CS551
Distributed Hash Tables

Unstructured Systems

Bill Cheng

http://merlot.usc.edu/cs551-f12

Distributed Hash Tables

- **Idea is easy, and defined by the interface**
  - ***put(key, data)*** **stores a data item with the specified key**
  - ***get(key)*** **retrieves data item(s) corresponding to key**
  - **key is usually a *hash of data contents***
- **Implementation is distributed over the wide area**

Uses of DHTs

- A network-wide structure can enable a wide variety of applications
  - file sharing
  - distributed file systems
  - anonymous publishing systems
  - flexible rendezvous for multicast applications

DHT Implementation

- Usually implemented as an overlay network
- A special class of overlays: ***content-addressable overlay networks***
  - a document is accessed using a descriptive title of the content rather than the location where the document is stored
    - a data object is represented by a point in a key space
  - at the core lies the distributed algorithm for content lookup and dissemination
- Why distributed?

Structured vs. Unstructured

- Structured system
  - *there is global consensus on which network node a document is stored*
    - algorithmic mapping between document key and node identifier
  - example
    - CAN, *CHORD*, Tapestry, Pastry
- Unstructured system
  - no such consensus exists
    - document key and node identifier are irrelevant
  - example
    - Gnutella, *Freenet*

Structured vs. Unstructured (Cont...)

- Structured system
  - advantage
    - *efficient query*
    - *guarantee retrieval of existing documents*
  - disadvantage
    - hard to provide anonymity
    - problem caused by DoS and selective attacks
- Unstructured system
  - advantage
    - *simplicity in network maintenance*
    - *resistant to attacks*
  - disadvantage
    - retrieval cost not bounded
    - search may fail even if the requested data exists

CS551
Freenet
[Clarke02a]

Bill Cheng

http://merlot.usc.edu/cs551-f12

Freenet

- Early peer-to-peer system
- Goals:
  - anonymity for publisher
  - anonymity for consumer
  - deniability for participants in the overlay
- Algorithms for
  - search
  - insertion

Keys in Freenet

- Content-hash key
  - hash the entire content of a file using SHA1
  - very low probability of collision
  - but how do you find a file?
- Signed-subspace key
  - create a container file that describes a collection of files or documents
    - like a directory
    - container file hashed by a descriptive name
  - to access this file, you need to know the name of the container file

Basic Idea: Inserting Data

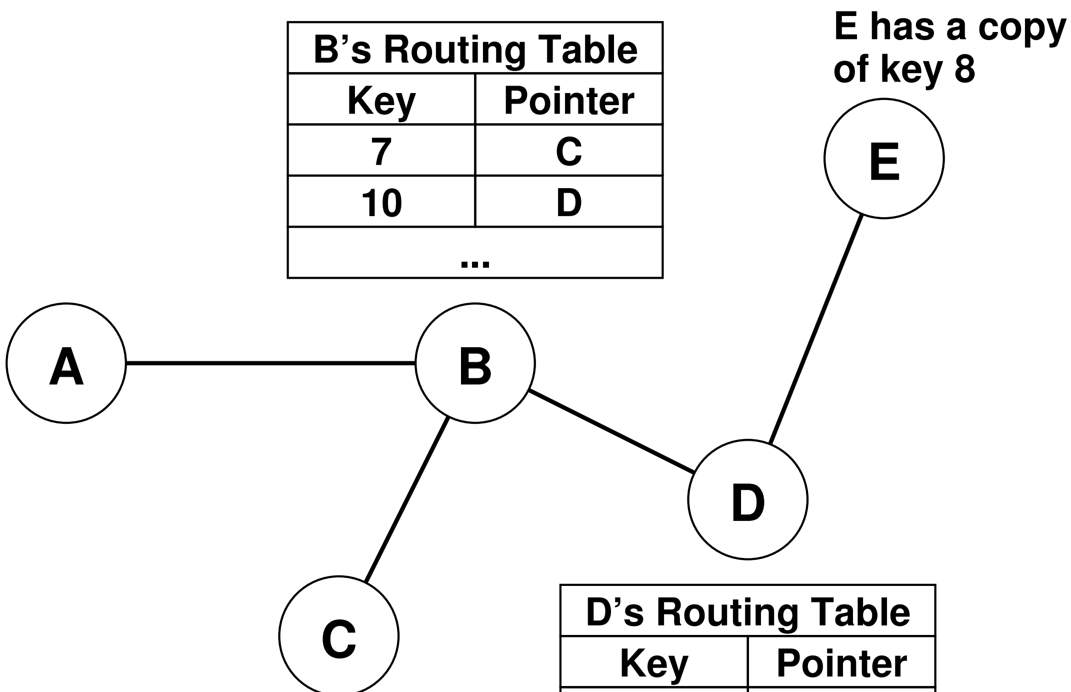
- Each node has routing table that maps keys to neighbors
- Let k be the key of the data item to be inserted
- In routing table, find k’ that is closest to k
  - steepest ascent hill climbing
- Send data item to the associated neighbor
  - use second closest key k'' if this action would cause a loop
- Cache data at each intermediate node

