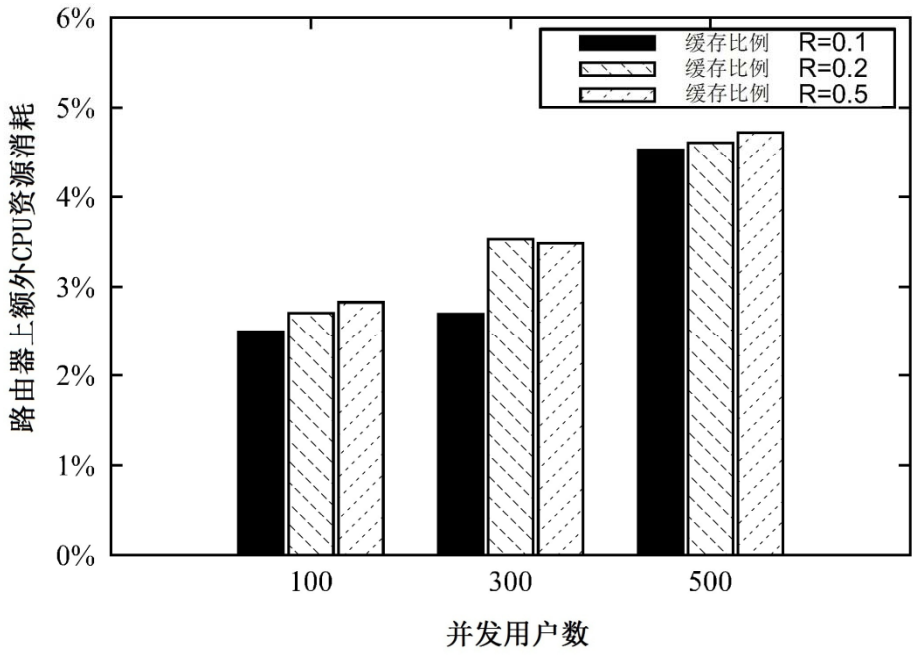
- Reduce transmission delay
- Better Scalability





Performance Evaluation（4）

- Acceptable computation overhead



Conclusion

- A framework of router caching for streaming medias
- Service recognition and caching at routers
- Cooperation among routers

Thanks ! ☺

Data Regeneration Processes with Network Coding in Distributed Storage Systems

Xin Wang
School of Computer Science, Fudan University
October 3, 2012

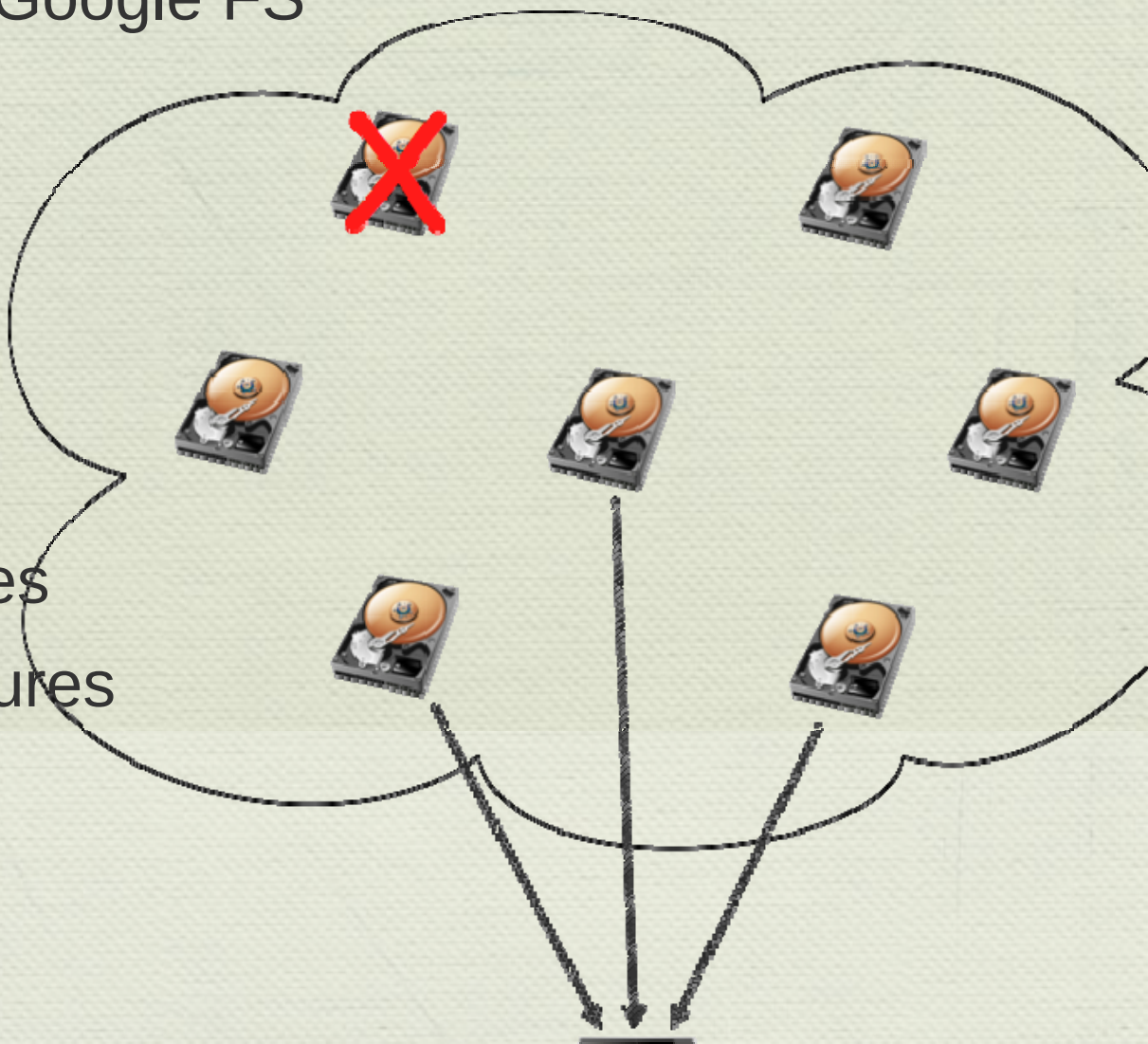
Distributed Storage Systems

Wide applications

- distributed processing systems: Google FS
- archival file systems
- P2P storage systems: Wuala

Features

- a substantial amount of data
- a large number of storage devices
- storage nodes are subject to failures



Data Integrity

Protect data to tolerate device failures by redundancy

- Replications
- Erasure Codes: Reed-Solomon codes, LDPC codes, and etc.
  - MDS property: any k among n encoded blocks can recover the original file
  - provide higher data integrity

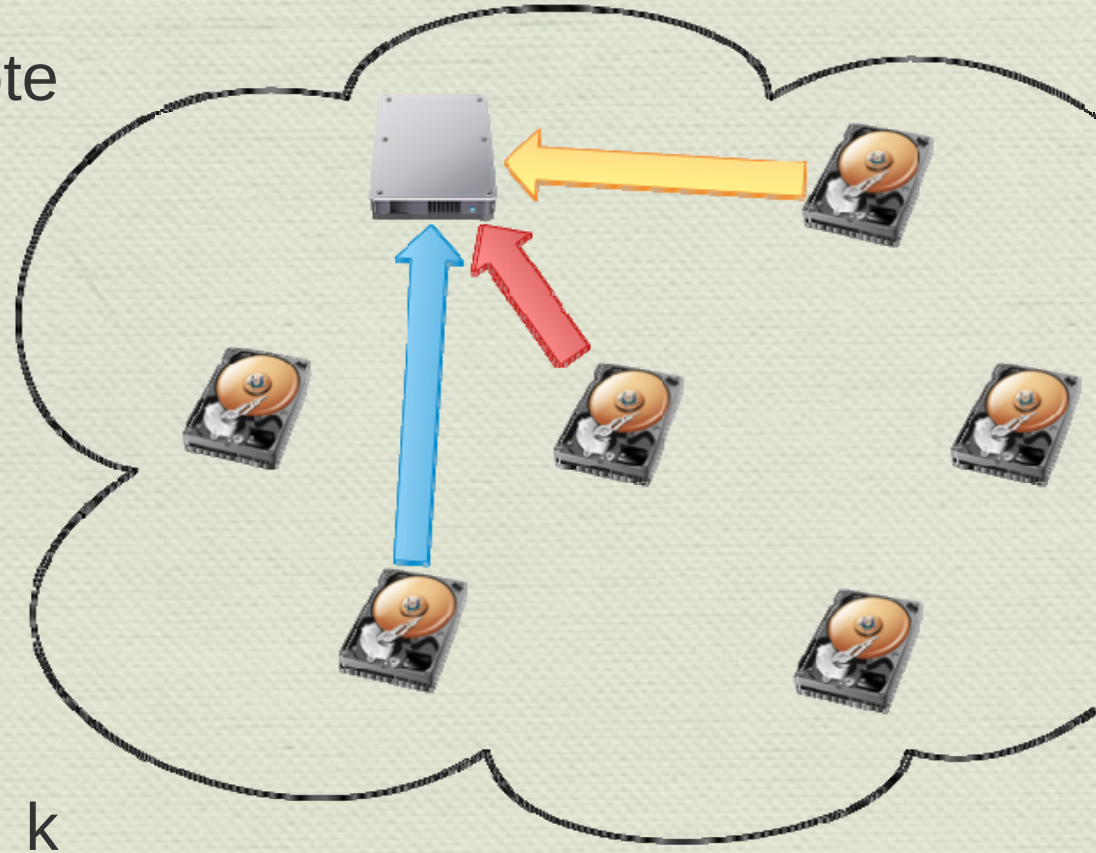
Maintenance of redundancy

- maintain a consistent amount of redundancy
- regenerate lost redundancy after failures
- different redundancy schemes lead to different regeneration process

