Support Mobile and Distributed Applications with Named Data Networking

Zhenkai Zhu

Computer Science Department
University of California, Los Angeles
CA, 90095

zhenkai@cs.ucla.edu

May 22, 2013

Research Overview

Supporting mobility in the global Internet

- thoroughly studied existing IP mobility solutions
- developed solution for the scalability problem in Global HA to HA protocol
- provided a new perspective of mobility support

Exploring new design patterns for distributed applications over NDN

- explored naming conventions in application design
- proposed a general-purpose dataset synchronization protocol that has the potential of supporting a wide range of distributed applications
- developed security solutions for distributed applications using NDN's data-centric approach

Motivation

- Today's Internet is increasingly mobile
- IP's host-to-host communication model is increasing challenged by emerging communication patterns
- Despite years of IP mobility support research, few solutions enjoyed wide adoption
  - it is time to rethink what is mobility support and how to align mobility support with the applications' need
- NDN aims to accommodate emerging communication patterns by taking fundamental changes to today's IP Internet architecture
  - we must learn how to design applications to fully exploit the benefits of such an architectural shift by trying out new applications
- This work is to push forward this new front of networking research

Supporting mobility in the global Internet

The IP mobility problems

- How to find a mobile after it moves?
- How to minimize the interruption to the communications?
- How to make sure one is talking to the right host?

Solution space

- One can reach a mobile by the same IP address regardless of whether it moves or not
  - e.g. broadcast the mobile's location via routing
- One can find the mobile's location based on a stable piece of information
  - e.g. home address, DNS name
  - need to maintain an up-to-date mapping

Three basic components of IP mobility support

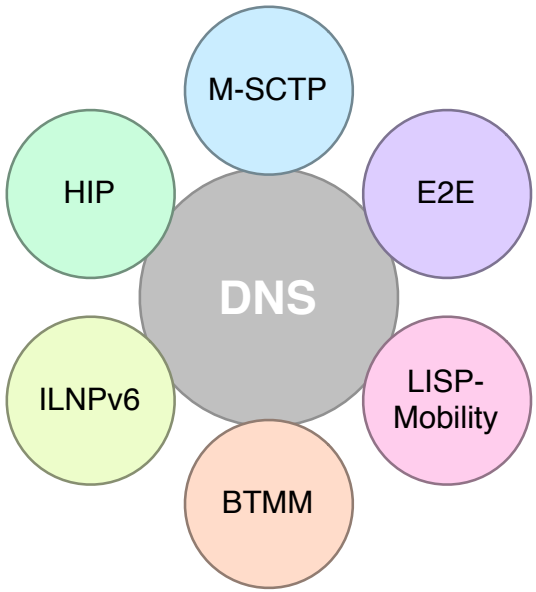
- Stable identifier
  - sender's knowledge about the mobile that does not change due to mobility
- Locator
  - an IP address for the mobile's current location
- Mapping between the two
  - kept in dedicated network entities
  - or by the routing system

IP mobility solutions

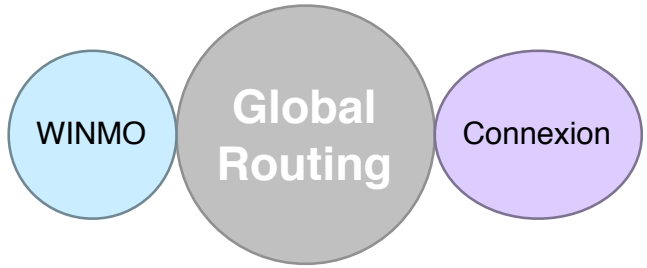
I conducted a complete and systematic survey of IP mobility solutions that have been proposed in order to gain a good understanding of the solution space and shed light on future efforts.

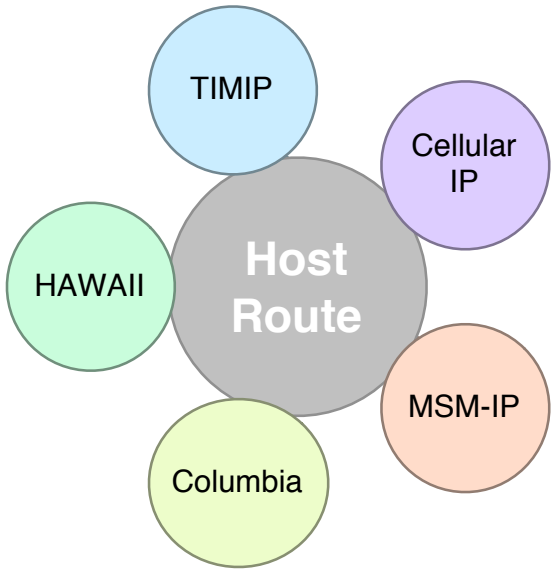
Protocol	Year	Protocol	Year
Columbia	1991	TIMIP	2001
Virtual IP	1991	M-SCTP	2002
LSR	1993	HIP	2003
Mobile IP	1996	Connexion	2004
MSM-IP	1997	ILNPv6	2005
Cellular IP	1998	Global HAHA	2006
HMIP	1998	PMIP	2006
FMIP	1998	BTMM	2007
HAWAII	1999	WINMO	2008
NEMO	2000	LISP-Mobility	2009
E2E	2000		

Mapping



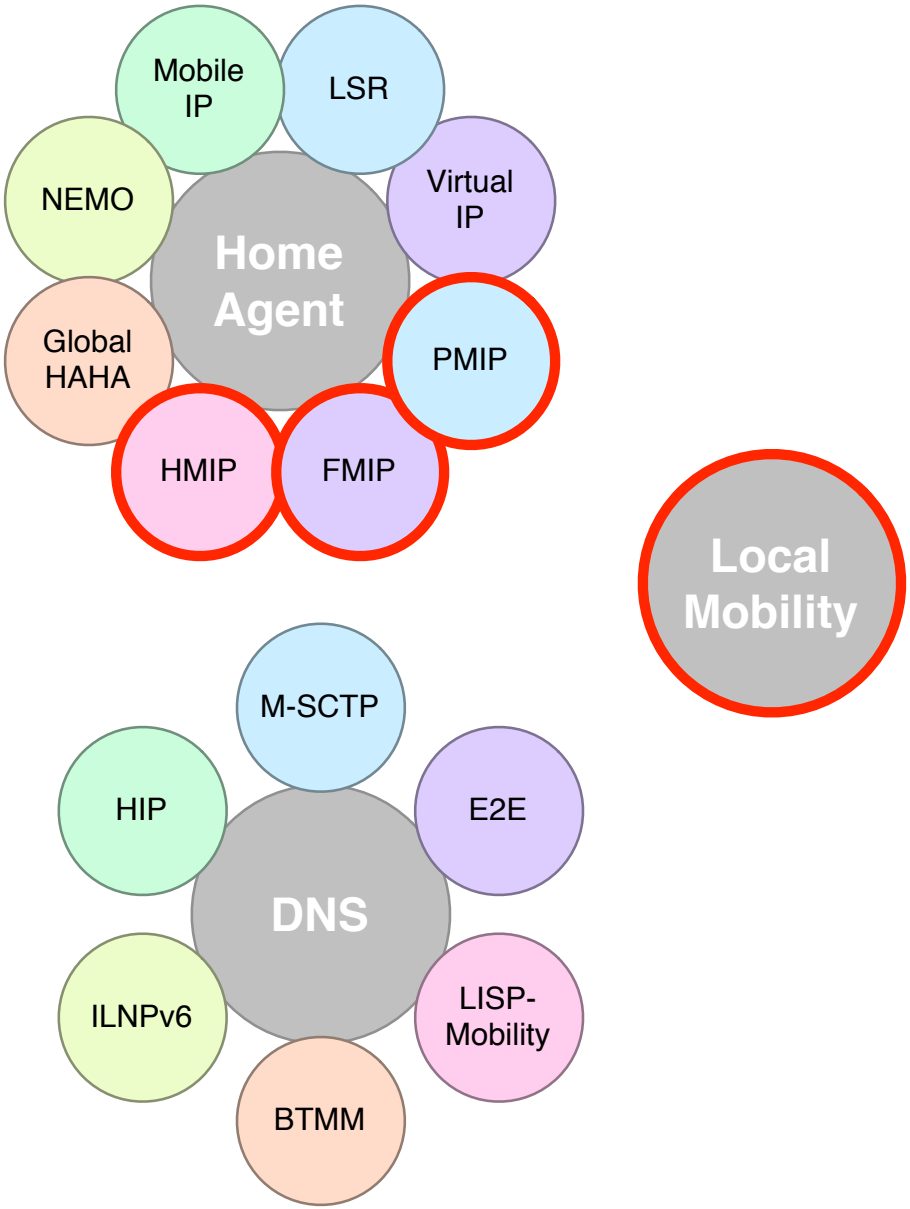
Routing





Mapping

Routing



Summary of IP mobility solution space

- Mobility support essentially involves three basic components
  - a stable identifier, a locator and the mapping between the two
- Broadcast (routing) based approach is simplest, most robust, and work well in small networks, but raises scaling concerns when applied to large networks
- Mapping based approach (home agent, DNS, etc.) can scale well with large network, but introduces dependency on a 3rd party.
- Managing local movements local helps improve the performance and scalability

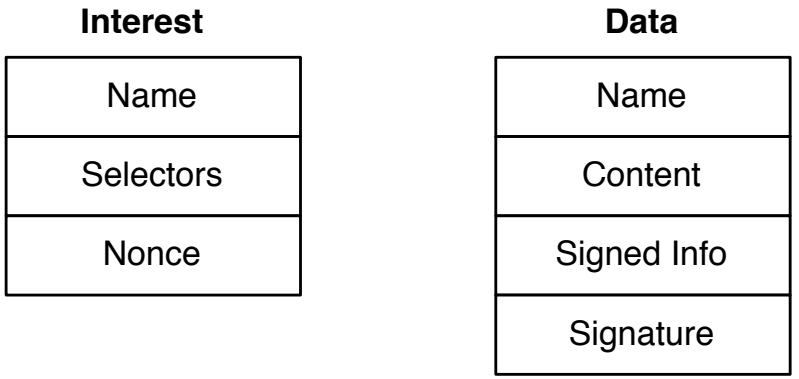
Limitations of IP mobility support

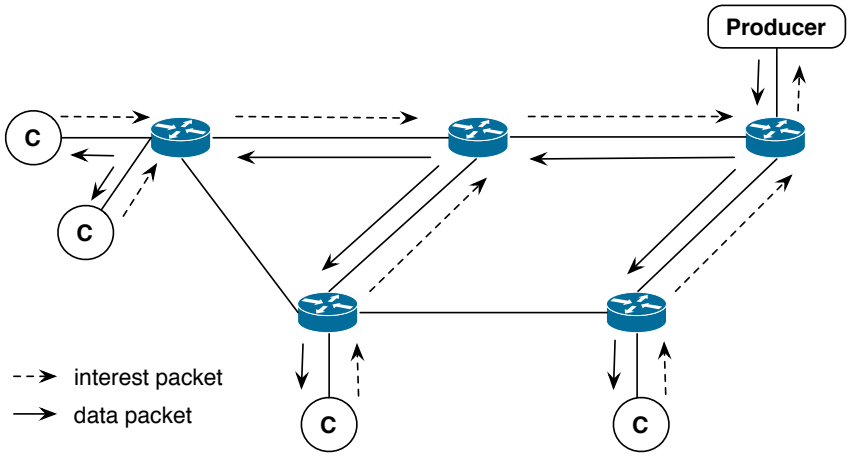
- Security is tied to IP addresses
  - often weak and need frequent updates during the move
  - quite a few proposals didn't even mention about security
- Incomplete solution: only considers cases when mobiles are connected to the infrastructure
  - yet mobility can easily lead to intermittent connectivity or ad hoc connectivity among a set of mobiles
  - ad hoc mobility and DTN becomes separate branches in networking research with their own solutions
- Because IP is for point-to-point delivery, so far mobility solutions also focus only on maintaining point-to-point sessions