Basic Idea: Retrieving Data

- Algorithm similar to insertion
- Data retrieval can fail
  - TTL might be exceeded without hitting a cached copy
  - we'll see example
- Aggressive caching: as data is being retrieved, intermediate nodes cache copy
- Key to good performance
  - over time, different nodes specialize in different parts of the key space (a node is likely to store data items whose keys are near each other)

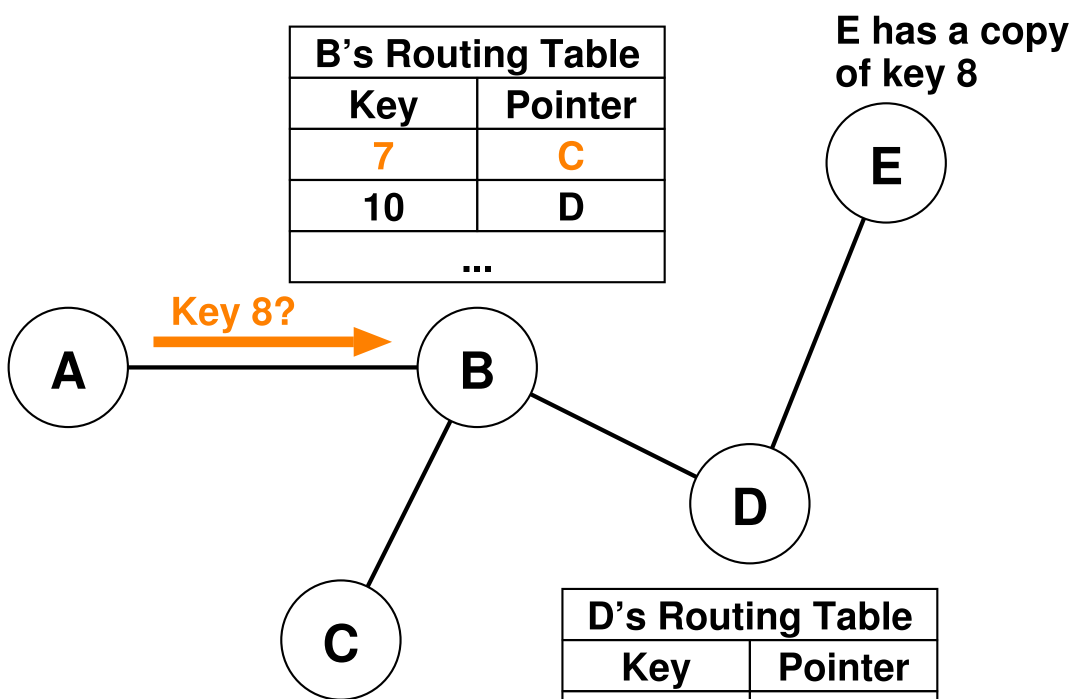
An Example Search in Freenet Network

B’s Routing Table	
Key	Pointer
7	C
10	D
...	



D’s Routing Table	
Key	Pointer
8	E
...	