Regeneration Process

Replicated redundancy

- a newcomer gets a copy from a remote storage device
  - simple operations
  - low computational cost
  - low delay
  - high storage cost



Coded redundancy

- a newcomer obtains data from other k storage devices (providers)
  - maintain a high data integrity
  - high computational cost on a single node

Regeneration Time

Previous idea: design codes to minimize the traffic

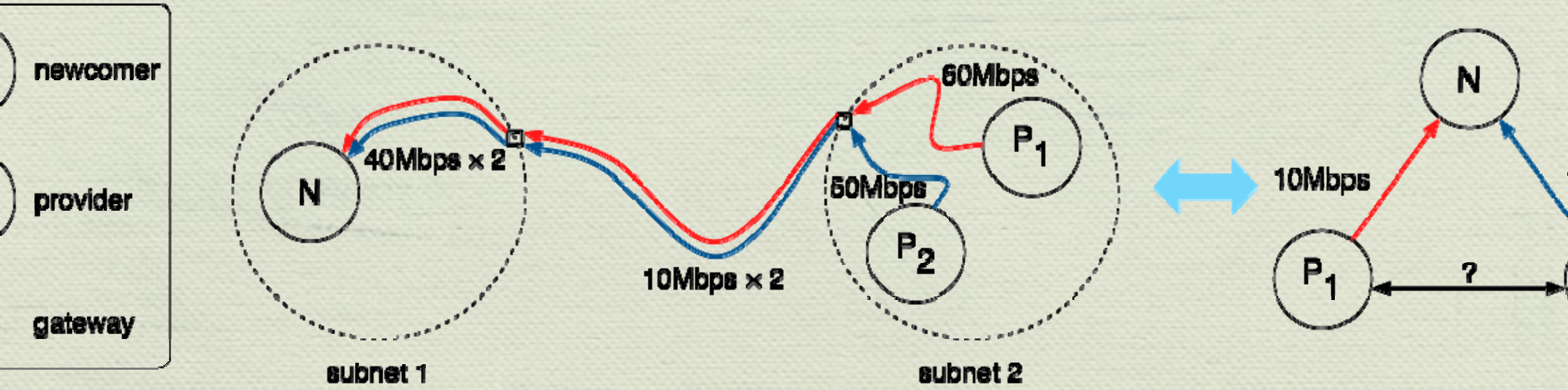
- regenerating codes
  - contact more than k providers
  - stores more data than conventional erasure codes
- bandwidth consumption approximately as well as replication

Our idea: design regeneration processes to save time

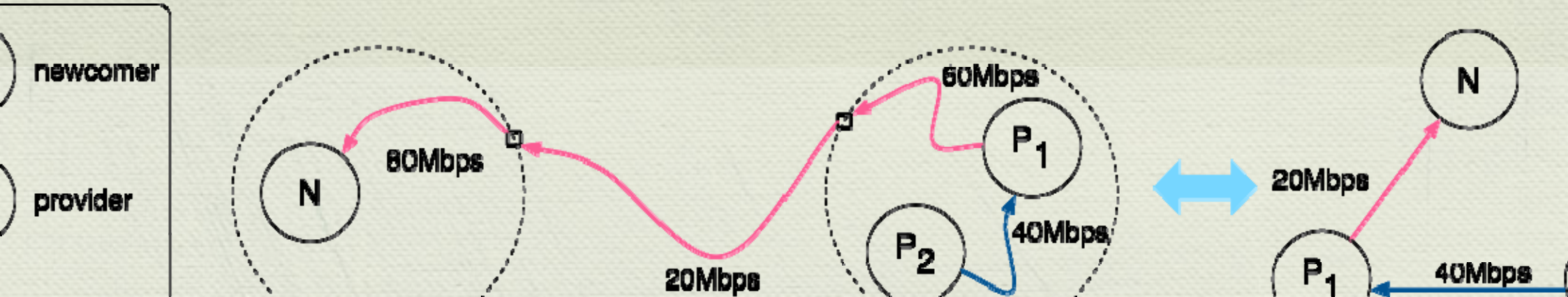
- exploit the bandwidth heterogeneity
- pipeline the regeneration process

Exploit the bandwidth heterogeneity

- bandwidth of a link inside a subnet is usually more available than that betwe two subnets
- the farther away two nodes are, the less available the bandwidth between th are likely to be



Bypass "bottleneck" links in a Peer-to-Peer fashion (INFOCOM 2010)