A new perspective on mobility support

What NDN brings

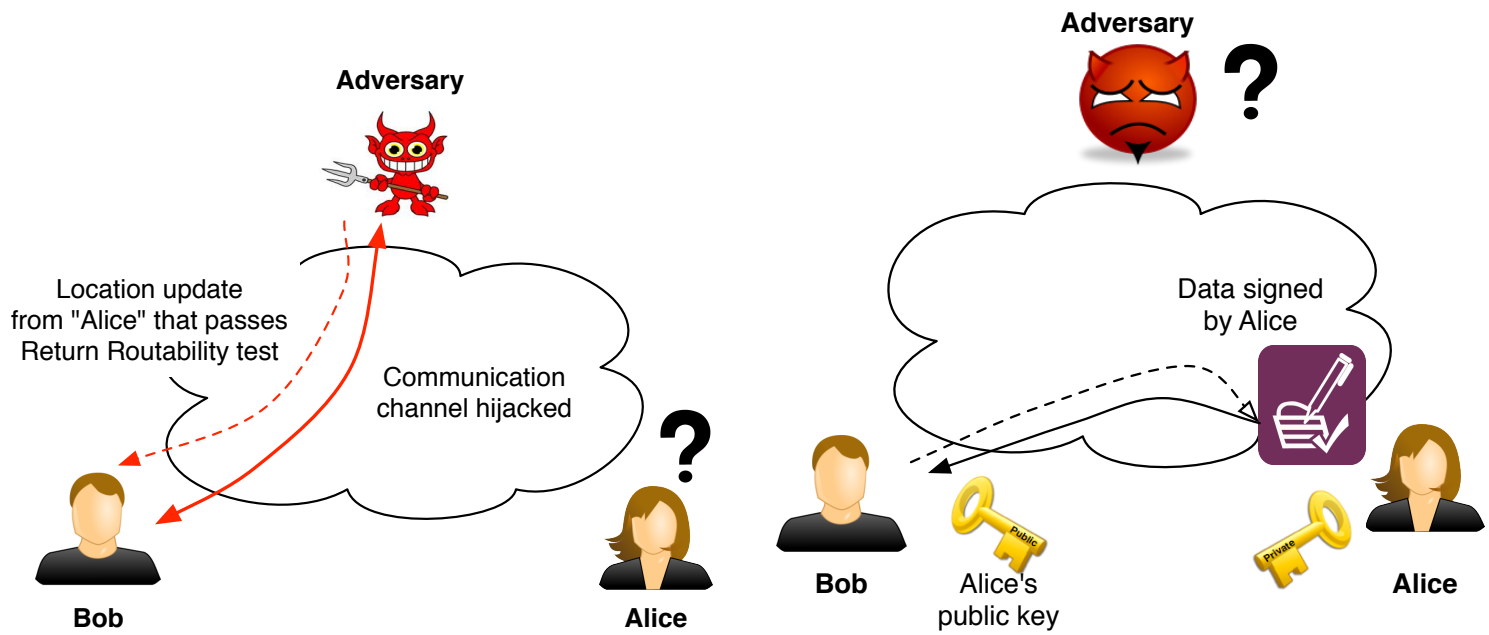
- Data-centric security
  - every piece of data is named and signed
  - where the data is obtained doesn't matter
- Receiver-driven communications
  - must send Interest for data
  - data follows the reverse path back to requester
- Intelligent data plane
  - per-packet state at routers





Security done right

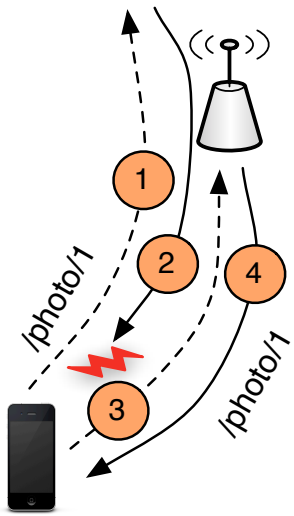
- Security has always been a big challenge for IP mobility solutions
- NDN secures data
  - decouples trust in data from trust in host



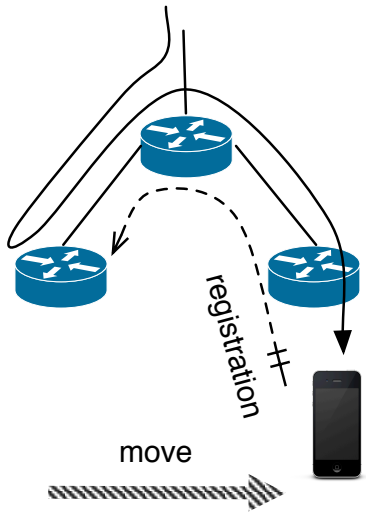
(a) IP (b) NDN

Enhanced performance with network caching

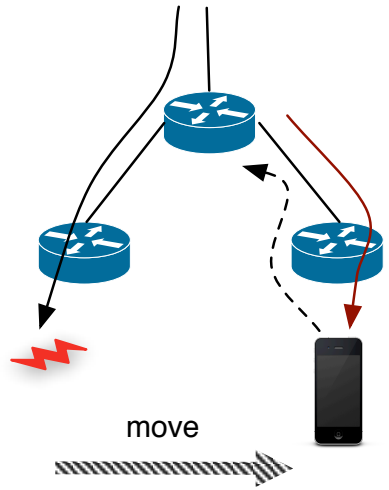
- Data can be easily cached anywhere
  1. fast recovery for packet drops at last (wireless) hop
  2. smooth handoff without special optimization
  3. always fetch popular content from nearest cache or repository



(1) last hop recovery



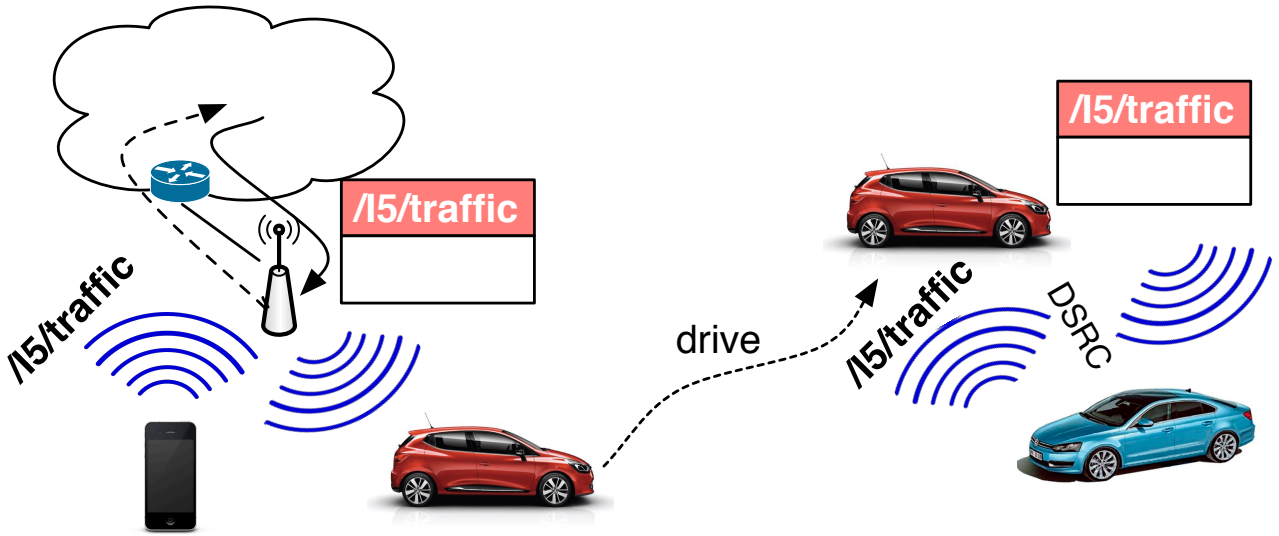
(2.a) optimization for handoff in IP



(2.b) handoff in NDN

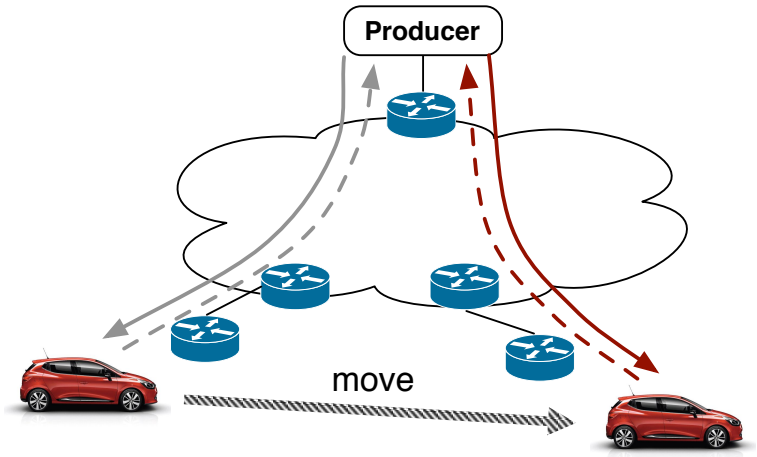
Naturally support DTN and ad hoc networks

- NDN already provides two basic building blocks for DTN and ad hoc networks
  - a way to identify data → every piece of data is named
  - a way to carry data around → data is secured and can be cached anywhere
- Furthermore, NDN fully utilizes the broadcast nature of wireless communications
  - Interest can be broadcasted and whoever has the data can reply
  - Overheard data can be cached



Naturally support mobile consumers

- IP is about connectivity
  - has to acquire IP address and report IP address changes so that the other end can "push" data
- In NDN, data flows back along the reverse path of the Interest
  - no need to acquire names
  - no need to report location changes

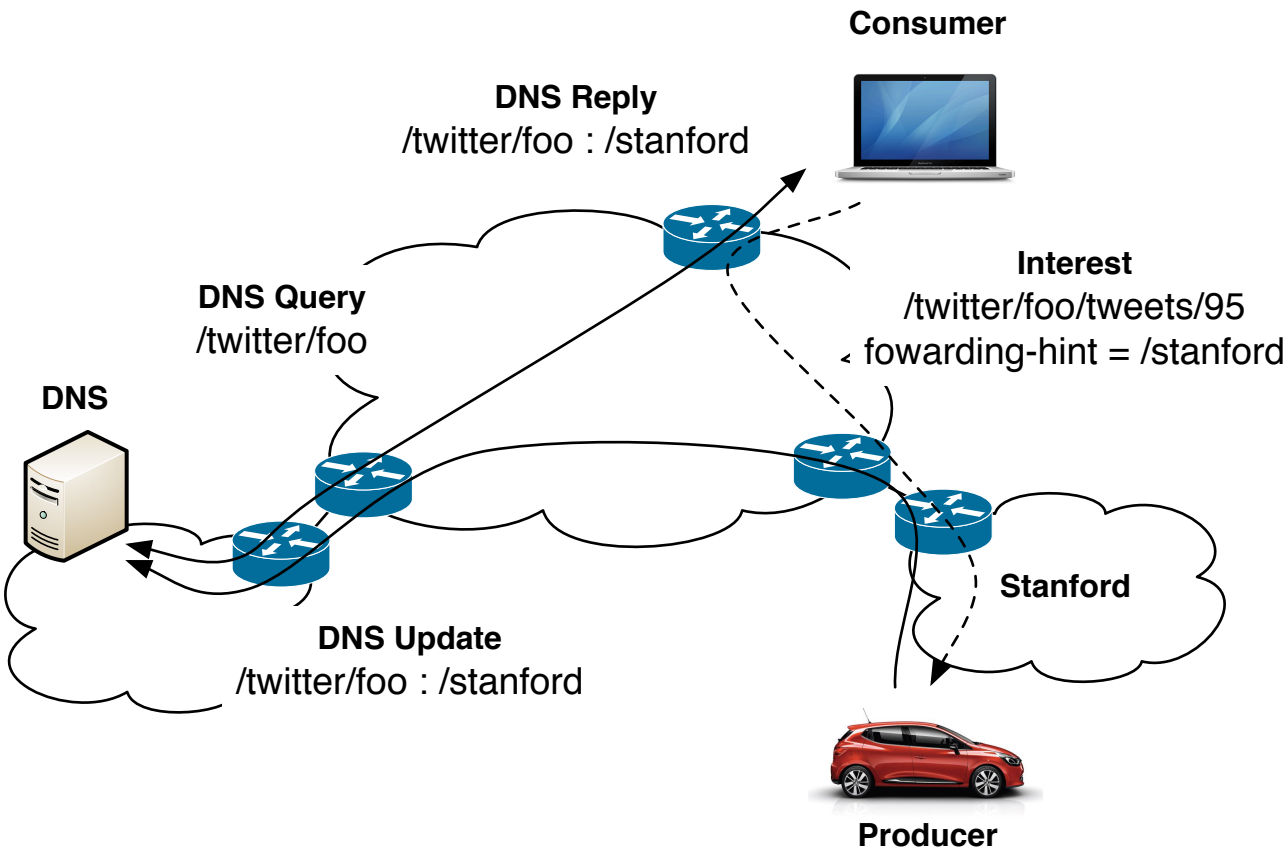


Support for mobile producers

- The network needs to route Interests based on data names
  - producers whose names are not on FIB must have a way for Interests to reach them