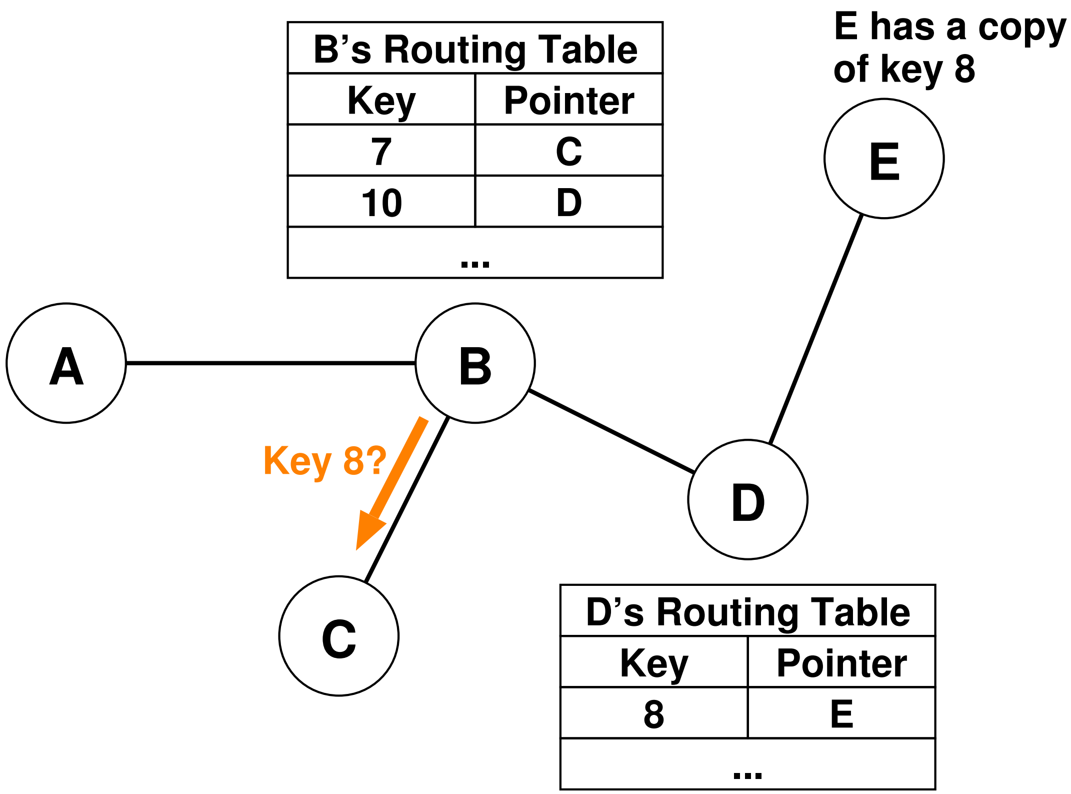
A Search in Freenet Network



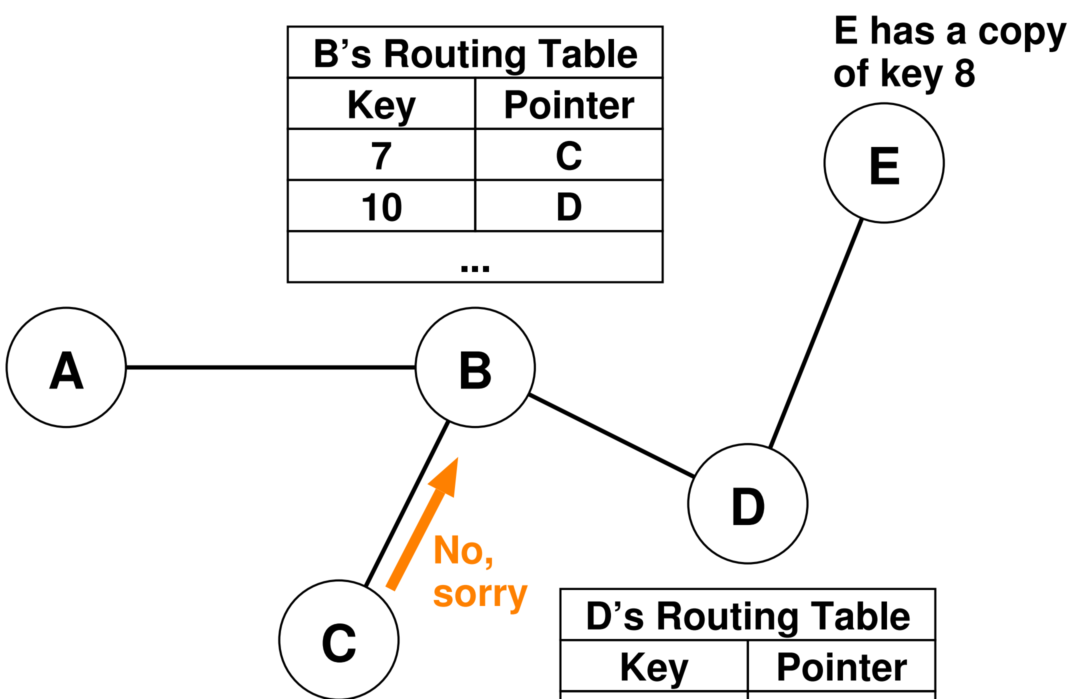
B's Routing Table	
Key	Pointer
7	C
10	D
...	

D's Routing Table	
Key	Pointer
8	E
...	