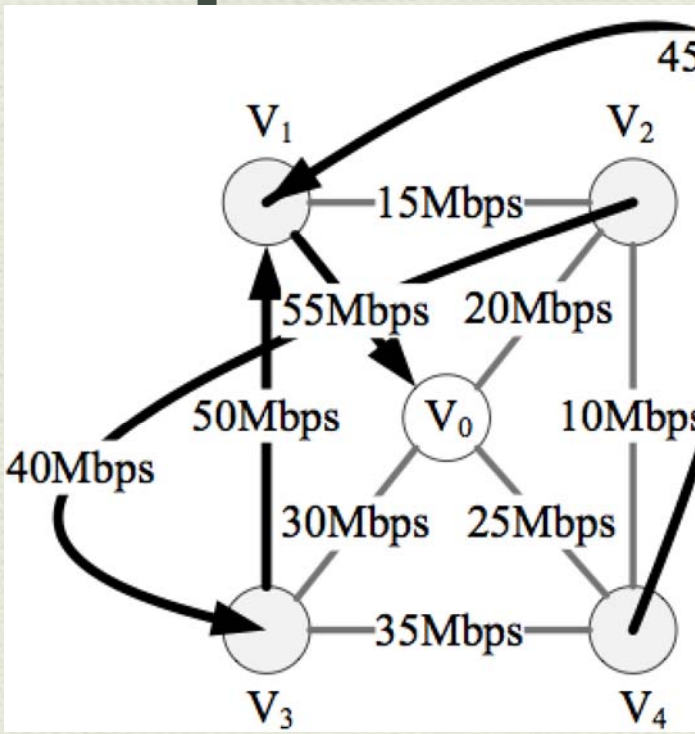


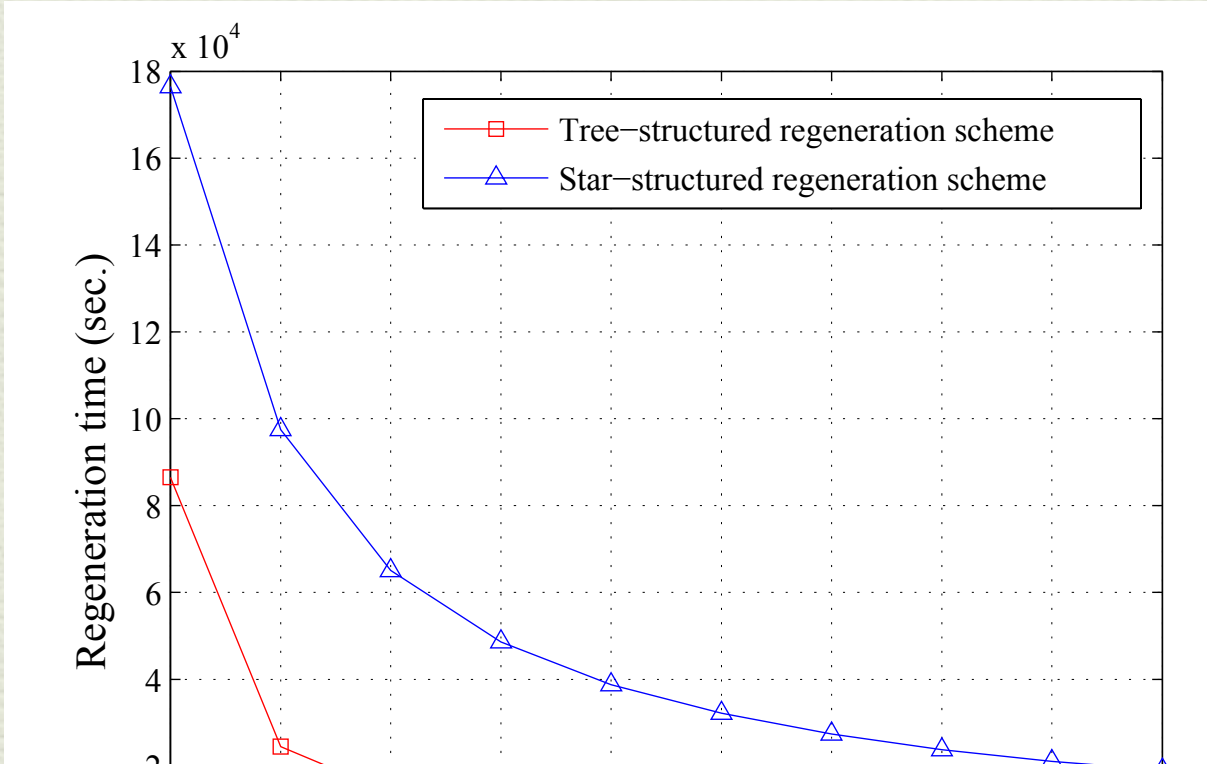
Tree-structured Repair

ewcomer as the root, all
des transmit data upward

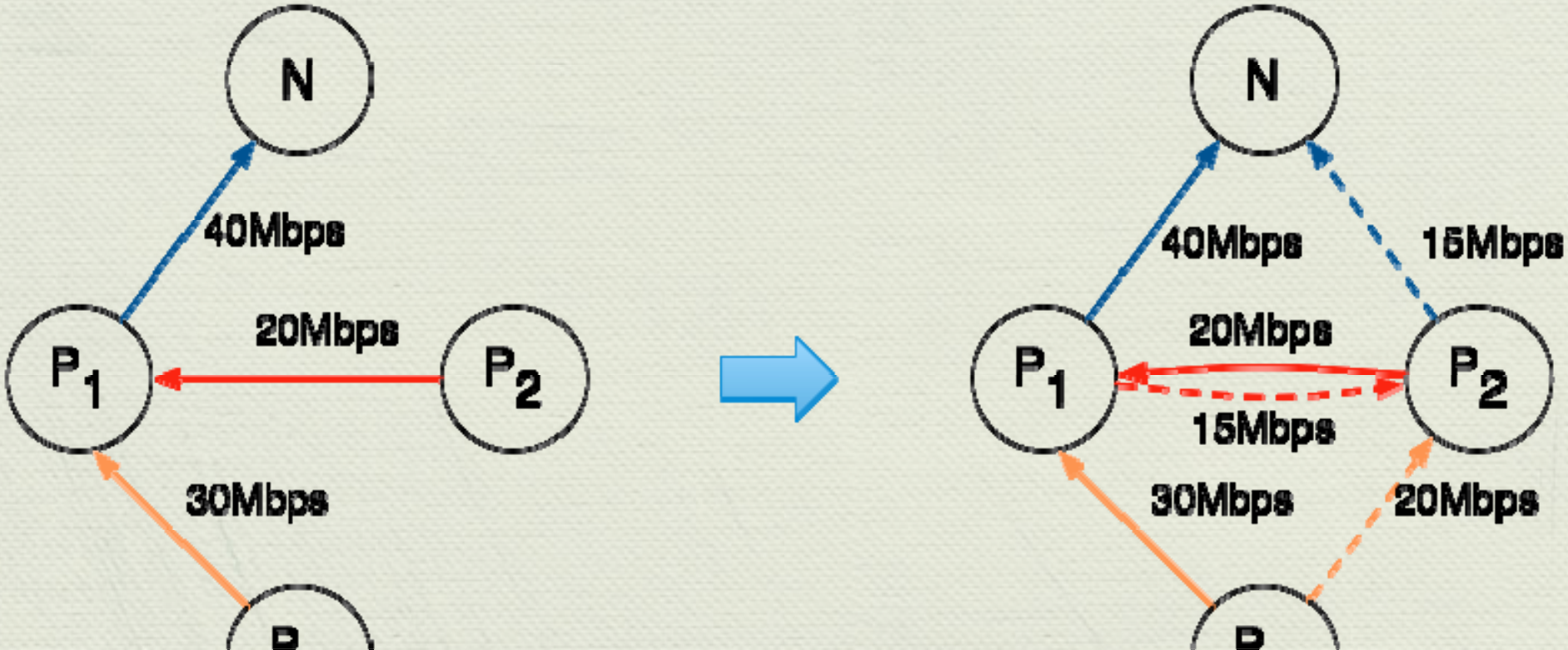
on-leaf nodes performs
etwork coding

aximum Bandwidth
anning Tree





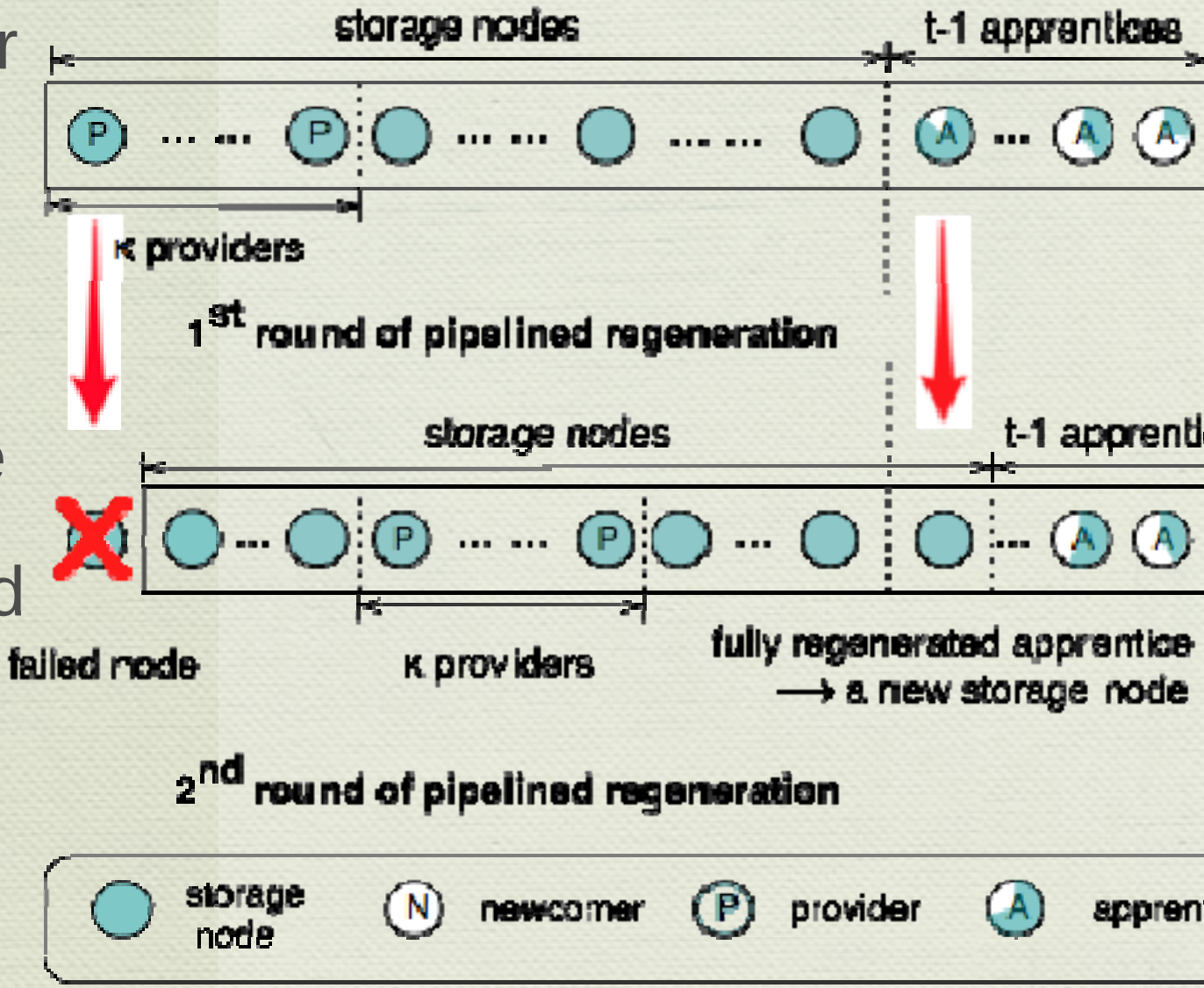
- one tree is not enough to fully exploit the path diversity
  - network bottleneck can be inside the network in the Internet
  - multiple paths in modern data center topologies
- build parallel regeneration trees
  - exploit the bandwidth diversity implicitly
  - greedy and optimal algorithms & performance analysis

