Peer-to-Peer Networks

Mathew Lowery

Outline

- Introduction
  - What is a P2P network?
- Examples
  - Napster, Morpheus, etc.
- Searching in P2P network
  - Flooding
  - Indices
- Bid trading
  - Digital archives created by swapping space

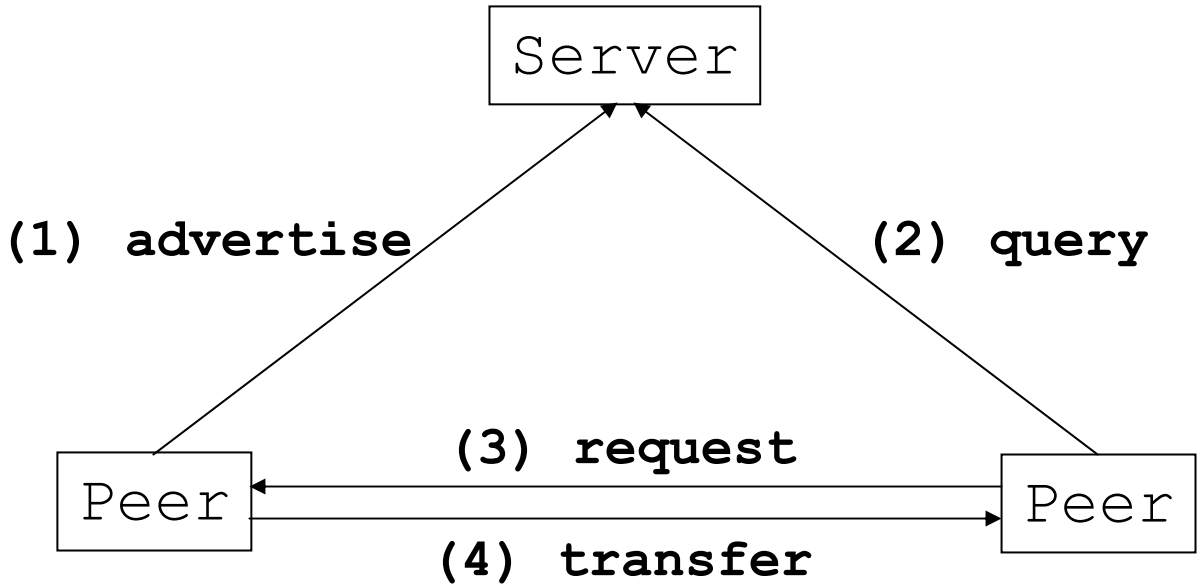
What is a P2P Network?

- Distributed system where all nodes (called peers) have equal roles and capabilities
- Information and service exchange occurs directly between peers
- Strengths include:
  - Self-organization
  - Load-balancing
  - Adaptation
  - Fault tolerance

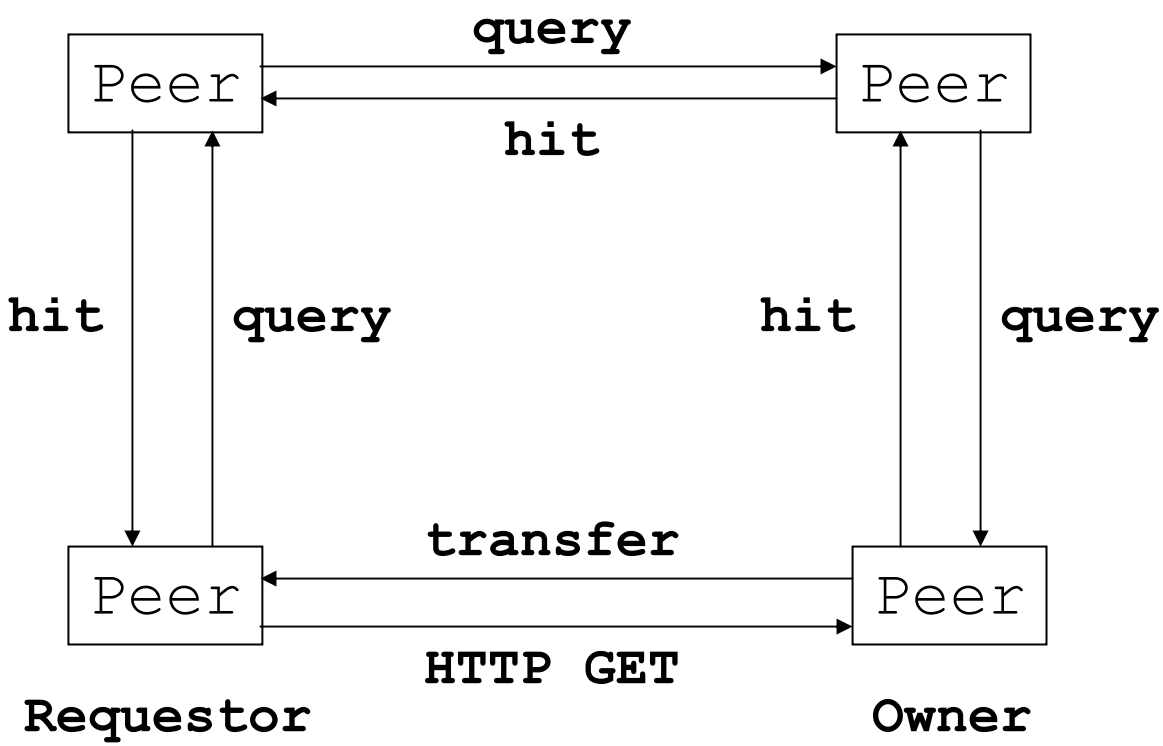
Examples of P2P Networks

- File sharing applications
  - Napster – centralized content search; direct connection for transfer
  - Gnutella – open source protocol; decentralized
  - Morpheus/KaZaA – centralized authentication but otherwise decentralized; proprietary protocol
- File storage applications
  - Freenet – open source; files replicated automatically
- CPU cycle sharing
  - SETI@home – analyzes data from telescopes on personal computers
- Instant messaging