A Search in Freenet Network



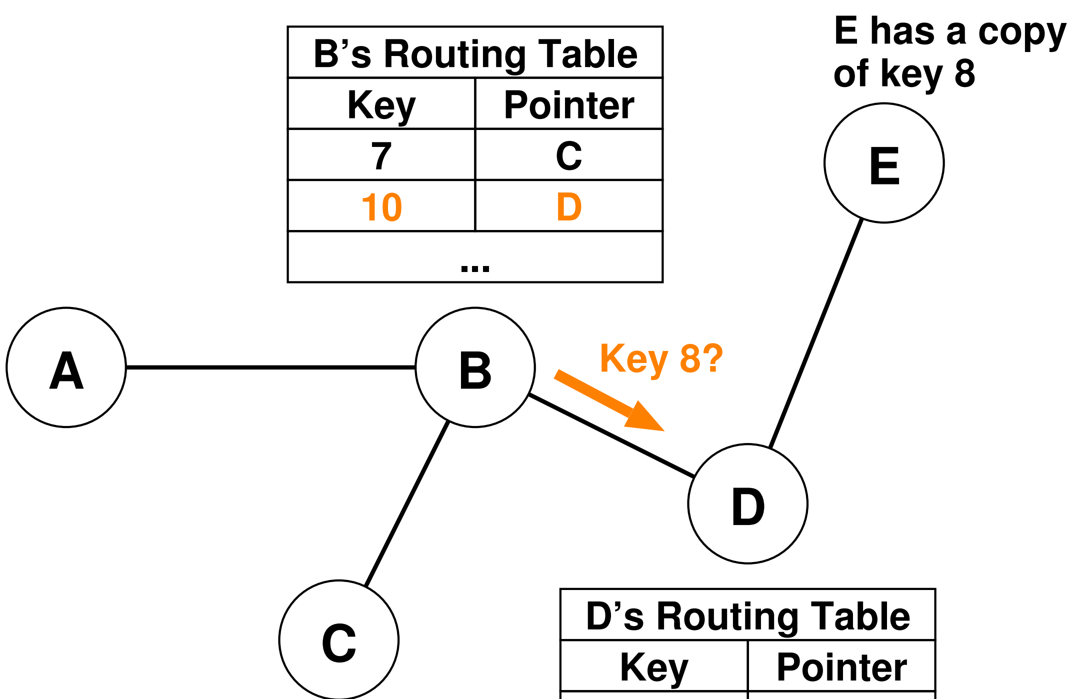
A Search in Freenet Network



B's Routing Table	
Key	Pointer
7	C
10	D
...	

D's Routing Table	
Key	Pointer
8	E
...	

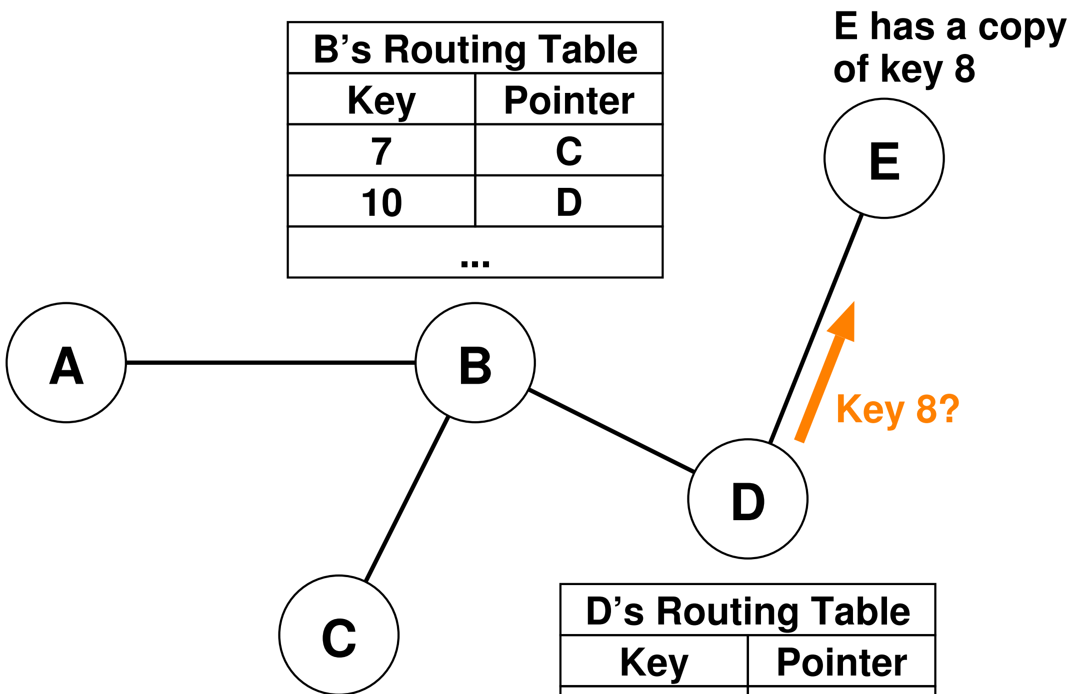
A Search in Freenet Network



B's Routing Table	
Key	Pointer
7	C
10	D
...	

D's Routing Table	
Key	Pointer
8	E
...	