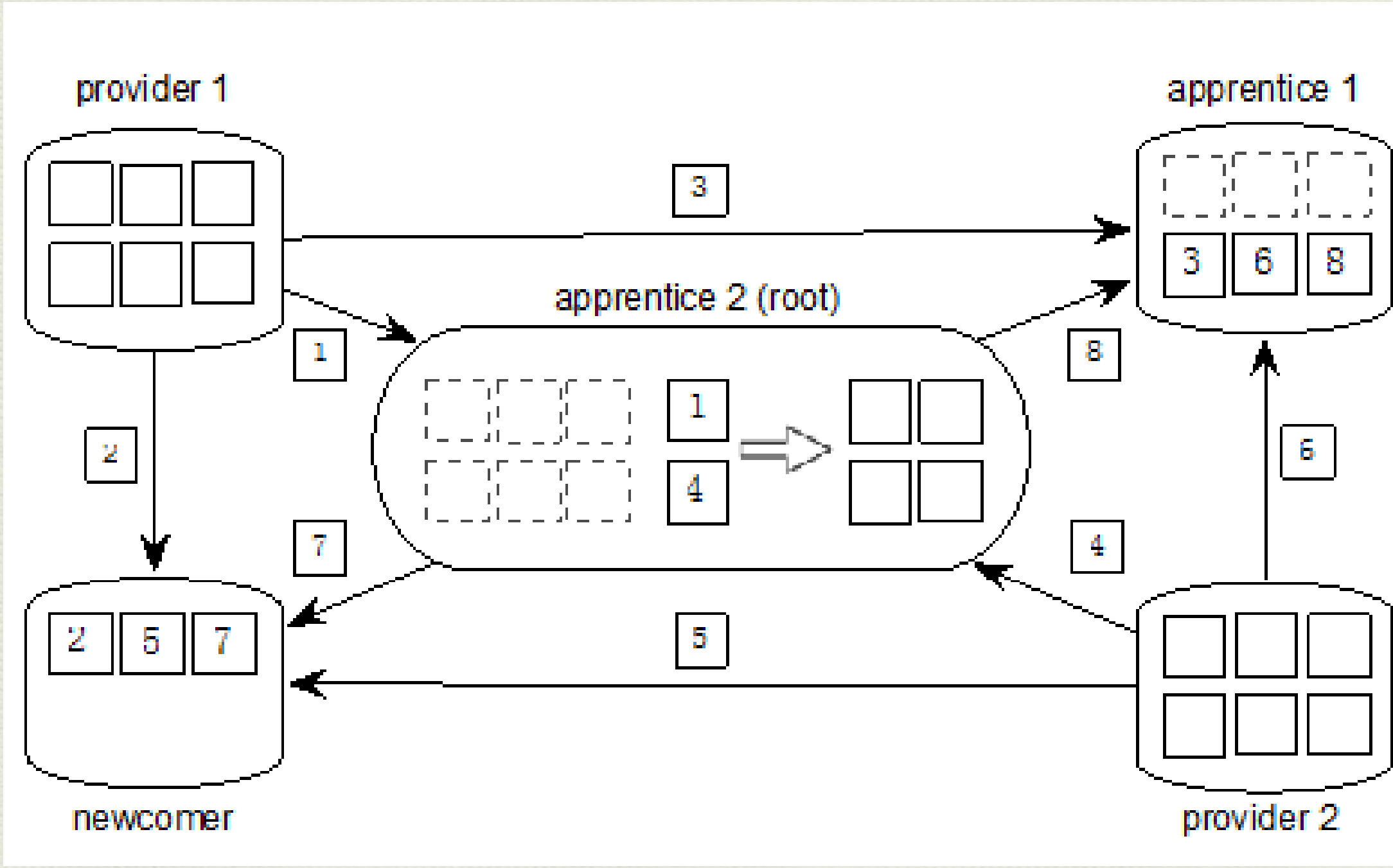


Pipeline the regeneration process

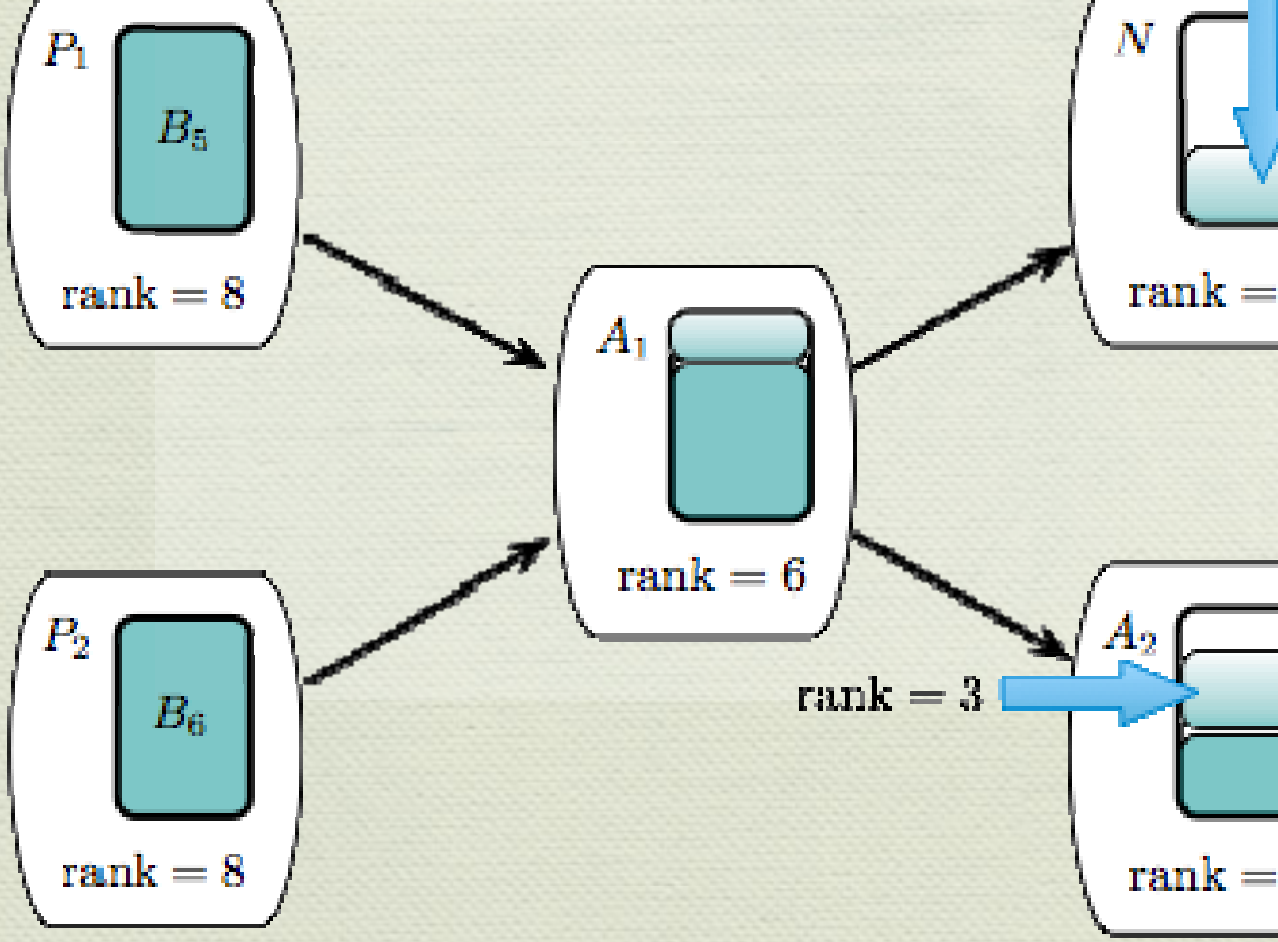
generation

- a newcomer becomes partially regenerated after regeneration
- partially regenerated nodes are referred to as apprentices
- apprentices receive more and more data in each round of regeneration and finally "graduate" to become a storage node
- one apprentice will be fully regenerated after each round of regeneration




provider 1
apprentice 1
3
3
6
8
apprentice 2 (root)
1
8
1
4
2
6
7
4
2
5
7
5
newcomer
provider 2

- root: the apprentice with the highest rank
- all providers send data to the root
- the root becomes fully regenerated
- the root sends coded data to other apprentices and the newcomer



Performances

- participating nodes: the order of the square root of the number used in the conventional regeneration process
- reduce bandwidth consumption to maintain the same level of data availability

Extensions

- support both random linear codes and regenerating codes

Summary

Exploit the bandwidth heterogeneity

- 3 full papers: (IWQoS 2009), INFOCOM 2010, CollaborateCom 2010
- 1 poster: USENIX FAST 2010
- 1 China patent

Pipeline the regeneration process

- 1 paper: NetCod 2011
- 2 papers under review: TPDS, INFOCOM 2013

Future plan

- Combine them together
- functional repair -> exact repair
- more support for regenerating codes (both MSR and MBR codes)

Thank you!