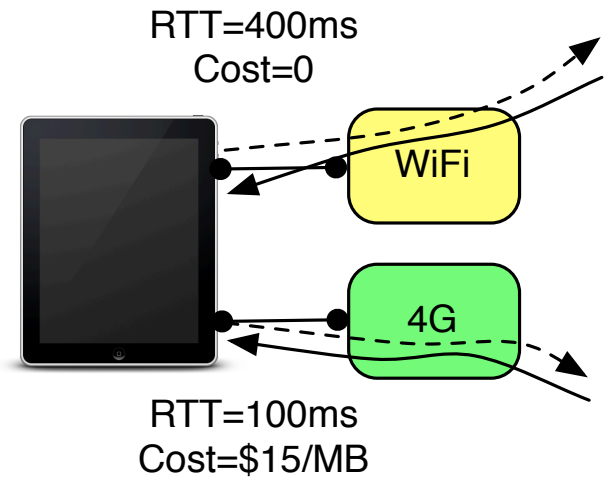
- The lessons we learned from IP mobility research can be applied
  - stable identifier: the name space assigned to a producer
    - e.g. /ndn/ucla.edu/cs/foo
  - locator: the name prefix that hints the location of a producer
    - e.g. /ndn/stanford.edu
  - mapping:
    - broadcast: simplest and robust, and takes advantage of broadcast nature of wireless communications
    - DNS: one can do DNS lookup to find the mobile

An example of DNS based support for mobile producer

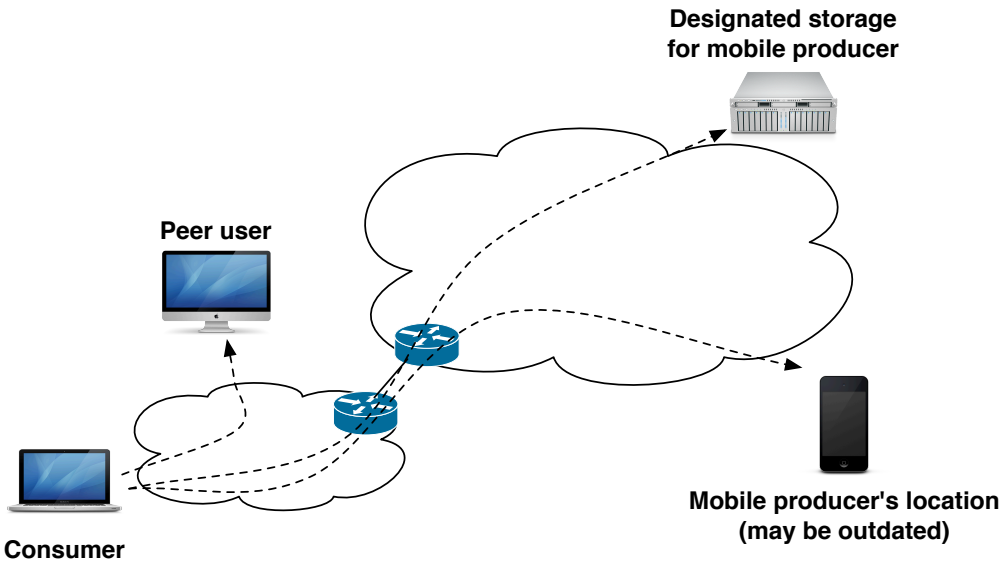


Flexible data fetching

- NDN's *forwarding strategy* provides the much desired flexibility in fetching mobile producer's data
  - enabled by packet statistics and loop-free Interest forwarding
  - also take into considerations of each face's properties (e.g. broadcast interface, traffic cost) and namespace



(a) utilizing multiple interfaces



(b) exploring alternative path

Publications on mobility support

- Supporting mobility in the global Internet (ACM MobiCom MICNET Workshop '09)
- Supporting mobility for Internet cars (IEEE Communications Magazine 2011)
- SAIL: A scalable approach for wide-area IP mobility (IEEE Infocom MobiWorld Workshop '11)
- Home as you go: an engineering approach to mobility-capable extended home networks (AINTEC '11)
- A survey of mobility support in the Internet (IETF RFC 6301)
- Understanding Apple's back to my mac service (IETF RFC 6281)
- A new perspective on mobility support under submission to MobiArch '13

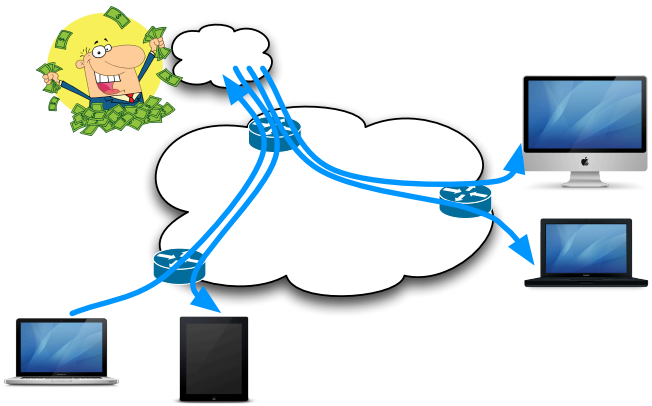
Exploring new design patterns for distributed applications over NDN

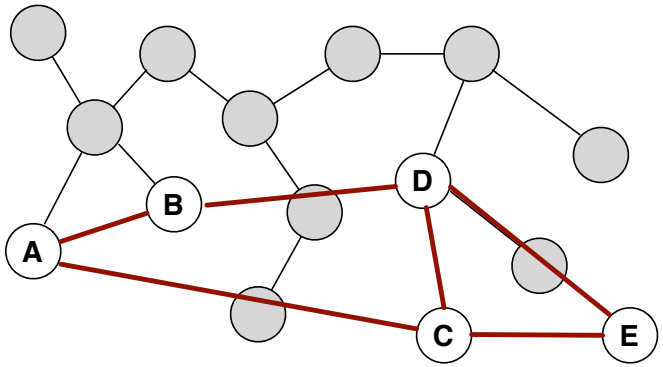
Why new design patterns

- NDN uses application defined names for data delivery
  - so apps can now use names to achieve new functions that they were unable to do before
- Exactly how to design apps to take such advantages is yet to be explored
- NDN research has been taking on an app-driven approach to get the answer

Distributed applications in today's Internet

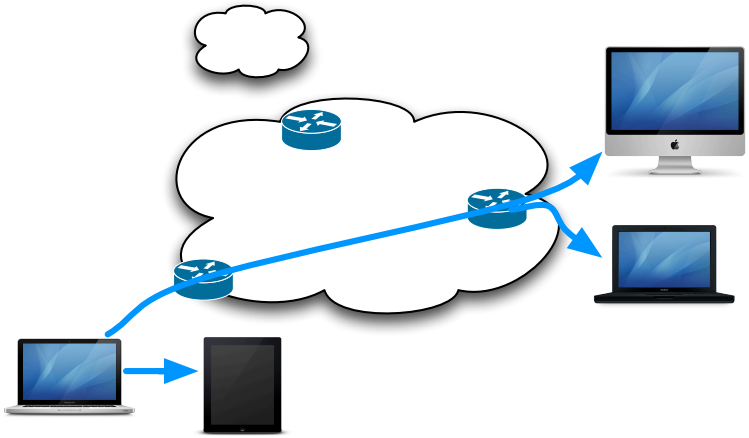
- Centralized designs
  - centralized control
  - single point of failure
  - inefficient data distribution
- Peer-to-peer designs
  - efficient data distribution may not be the goal
  - mis-matching between P2P overlay and underlying topology