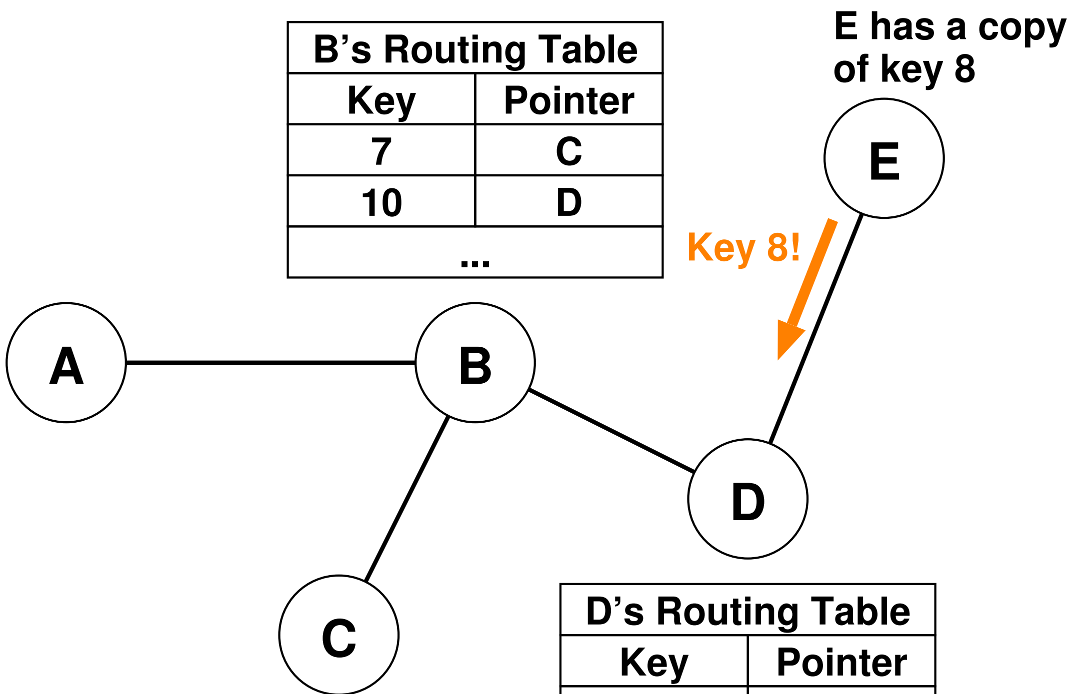
A Search in Freenet Network



B's Routing Table	
Key	Pointer
7	C
10	D
...	

D's Routing Table	
Key	Pointer
8	E
...	

A Search in Freenet Network

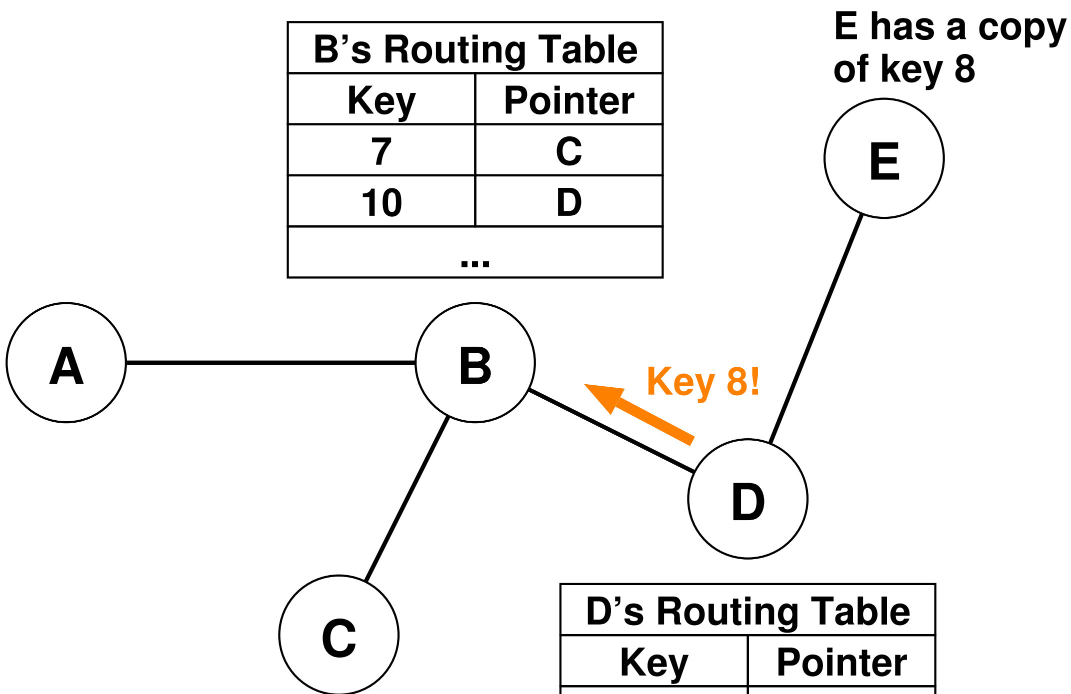


B's Routing Table	
Key	Pointer
7	C
10	D
...	

D's Routing Table	
Key	Pointer
8	E
...	