For more information:
http://sonic.fudan.edu.cn/~xinw/

Data Regeneration Processes with Network Coding in Distributed Storage Systems

Xin Wang
School of Computer Science, Fudan University
October 3, 2012

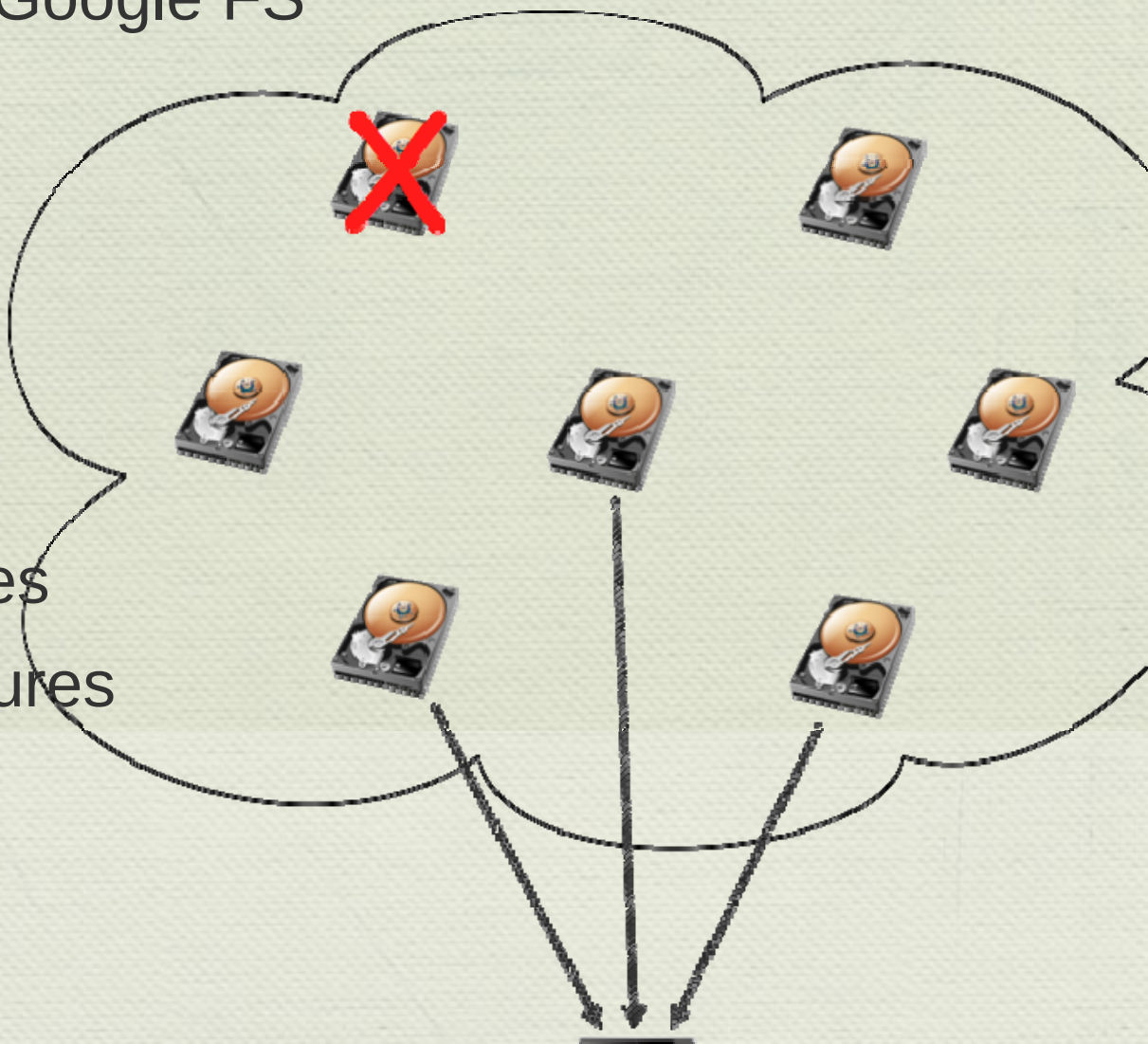
Distributed Storage Systems

Wide applications

- distributed processing systems: Google FS
- archival file systems
- P2P storage systems: Wuala

Features

- a substantial amount of data
- a large number of storage devices
- storage nodes are subject to failures



Data Integrity

Protect data to tolerate device failures by redundancy

- Replications
- Erasure Codes: Reed-Solomon codes, LDPC codes, and etc.
  - MDS property: any k among n encoded blocks can recover the original file
  - provide higher data integrity

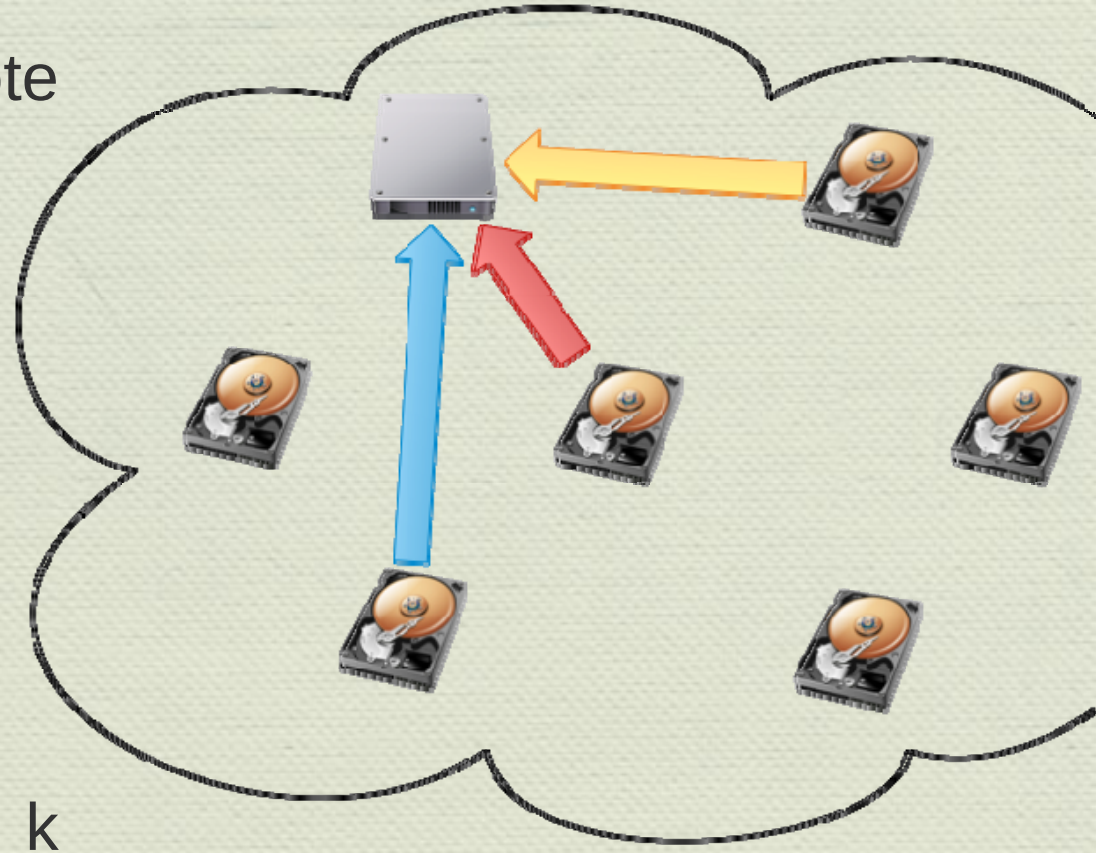
Maintenance of redundancy

- maintain a consistent amount of redundancy
- regenerate lost redundancy after failures
- different redundancy schemes lead to different regeneration process