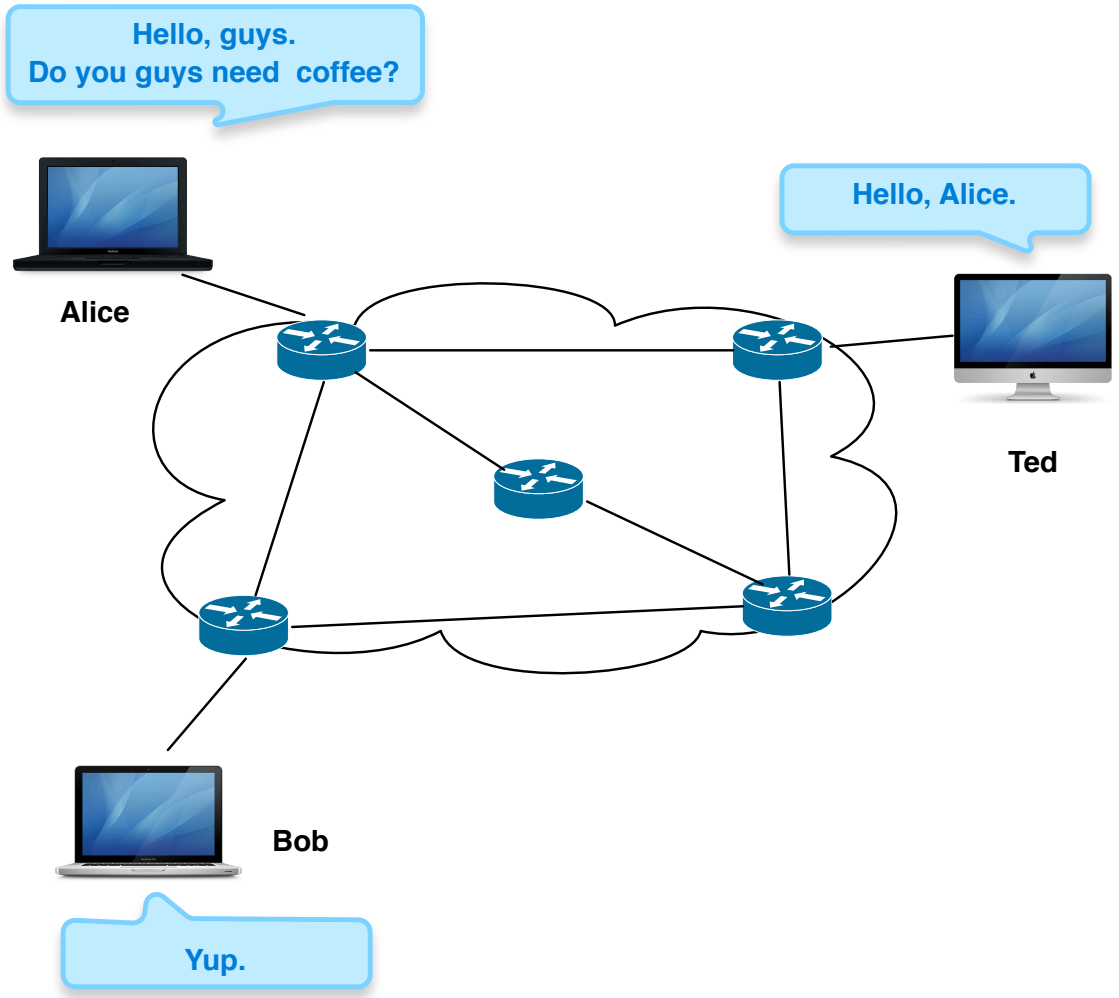


Distributed applications over NDN: Goals

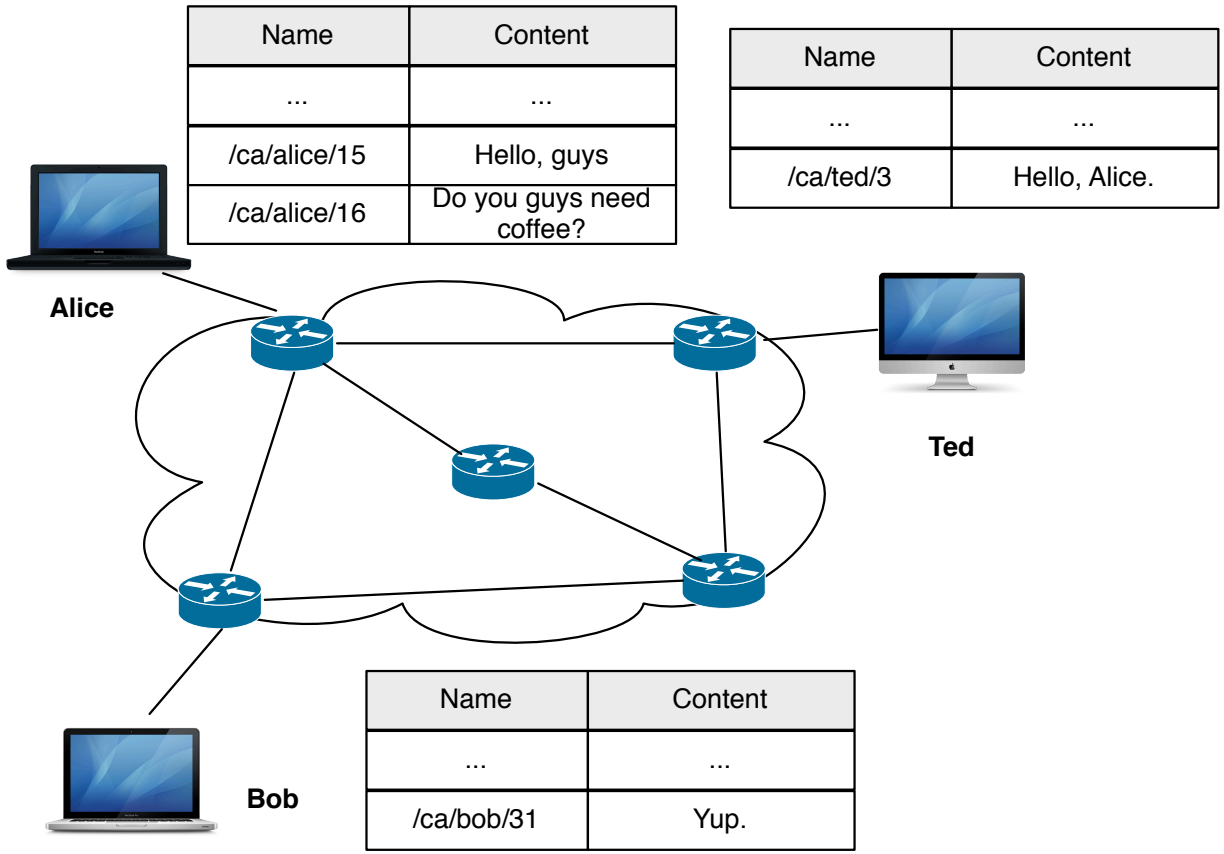
- Robust, fully decentralized
- Efficient data distribution
- Mobile friendly
- Secure



A simple distributed application: group text chat



A simple distributed application: group text chat



The problem is really synchronizing the chat dataset among the three.

Decentralized dataset synchronization

- The need is ubiquitous in distributed applications
  - retrieving all messages to a chatroom
  - collecting a list of conferences and speakers in audio conferencing
  - collecting all revisions in collaborative editing

- The problems:
  - synchronizing a dataset produced by multiple parties in an unpredictable pattern
  - no pre-knowledge about parties

The role of naming

- Naming is an important aspect of NDN application design
- Names determine how Interests are forwarded in the network
  - and also identifies the application process if it reaches the producer
- Proper naming can often simplify application designs

Naming in dataset synchronization

- Use broadcast namespace for rendezvous
  - Interest carrying broadcast name will be broadcasted in a domain
  - reaches all potential data producers

/ndn/broadcast/chronos/lunch-talk/4b01...

(1) (2) (3)

- Use routable namespace for actual data items
  - use broadcast names for all data would be prohibitively expensive

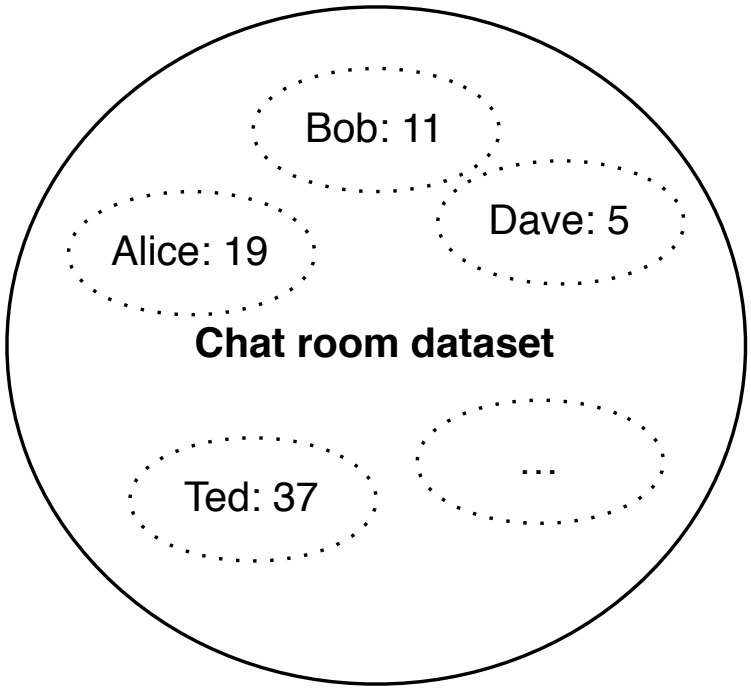
/wonderland/alice/chronos/lunch-talk/791

(1) (2) (3)

- Simplify the knowledge about a producer's data
  - e.g., name the data of a producer sequentially

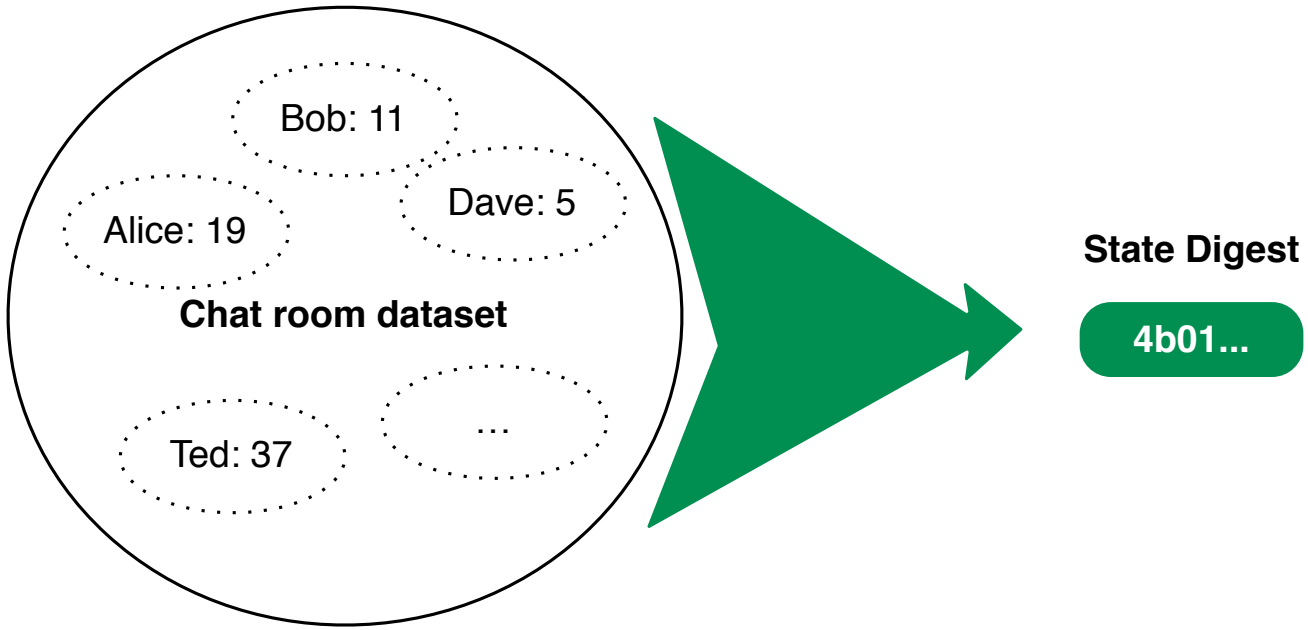
ChronoSync: decentralized dataset state synchronization

- Maintain knowledge of a dataset by tracking what each producer has produced so far



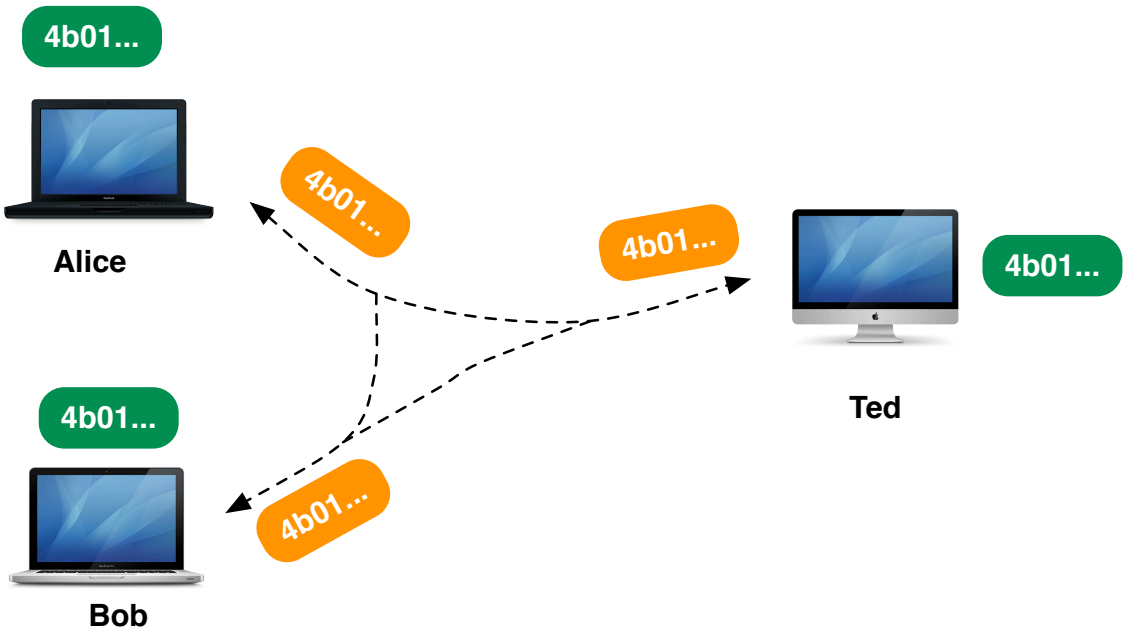
ChronoSync: decentralized dataset state synchronization

- Maintain knowledge of a dataset by tracking what each producer has produced so far
- Compactly represent a user's knowledge of the dataset in a hash, or state digest



ChronoSync: decentralized dataset state synchronization

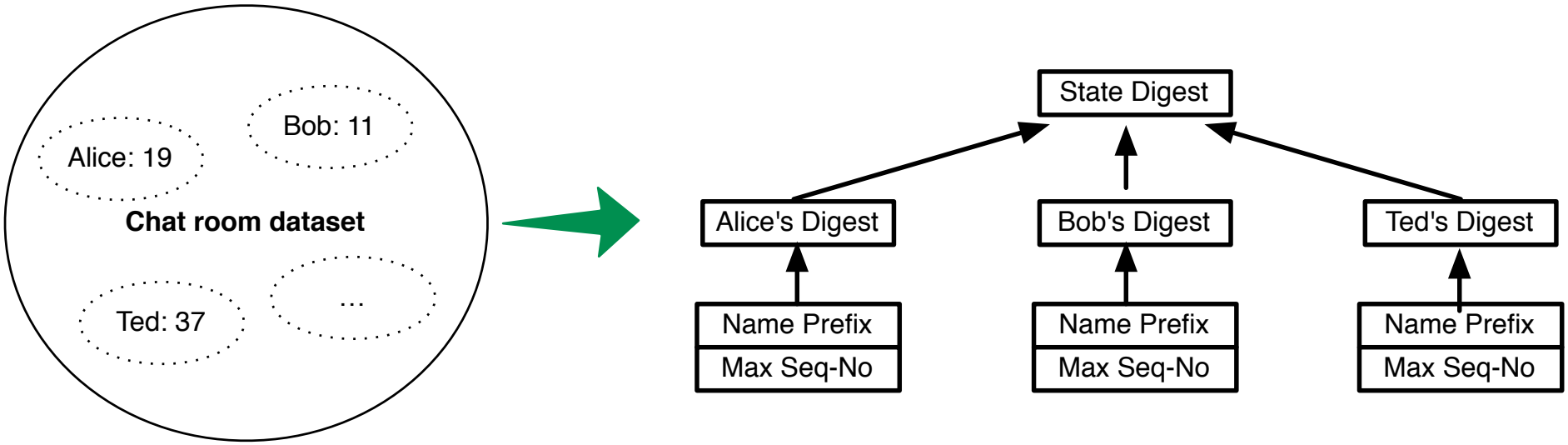
- Maintain knowledge of a dataset by tracking what each producer has produced so far
- Compactly represent a user's knowledge of the dataset in a hash, or state digest
- Exchange state digests among all users using broadcast (sync) Interests



/ndn/broadcast/chronos/lunch-talk/4b01...