A Search in Freenet Network

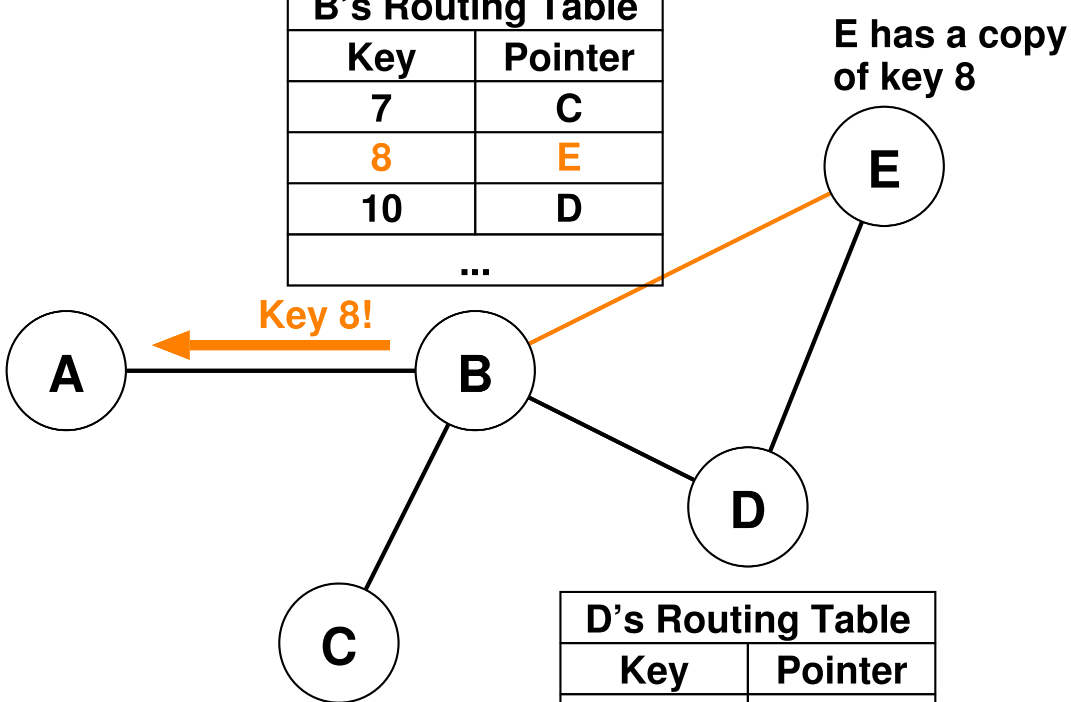
B's Routing Table	
Key	Pointer
7	C
10	D
...	



D's Routing Table	
Key	Pointer
8	E
...	

A Search in Freenet Network

B's Routing Table	
Key	Pointer
7	C
8	E
10	D
...	



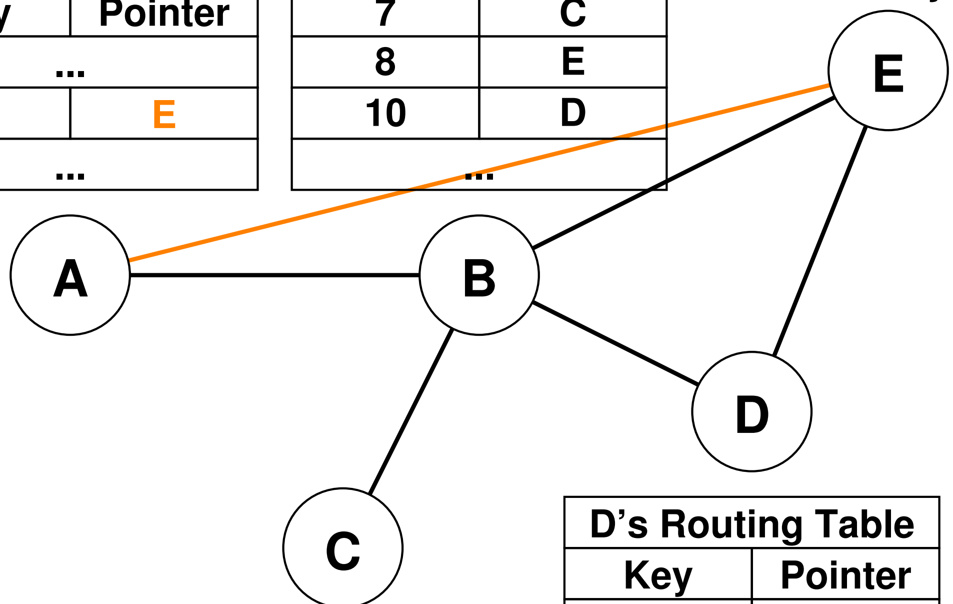
D's Routing Table	
Key	Pointer
8	E
...	

A Search in Freenet Network

A's Routing Table	
Key	Pointer
...	
8	E
...	

B's Routing Table	
Key	Pointer
7	C
8	E
10	D
...	

E has a copy of key 8



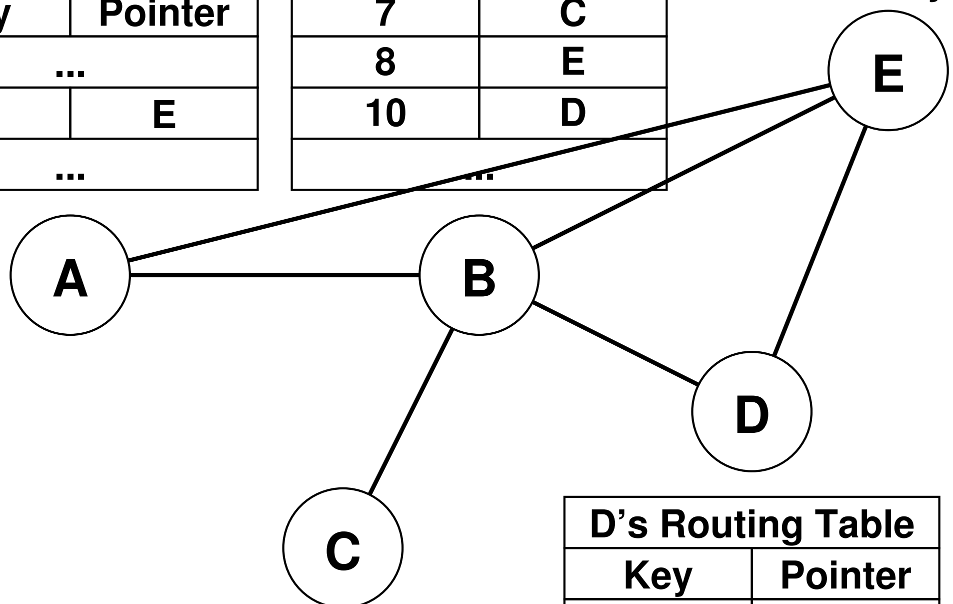
D's Routing Table	
Key	Pointer
8	E
...	