Regeneration Process

Replicated redundancy

- a newcomer gets a copy from a remote storage device
  - simple operations
  - low computational cost
  - low delay
  - high storage cost



Coded redundancy

- a newcomer obtains data from other k storage devices (providers)
  - maintain a high data integrity
  - high computational cost on a single node

Regeneration Time

Previous idea: design codes to minimize the traffic

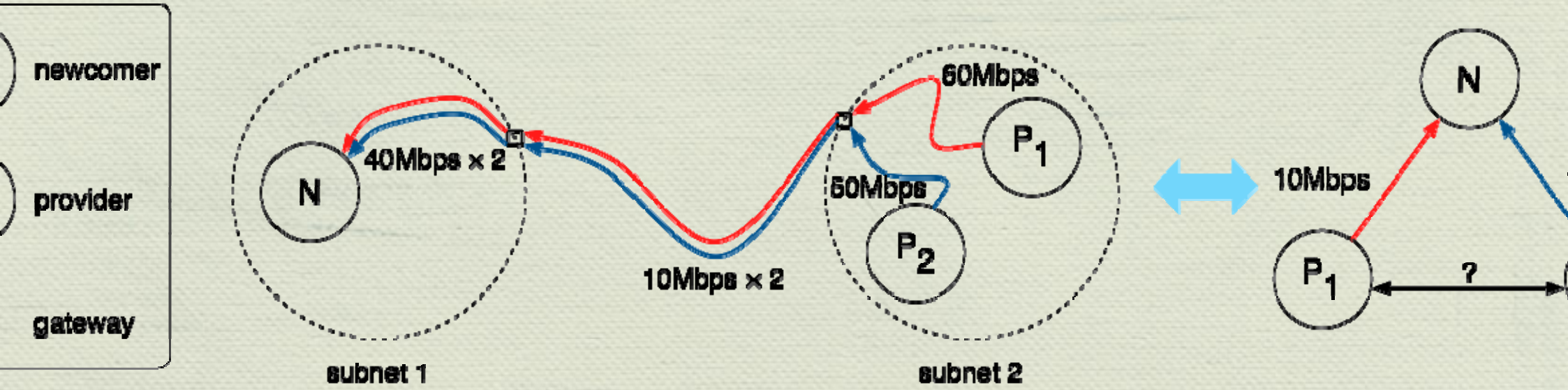
- regenerating codes
  - contact more than k providers
  - stores more data than conventional erasure codes
- bandwidth consumption approximately as well as replication

Our idea: design regeneration processes to save time

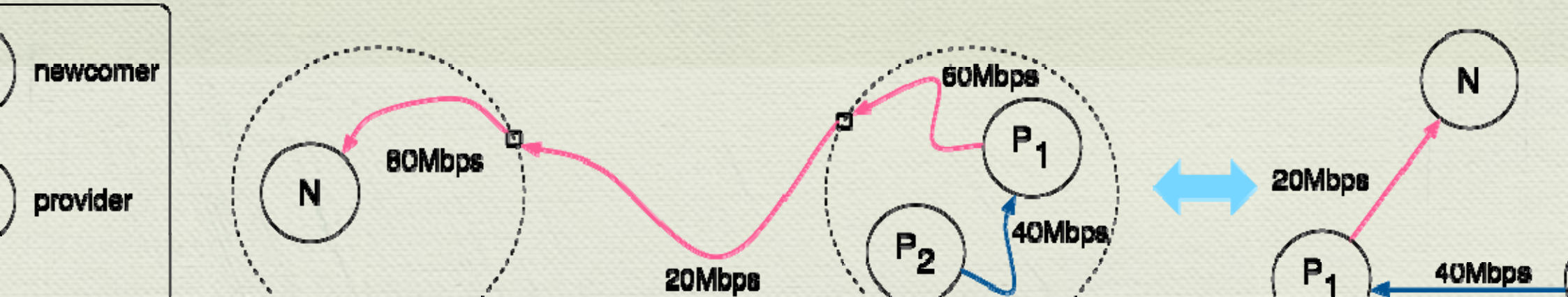
- exploit the bandwidth heterogeneity
- pipeline the regeneration process

Exploit the bandwidth heterogeneity

- bandwidth of a link inside a subnet is usually more available than that betwe two subnets
- the farther away two nodes are, the less available the bandwidth between th are likely to be



Bypass "bottleneck" links in a Peer-to-Peer fashion (INFOCOM 2010)