(1) (2) (3)

Tracking dataset state: digest-tree



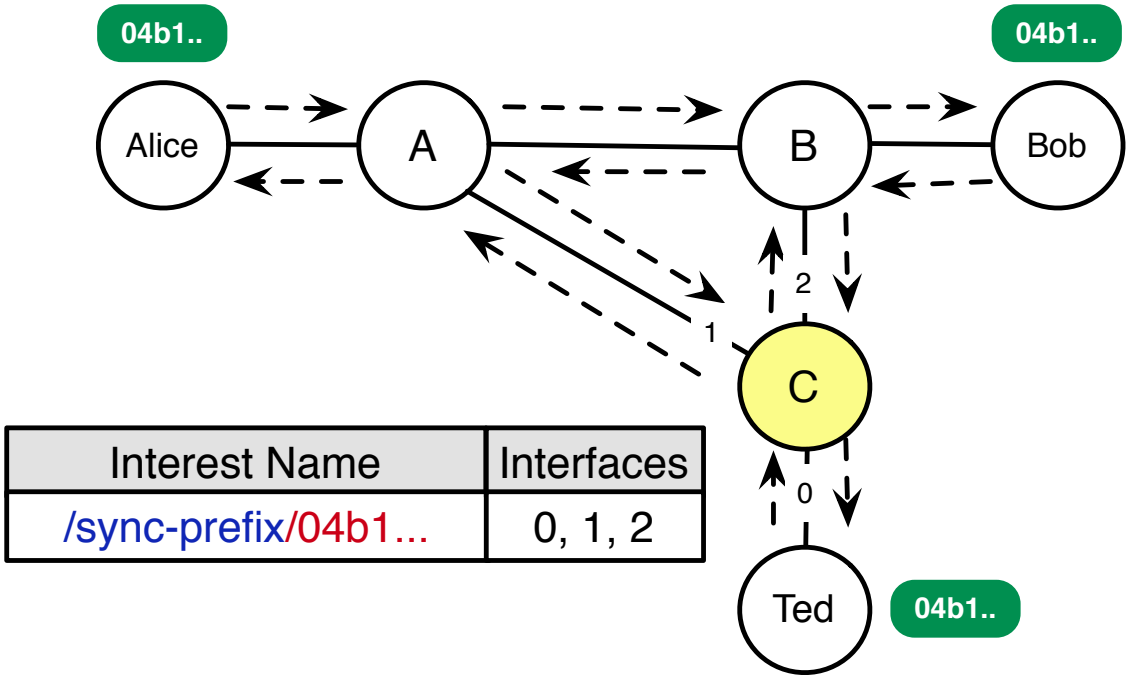
- Each child node maintains the knowledge of a producer
  - sorted according to the name prefix of each producer
- Hash children nodes in order to produce state digest
- When a producer's state changes, update the branch and re-calculate state digest

Tracking dataset state: digest-log

State Digest	Changes
0000...	Null
9w35...	[Alice's prefix, 1]
...	...
23ab...	[Bob's prefix, 31], [Alice's prefix, 19]
05t1...	[Bob's prefix, 32]

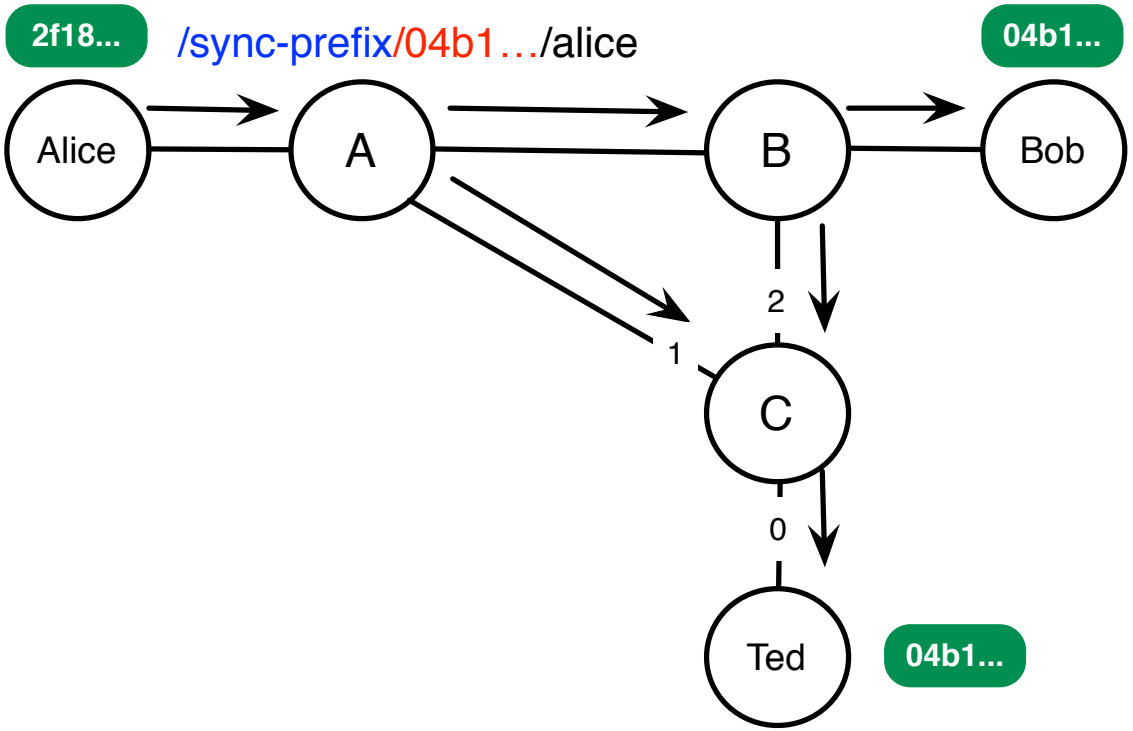
- Maintaining the history of state digest changes
  - useful when old state digest is received, e.g. from a user who just recovers from a temp disconnection
- Not essential for the correctness of ChronoSync
  - application can set an upper bound of log size

Stable state



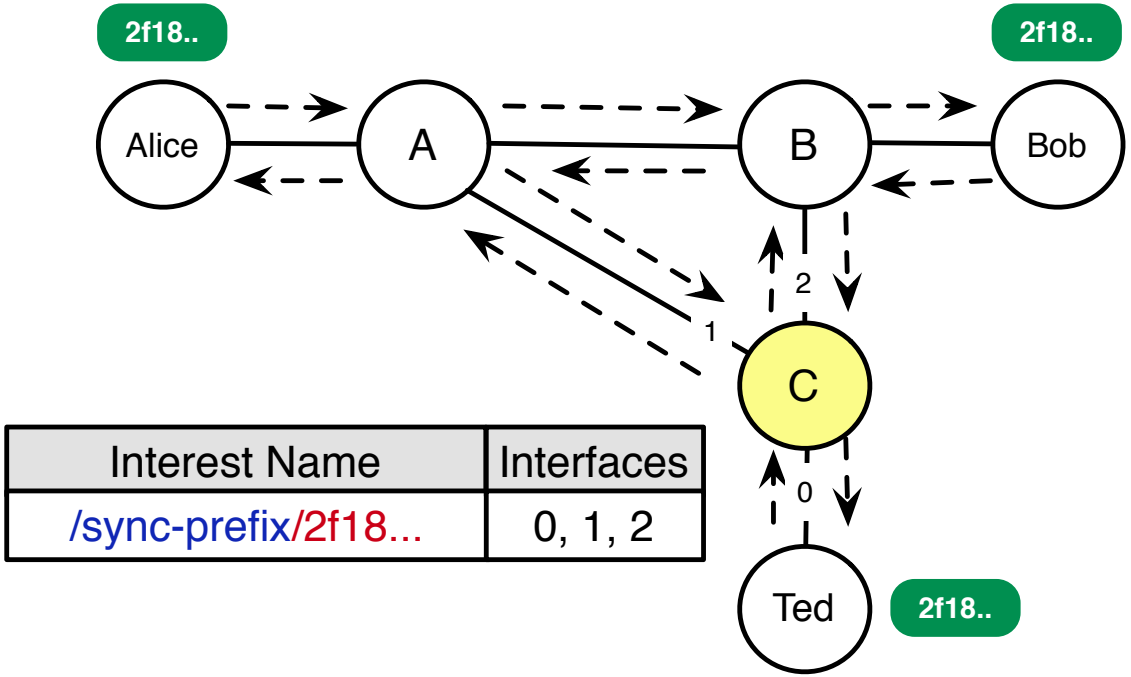
- Everyone has the same knowledge of a dataset
- Sync Interests carry an identical state digest
  - regardless of number of users in a group, at most one sync Interest sent over a link in one direction

Disseminate knowledge of new data



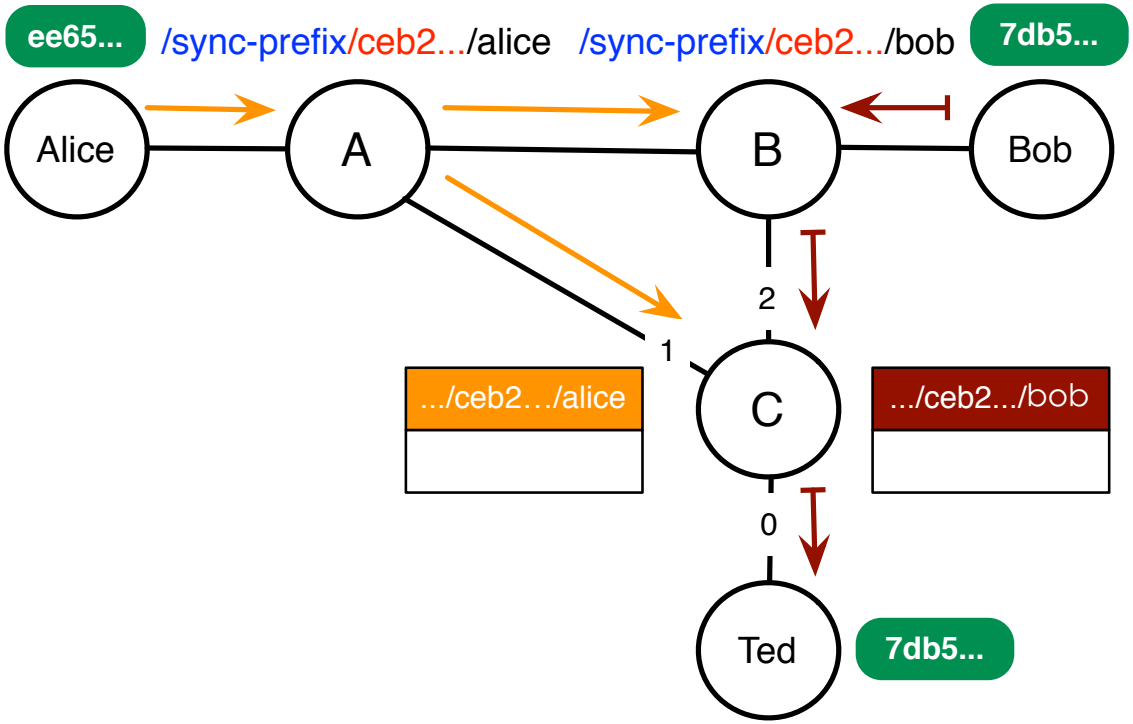
- After producing new data, a producer observes different state digest
  - reply the pending sync Interest with the new data info
- Others update their state digest once receiving the reply
  - producing the same new state digest