A Search in Freenet Network

A's Routing Table	
Key	Pointer
...	
8	E
...	

B's Routing Table	
Key	Pointer
7	C
8	E
10	D
...	

E has a copy of key 8



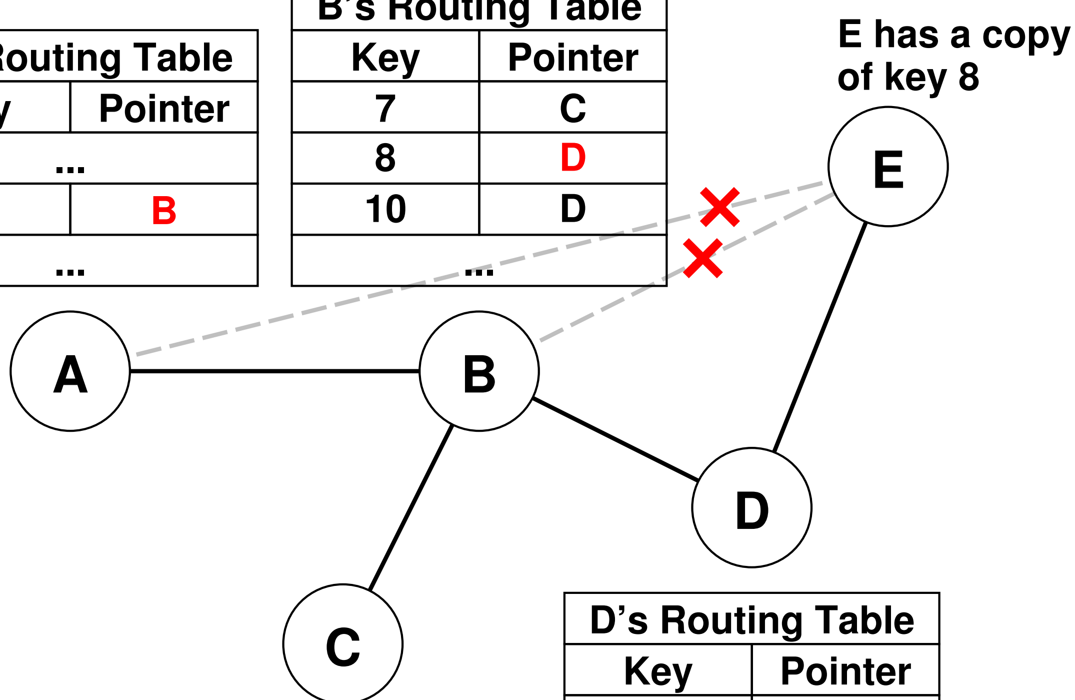
D's Routing Table	
Key	Pointer
8	E
...	

A Search in Freenet Network

A's Routing Table	
Key	Pointer
...	
8	B
...	

B's Routing Table	
Key	Pointer
7	C
8	D
10	D
...	

D's Routing Table	
Key	Pointer
8	E
...	



- Anonymity option can be turned on

Anonymity

- Publisher anonymity
  - data insertion tracks source, but any node can claim itself to be the source (and any node can claim it's just passing data along)
- Consumer anonymity
  - forwarder deniability
  - file contents are encrypted
- Differenent from other peer-to-peer system in this respect

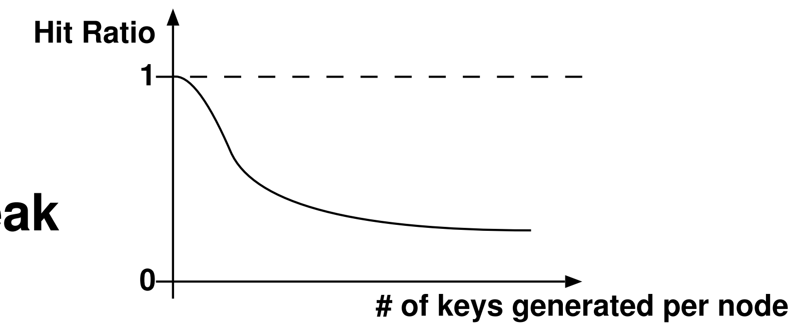
Discussions

- Context
  - there were other anonymous publishing systems
    - e.g., Onion Routing, Publius
- Impact
  - a real system
- Properties
  - novel routing scheme
  - not vulnerable to DoS attacks
  - cannot index the system
  - cannot easily manage the system

Freenet Performance

- The routing in Freenet is expected to run efficiently
  - nodes should come to specialize in locating sets of similar keys
  - nodes should become similarly specialized in storing clusters of files having similar keys.