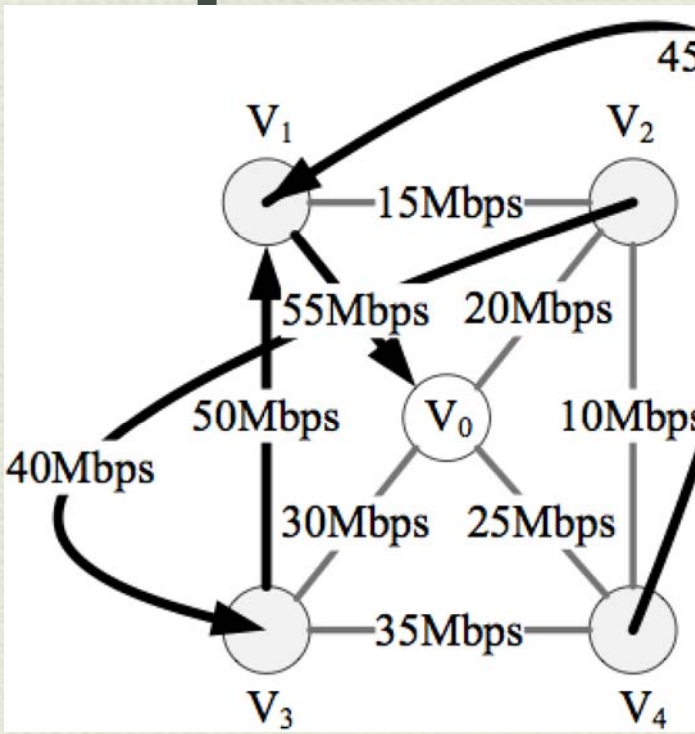


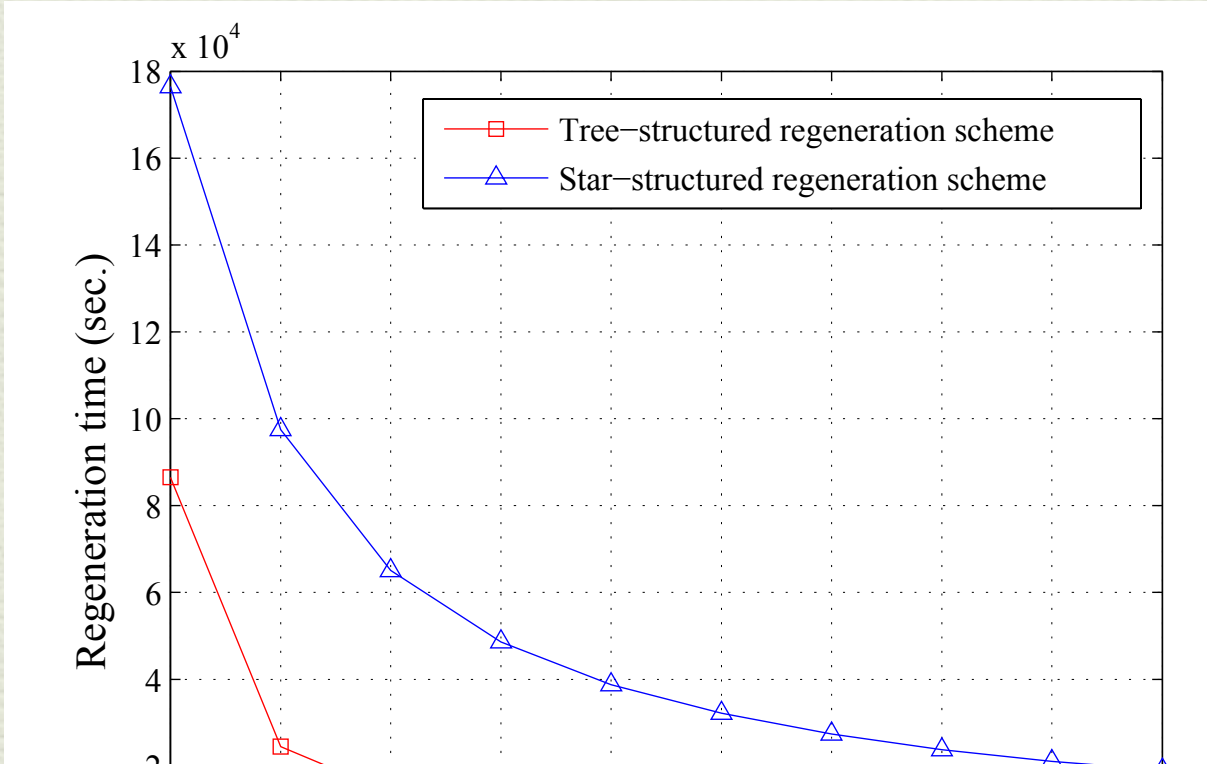
Tree-structured Repair

ewcomer as the root, all
des transmit data upward

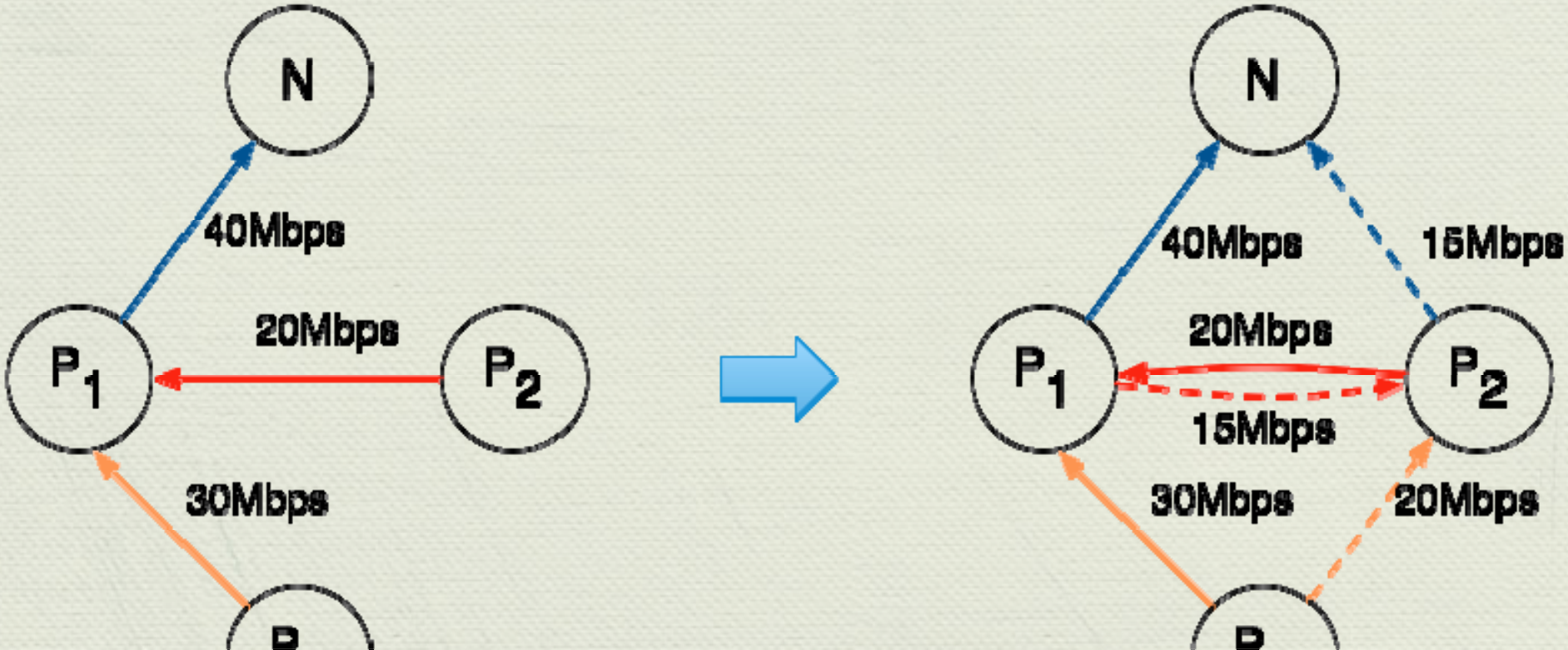
on-leaf nodes performs
etwork coding

aximum Bandwidth
anning Tree





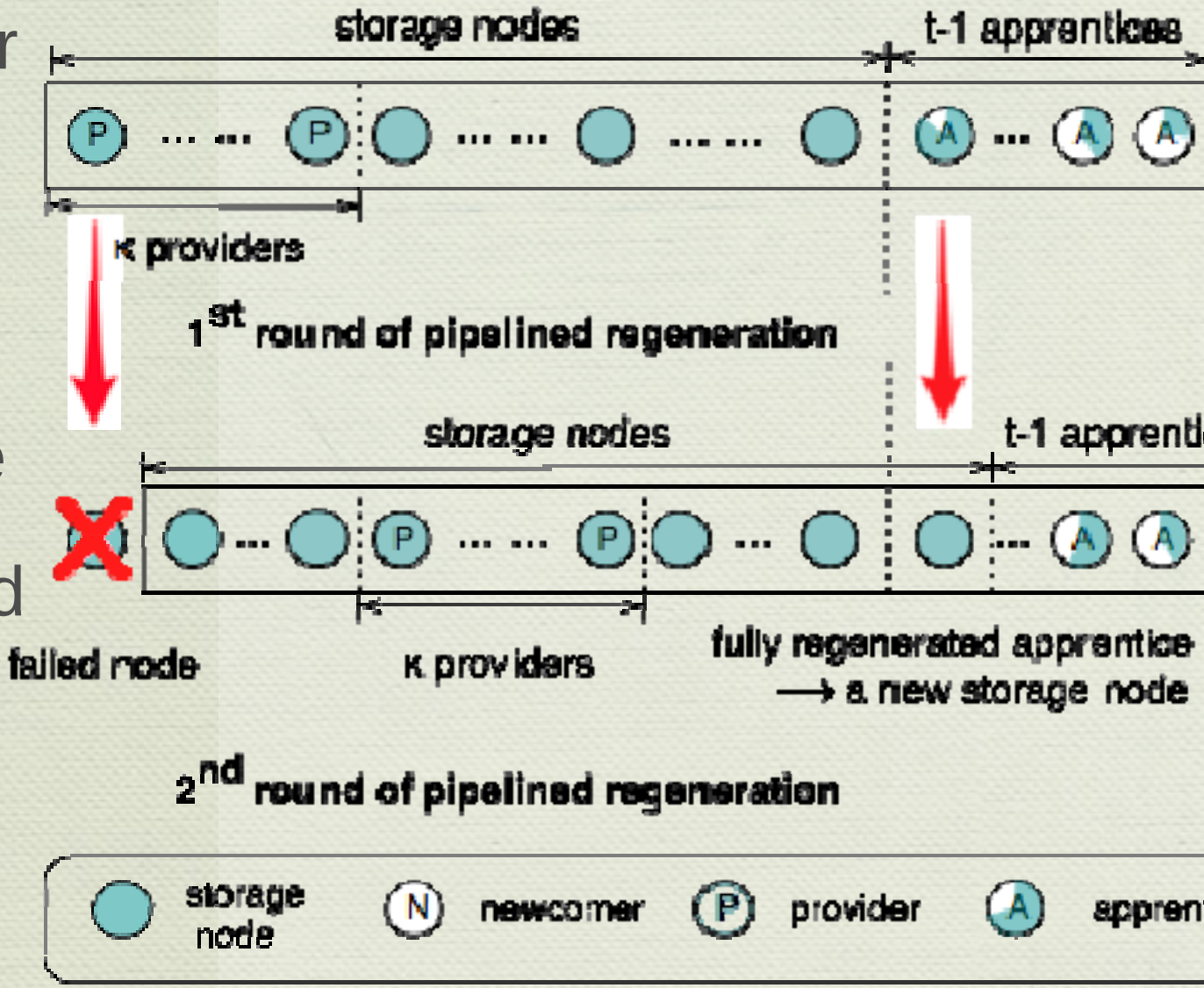
- one tree is not enough to fully exploit the path diversity
  - network bottleneck can be inside the network in the Internet
  - multiple paths in modern data center topologies
- build parallel regeneration trees
  - exploit the bandwidth diversity implicitly
  - greedy and optimal algorithms & performance analysis