Disseminate knowledge of new data



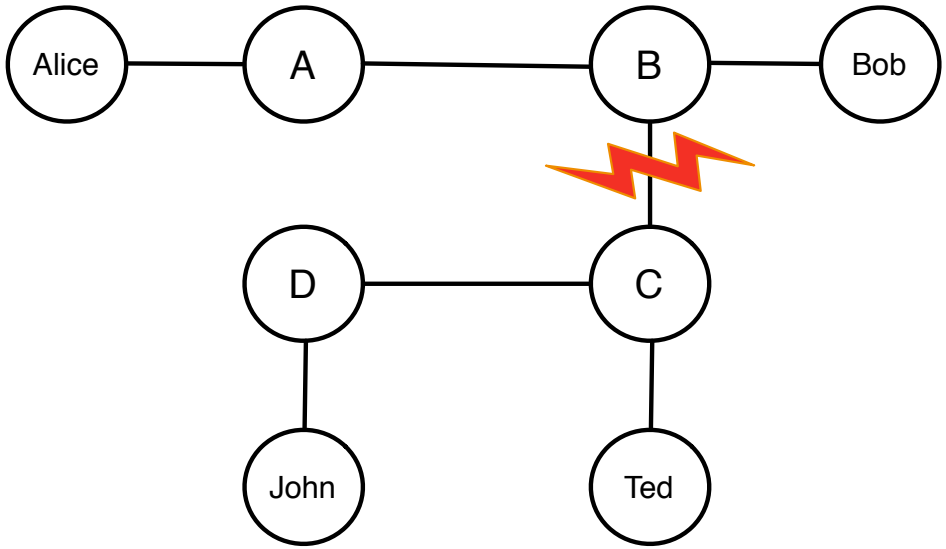
- After producing new data, a producer observes different state digest
  - reply the pending sync Interest with the new data info
- Others update their state digest once receiving the reply
  - producing the same new state digest

Handle simultaneous data generation



- Problem: one sync Interest can only retrieve one sync data
  - users are partitioned into groups with different knowledge about the dataset
- Optimization: use exclude filter
  - need to exclude at most $(N-1)$ known sync data if N producers generated data simultaneously

Problem: network partitions



State Digest	Changes
...	...
9w35...	[Alice's prefix, 1]
23ab...	[Bob's prefix, 31]
07t1...	[Alice's prefix, 2]
ks23...	[Alice's prefix, 3]

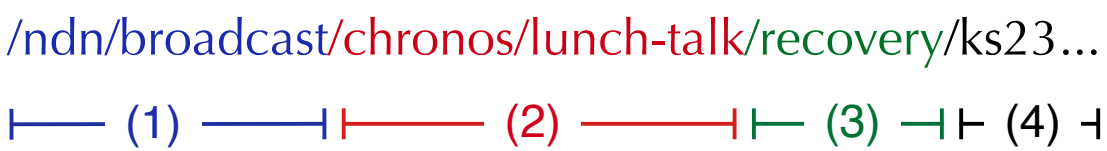
Alice and Bob's view

State Digest	Changes
...	...
9w35...	[Alice's prefix, 1]
ux53...	[Ted's prefix, 9]
ik29...	[John's prefix, 0]
990s...	[John's prefix, 2]

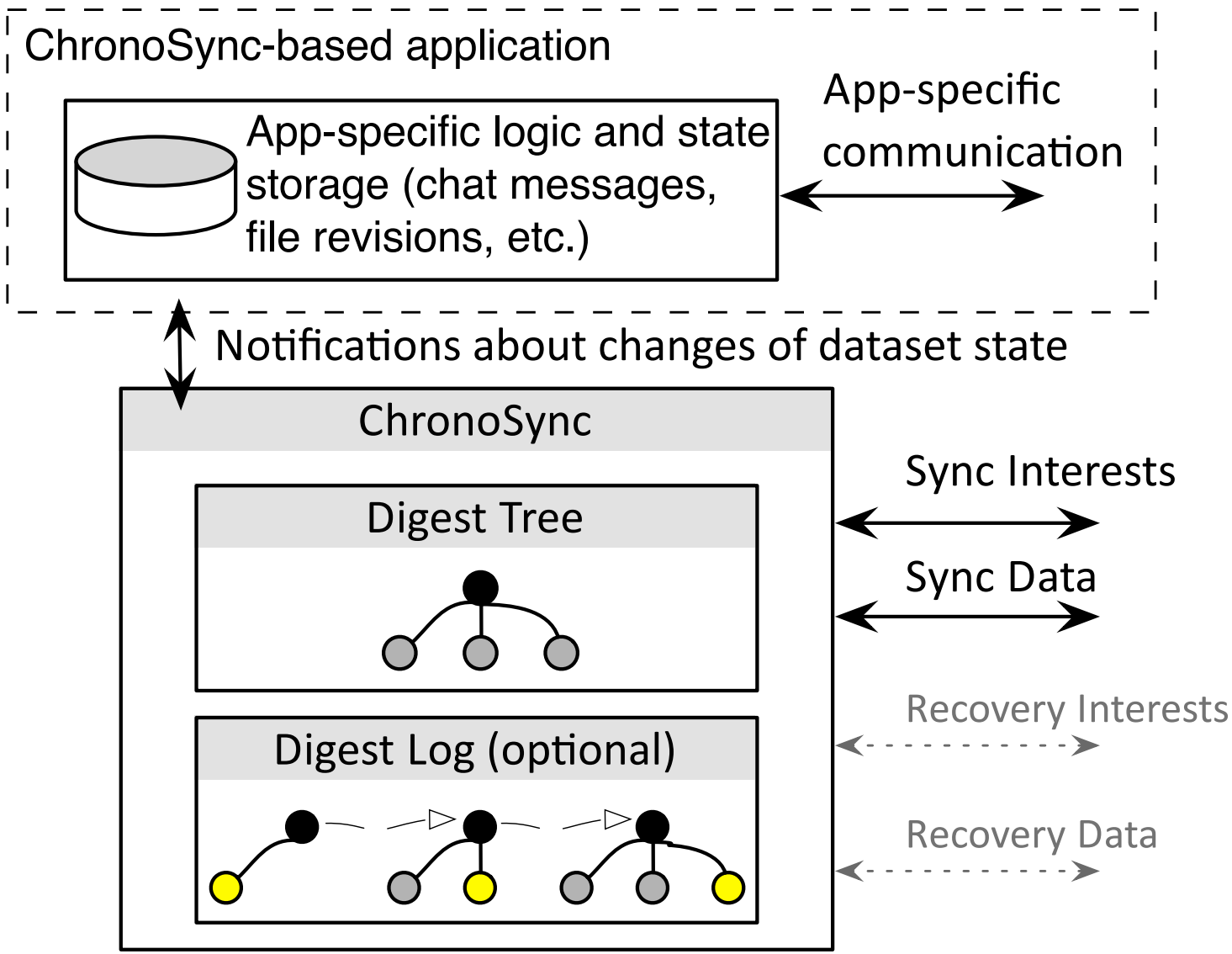
John and Ted's view

Handling network partitions

- Any set reconciliation algorithm can be used
- A simple recovery method is used as default
  - explicitly asking for information from the users who produced the unknown state digest
  - recovery data contains the whole snapshot of the dataset state (i.e. information about each data producer)

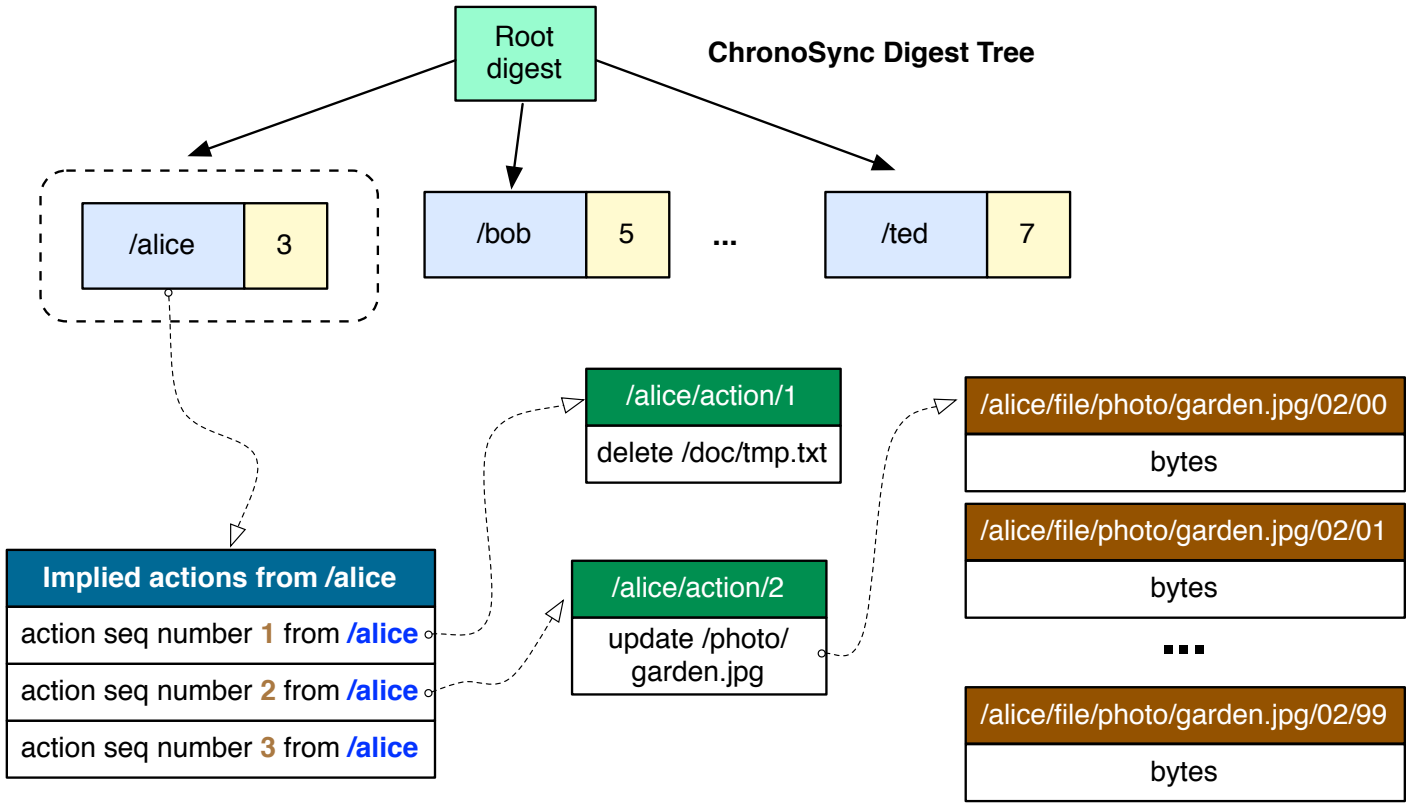


Overview of ChronoSync



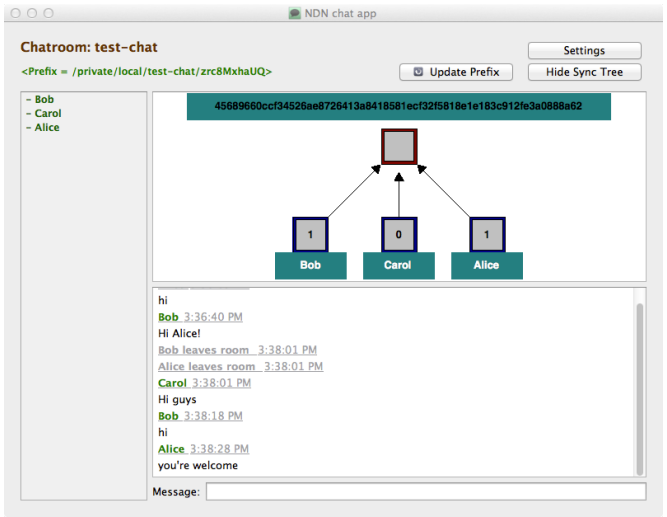
ChronoShare: distributed file sharing with ChronoSync

- Starting from an empty shared folder, the state of the folder is altered whenever an action from a participant is applied
- The state of the folder can be determined by applying all actions from all participants in a deterministic way
- ChronoSync can be used track the actions