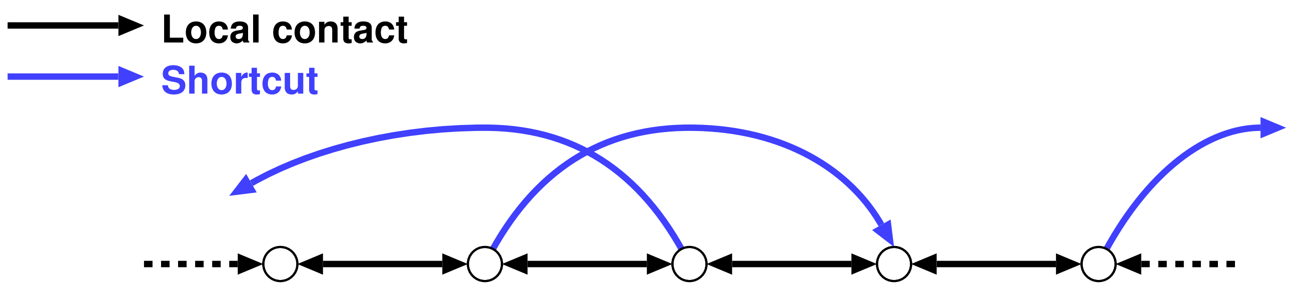
- But in reality, there is a ***step reduction*** in the ***hit ratio*** with increasing load
  - LRU replacement policy
  - Hypothesis: Freenet local caching actions could break up clusters caused by the Freenet routing mechanism
    - *note:* a node has a tendency of ***specializing*** in a key range



Small-world Model [Watts et al 1998]

- *"Six degrees of separation"*
- **A network between order and randomness**
  - **short-distance clustering (like regular graph)**
  - **long-distance shortcuts (result in short global path length like random graph)**

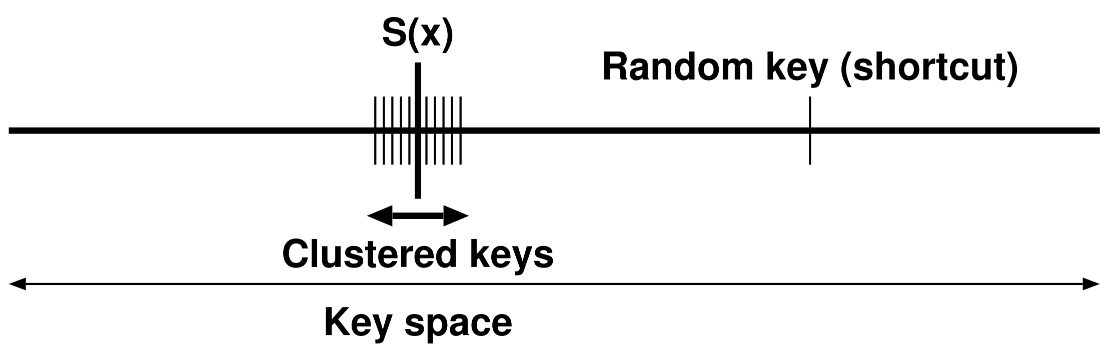
Kleinberg's Theorem [Kleinberg 1999]



- The network model consists of a one-dimensional linear network plus one long-distance shortcut for each node
- The expected steps to delivery a message in the network model is $O(\log^2 n)$ when the shortcut for each node is chosen with the probability inversely proportional to the distance

Clustering with Randomness [Zhang02a]

- To improve routing performance, we want the routing table at node x to conform to the small- world model



- Crucial Observation: such clustering can be achieved by just changing the route-cache replacement policy

Enhanced-clustering Cache Replacement Scheme [Zhang02a]

- Each node chooses a seed randomly when joining the network
- When a new key (file) u is to be cached, the node chooses in the current datastore the key v farthest from the seed
  - If Distance (u, seed) < Distance (v, seed), cache u and evict v (clustering)
  - If Distance (u, seed) > Distance (v, seed), cache u and evict v with probability P (randomness)
    - Can make P dependent on the two distances
- Result:
  - Enhanced-clustering with random shortcuts achieves both the high hit ratio and the low average hops per successful request

Performance Comparison of Several P2P Systems

System	*Expected hops per Request*	*Expected Routing Table Size*
CAN [Ratnasamy01a]	$O(dn^{1/d})$	$O(d)$
CHORD [Stoica01a]	$O(\log n)$	$O(\log n)$
Tapestry [Zhao et al 2000]	$O(\log n)$	$O(\log n)$
Kleinberg's unique SW model	$O(\log^2 n)$	$O(1)$
Idealized Freenet	$O(\log n)$	$O(\log^2 n)$