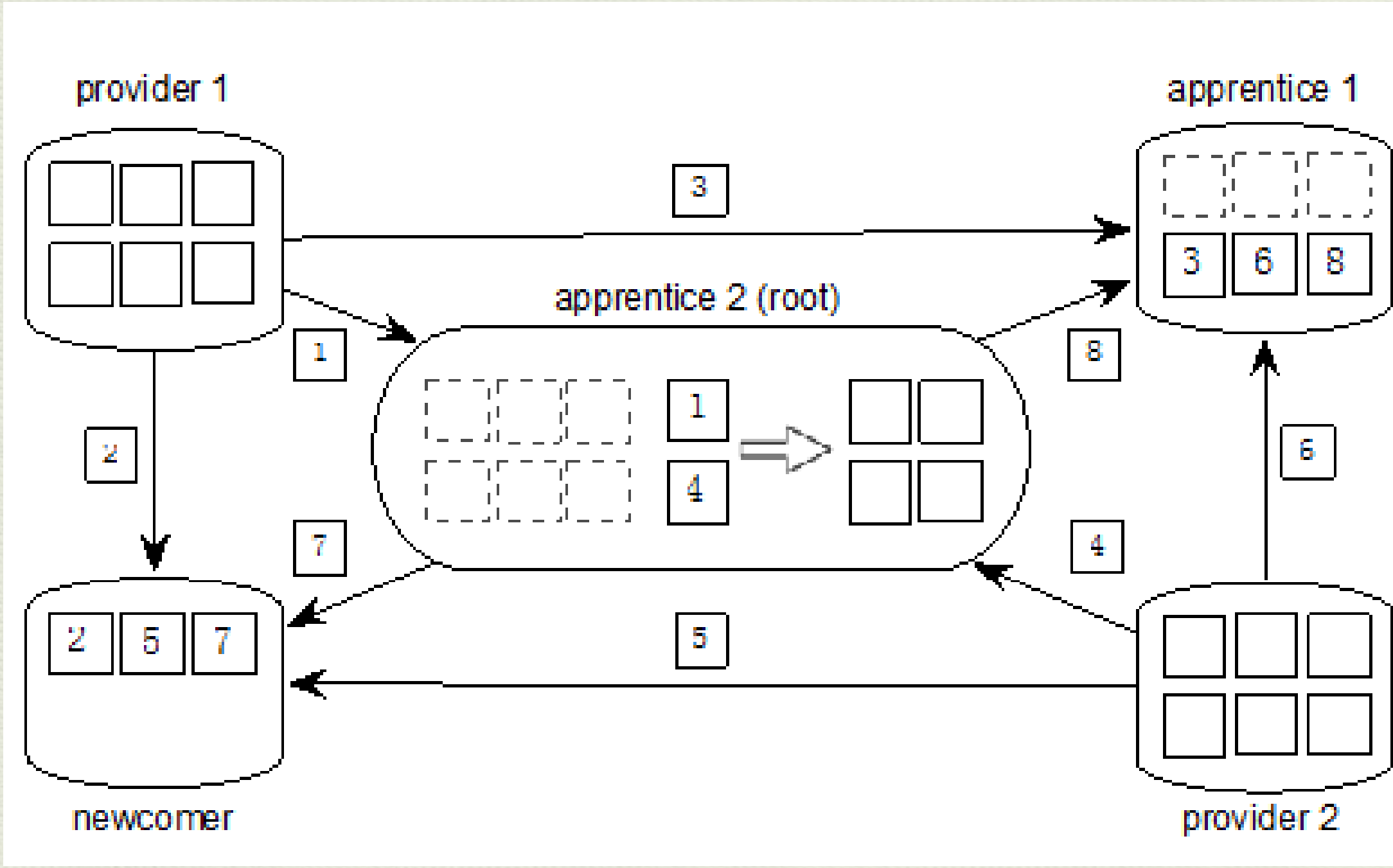


Pipeline the regeneration process

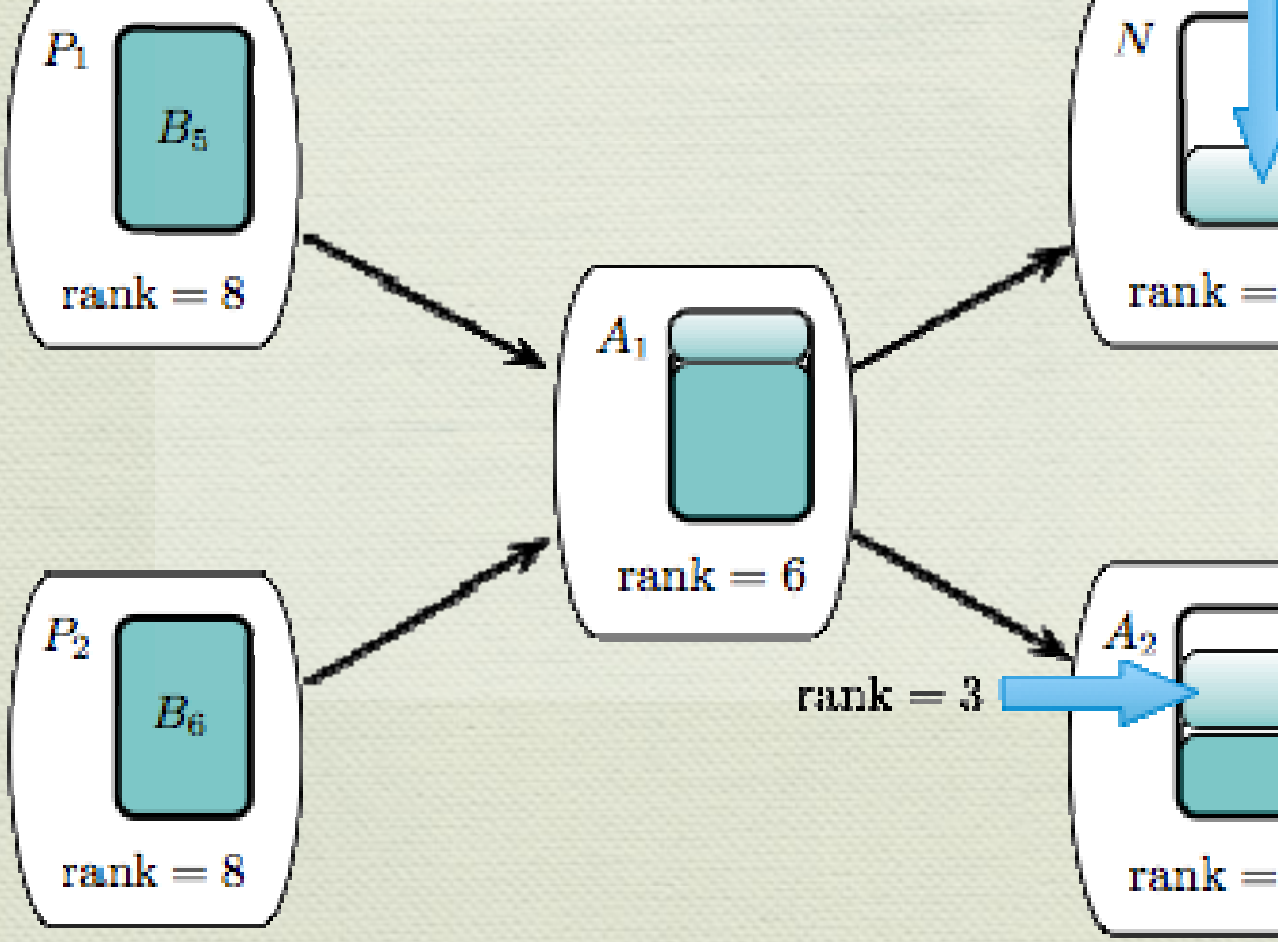
generation

- a newcomer becomes partially regenerated after regeneration
- partially regenerated nodes are referred to as apprentices
- apprentices receive more and more data in each round of regeneration and finally "graduate" to become a storage node
- one apprentice will be fully regenerated after each round of regeneration




provider 1
apprentice 1
3
3
6
8
apprentice 2 (root)
1
8
1
4
2
6
7
4
2
5
7
5
newcomer
provider 2

- root: the apprentice with the highest rank
- all providers send data to the root
- the root becomes fully regenerated
- the root sends coded data to other apprentices and the newcomer



Performances

- participating nodes: the order of the square root of the number used in the conventional regeneration process
- reduce bandwidth consumption to maintain the same level of data availability

Extensions

- support both random linear codes and regenerating codes

Summary

Exploit the bandwidth heterogeneity

- 3 full papers: (IWQoS 2009), INFOCOM 2010, CollaborateCom 2010
- 1 poster: USENIX FAST 2010
- 1 China patent

Pipeline the regeneration process

- 1 paper: NetCod 2011
- 2 papers under review: TPDS, INFOCOM 2013

Future plan

- Combine them together
- functional repair -> exact repair
- more support for regenerating codes (both MSR and MBR codes)

Thank you!

For more information:
http://sonic.fudan.edu.cn/~xinw/