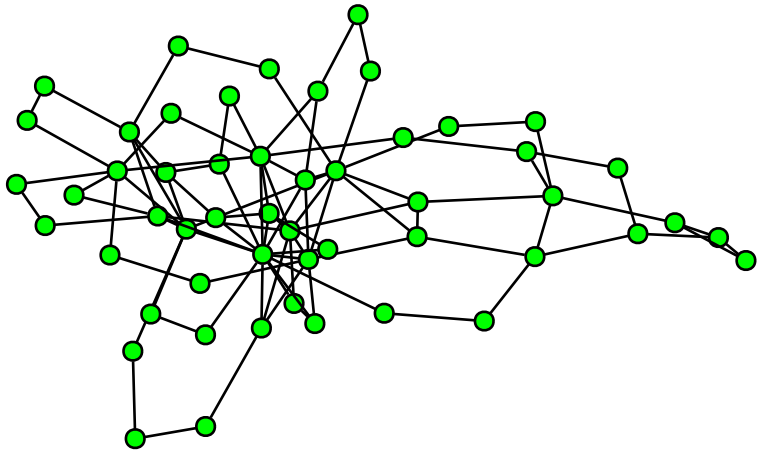
Evaluation

- Goals: to confirm
  - ChronoSync synchronize dataset state quickly and efficiently
  - ChronoSync is robust again network partitions, link failures and packet losses
- Methodology
  - simulation in ndnSIM module of NS-3
  - evaluate performance of ChronoChat, a group text chat app implemented based on ChronoSync
  - use central server based TCP/IP implementation of IRC as baseline



Evaluation setup

- Topology: 52 nodes, 84 links [1]
  - 100 Mbps bandwidth for each link
- Traffic: 1000 messages following exponential distribution with mean of 5 seconds [2]
- All nodes participate in one chatroom

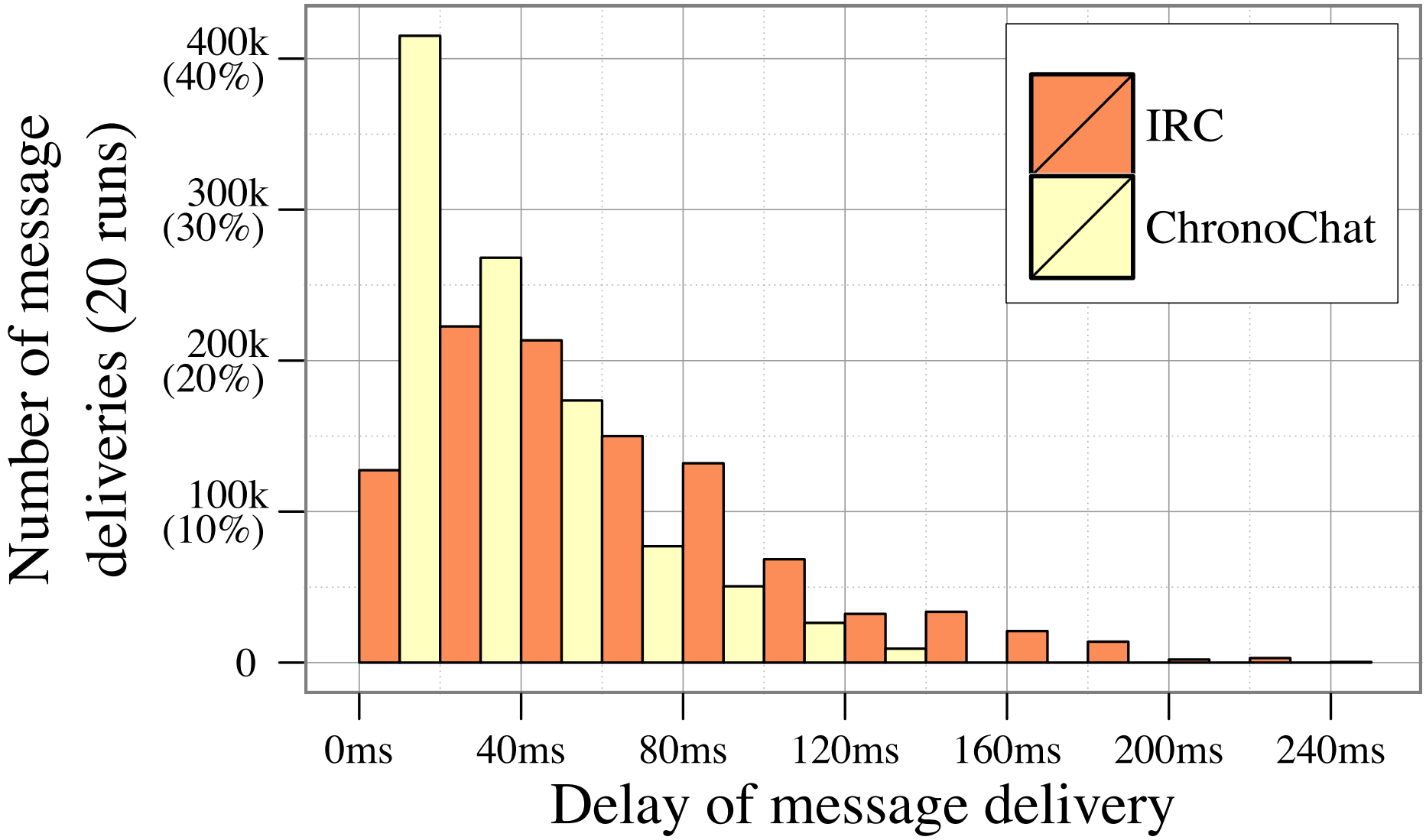


[1] N. Spring et al., Measuring ISP topologies with Rocketfuel.
[2] C. Dewes et al., An analysis of Internet chat systems.

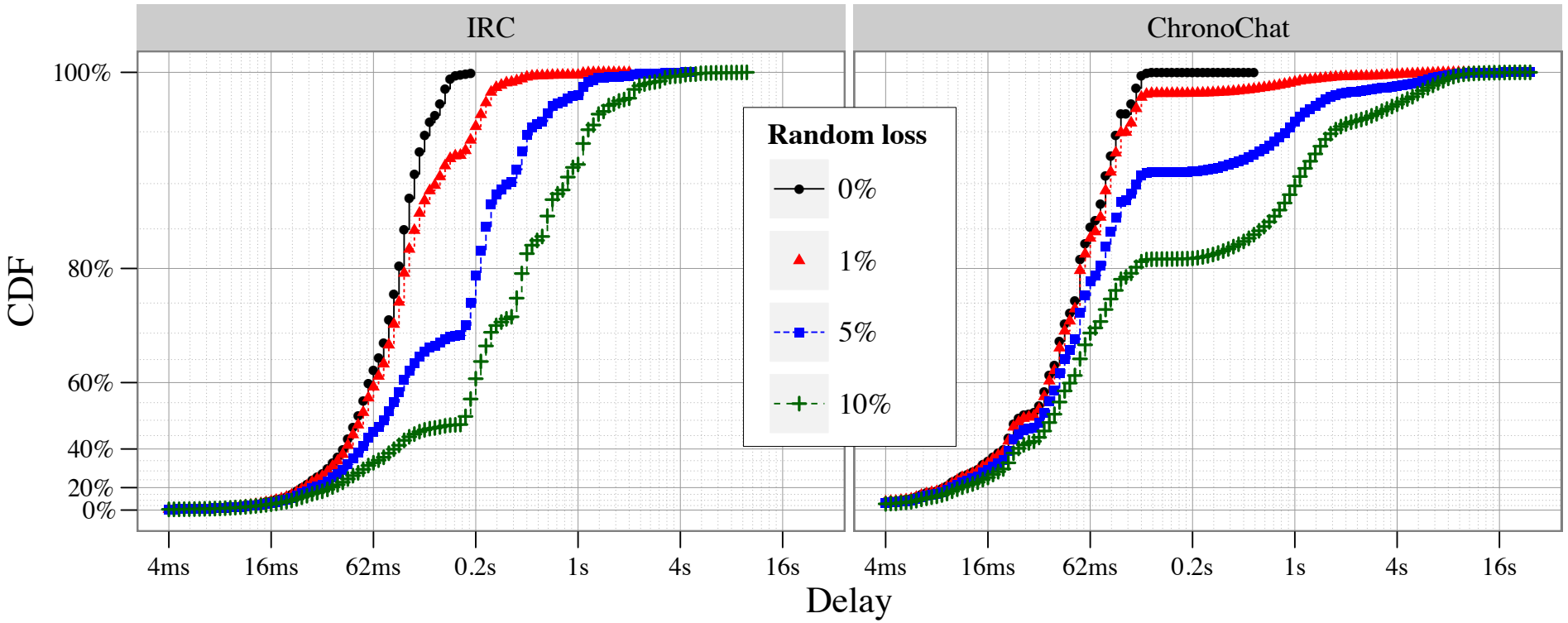
Metrics

- Synchronization delay: the time needed for the last user to learn a new data piece after its generation
- Communication pairs: the pairs of users that are still able to chat in face of network failures
- Traffic pattern: how is the traffic distributed over the links
- Total overhead: how many packets are transmitted

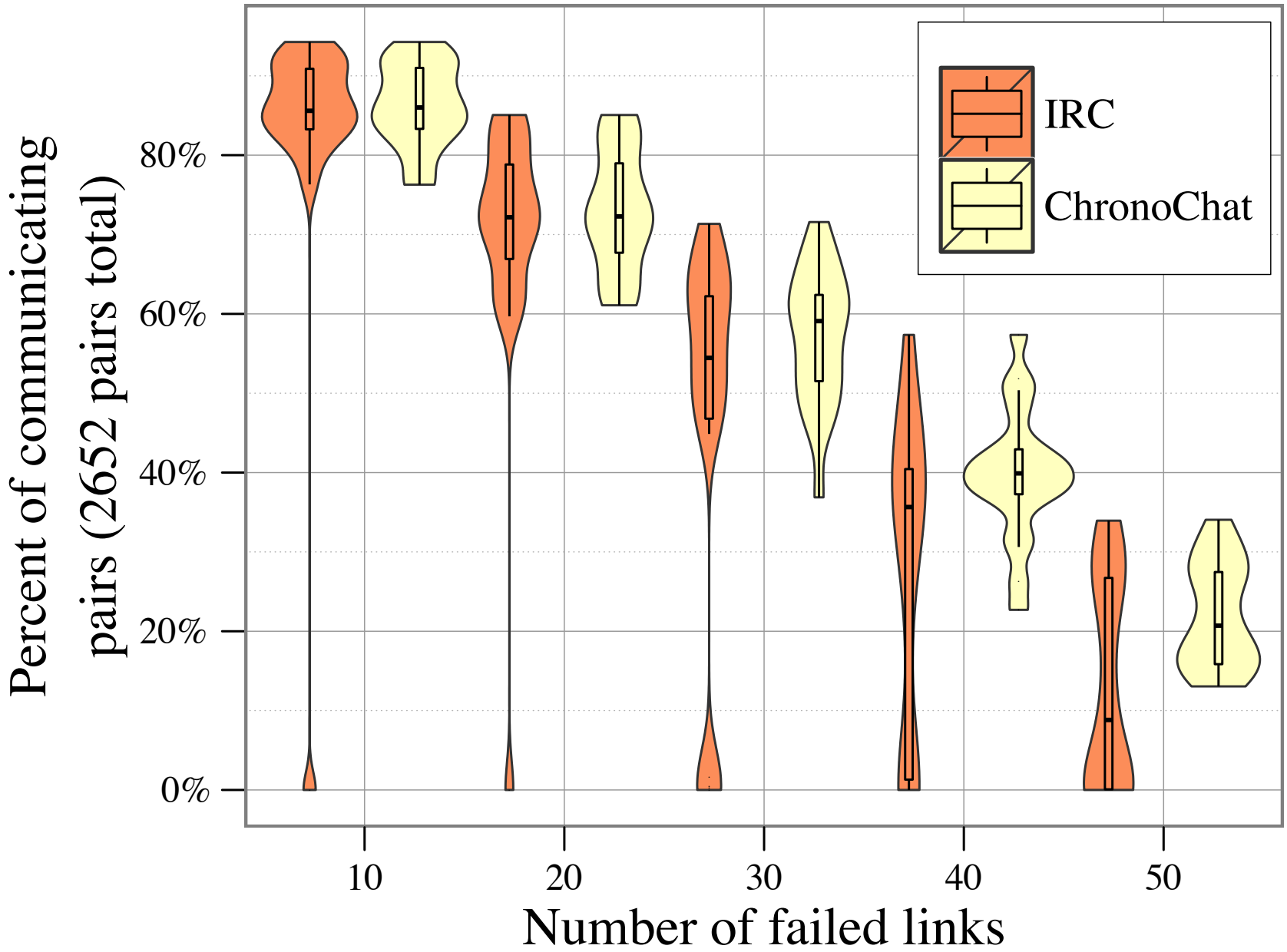
Synchronization delay



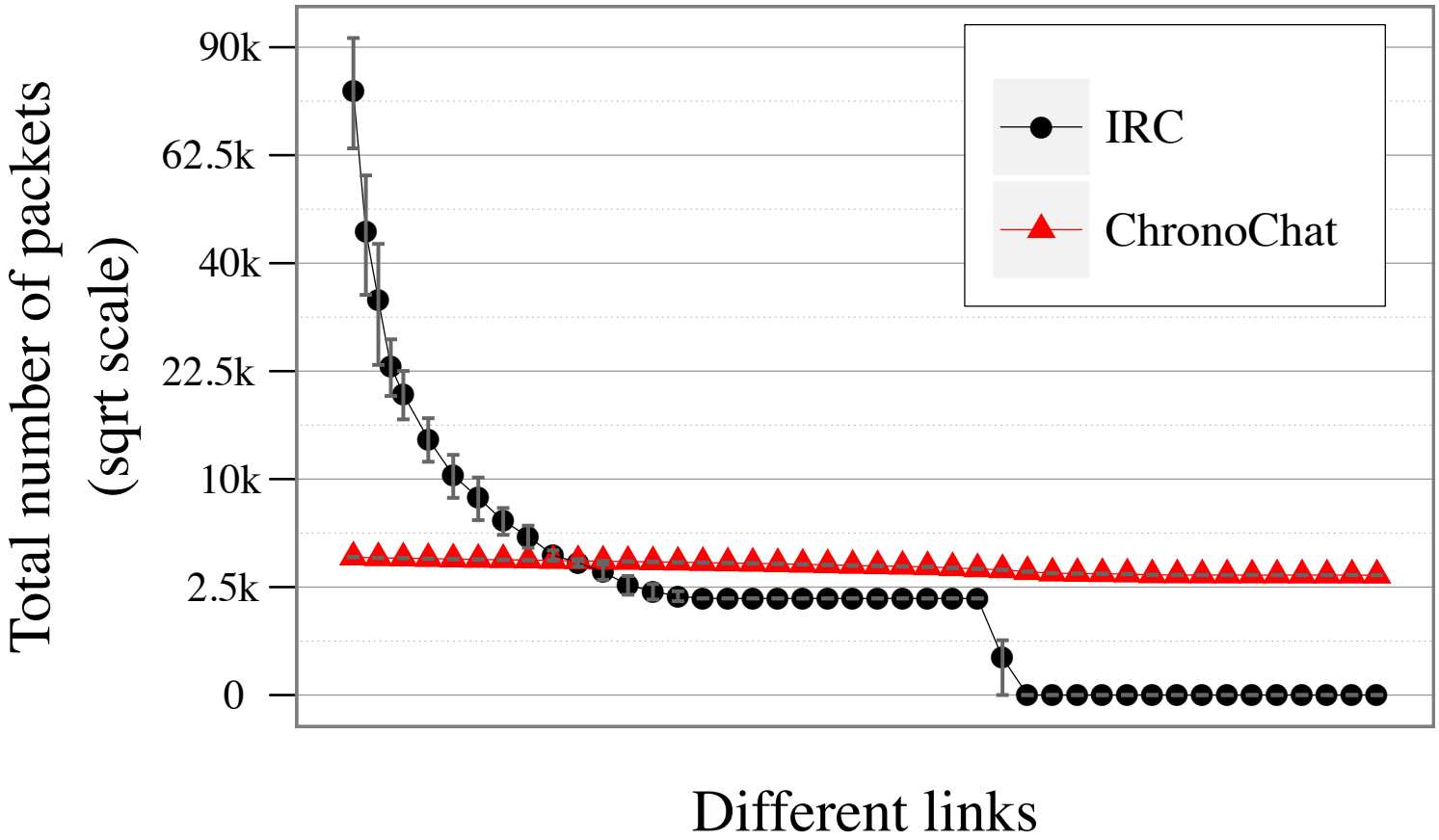
Synchronization delay under lossy environment



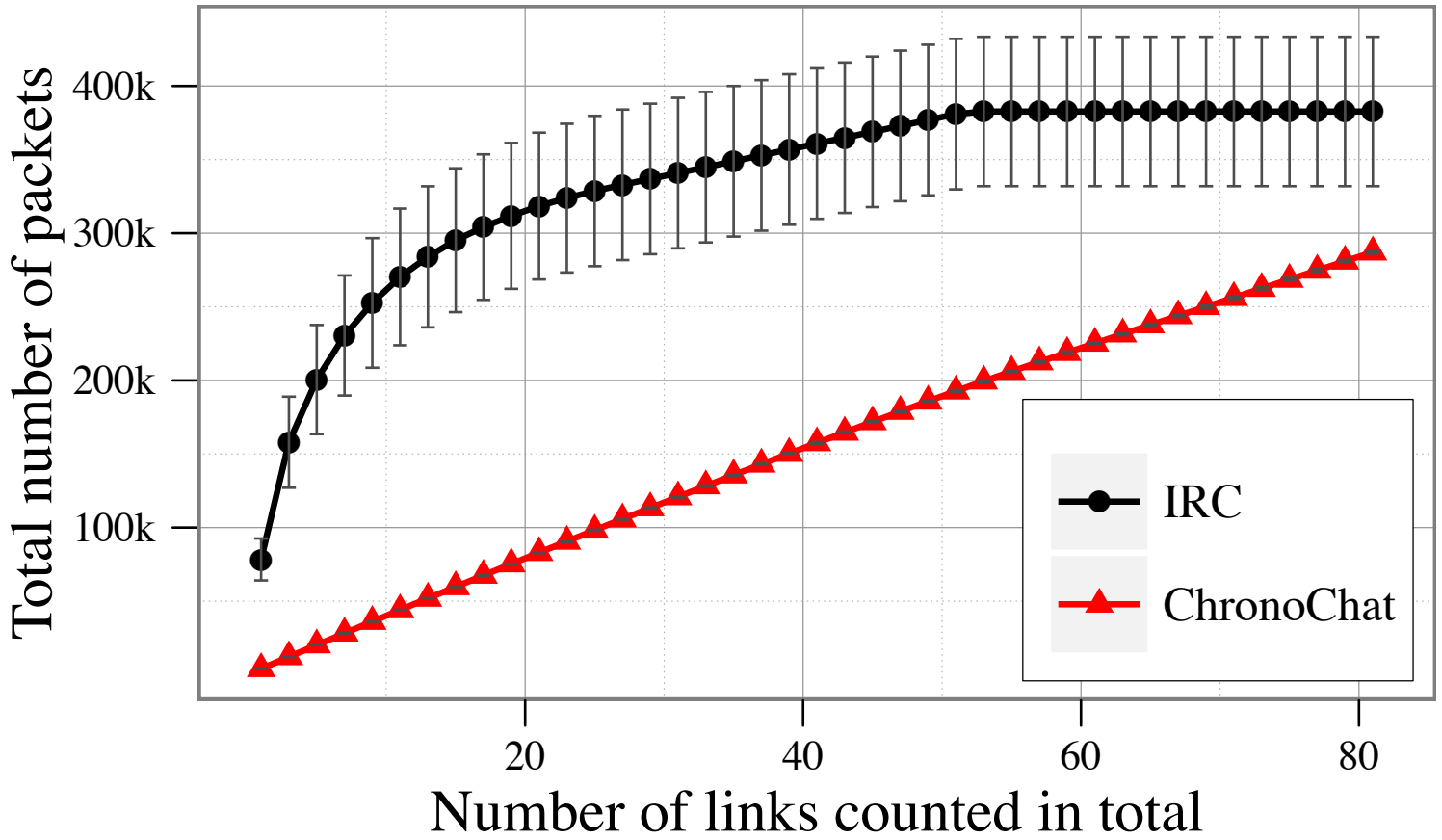
Resilience to network failures



Traffic pattern

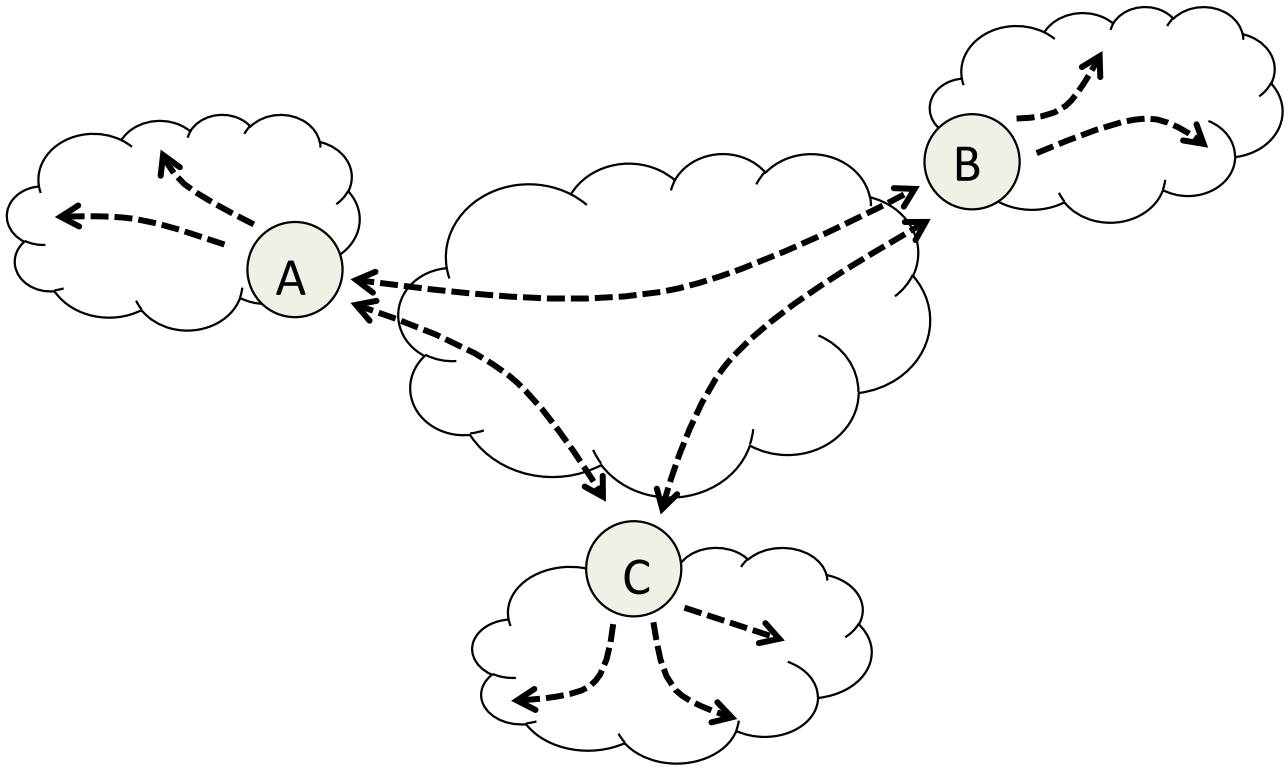


Total overhead



Discussion: broadcast in large networks

- Broadcast directly in large networks is costly
- A broadcast overlay can dramatically reduce the cost



Publications

- ACT: an audio conference tool over Named Data Networking ACM SIGCOMM ICN Workshop '11
- A new approach of securing audio conference tools AINTEC '11
- Chronos: serverless multi-user chat over NDN NDN Tech-Report 0008, 2012
- Let's ChronoSync: decentralized dataset state synchronization in Named Data Networking under submission

Software

- ACT: multiple party audio conference tool
- XMPP-MUC: multi-user text chat
- ChronoChat: Chronosync based multi-user text chat
- ChronoShare: ChronoSync based file sharing application
- peets: WebRTC based audio/video conferencing
- ChronoSync: C++ library
- py-chronos: python API for ChronoSync
- NDN.cxx: C++ API for NDN
- ndndump: packet analyzer for NDN

Concluding thoughts

- The Internet needs changes to accommodate billions of mobile devices and dramatically changed communication patterns.
- By aligning mobile communications with application's data-centric nature, we provided a completely different perspective on mobility support
- ChronoSync leverages NDN's naming and data multicast capability to achieve efficient and robust dataset state synchronization
  - effectively names the state of a dataset by its given digest at a given time
  - removes single point of failure and traffic concentration problems
  - provides useful building blocks in supporting distributed applications
- Look forward, we hope this work can help stimulate more discussions on the design space of mobile and distributed applications over NDN

Acknowledge

- Alexander Afanasyev made significant contribution to the design and implementation of ChronoSync. He is also a collaborator on ChronoShare and NDN.cxx.

- Chaoyi Bian made significant contribution to XMPP-MUC