Napster Diagram



Gnutella Diagram



2 Types of P2P Networks

- Persistence and availability guaranteed
  - Guarantee location of content (if it exists) within a bounded number of hops
  - Tightly control data placement and network topology
  - Only support search by identifier
  - Examples: Pastry, Tapestry, CAN, and Chord
- Persistence and availability not guaranteed
  - Cannot guarantee content location even if it exists
  - Intended for a wide range of users (enforcing data placement and network topology difficult)
  - Support richer queries
  - Examples: Gnutella, Napster, and Morpheus

Improving Search in P2P Networks

- 3 techniques
  - Iterative Deepening
  - Directed BFS
  - Local Indices
- Compared against search technique of Gnutella P2P network
- 3 new techniques decrease required aggregate bandwidth and processing with varying tradeoffs including time to satisfy, probability of satisfaction, and number of results

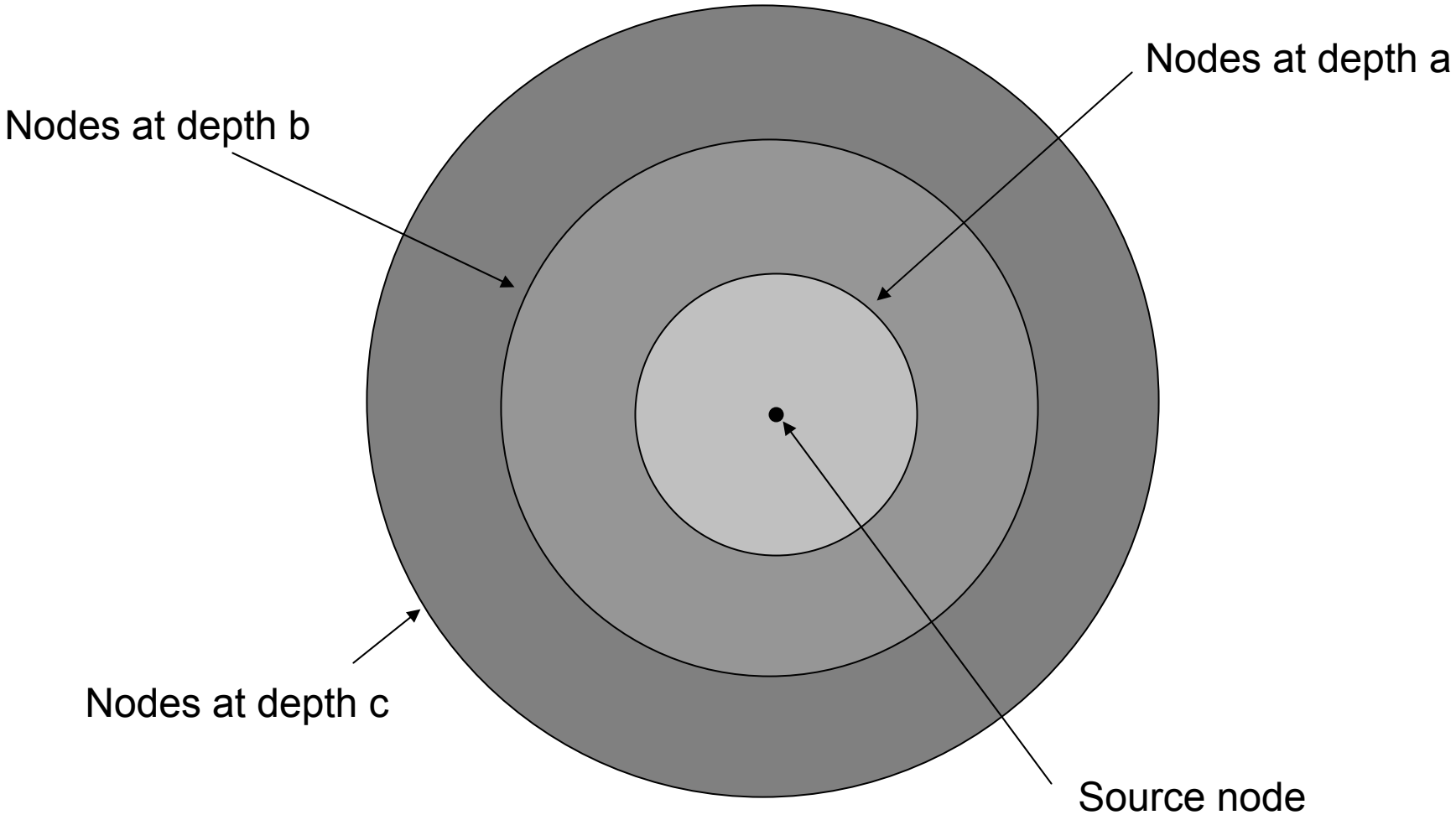
Iterative Deepening

- Parameters: policy P, time W
  - P: specifies depths for each iteration (e.g. P = {a, b, c})
  - W: time between iterations
- Algorithm for source node S:
  - For each iteration i
    - Perform BFS of depth P[i] (TTL = P[i])
    - Wait for time W
    - If query satisfied then done
    - Else send Resend message with TTL = P[i]

Iterative Deepening (continued)

- Algorithm for nodes at depth a:
  - If Query message then freeze query for at most W time units
  - If Resend message then unfreeze query by forwarding Query message with TTL = b - a
- Algorithm for all other nodes:
  - If Resend message then forward
  - If Query message then return results to source and forward query

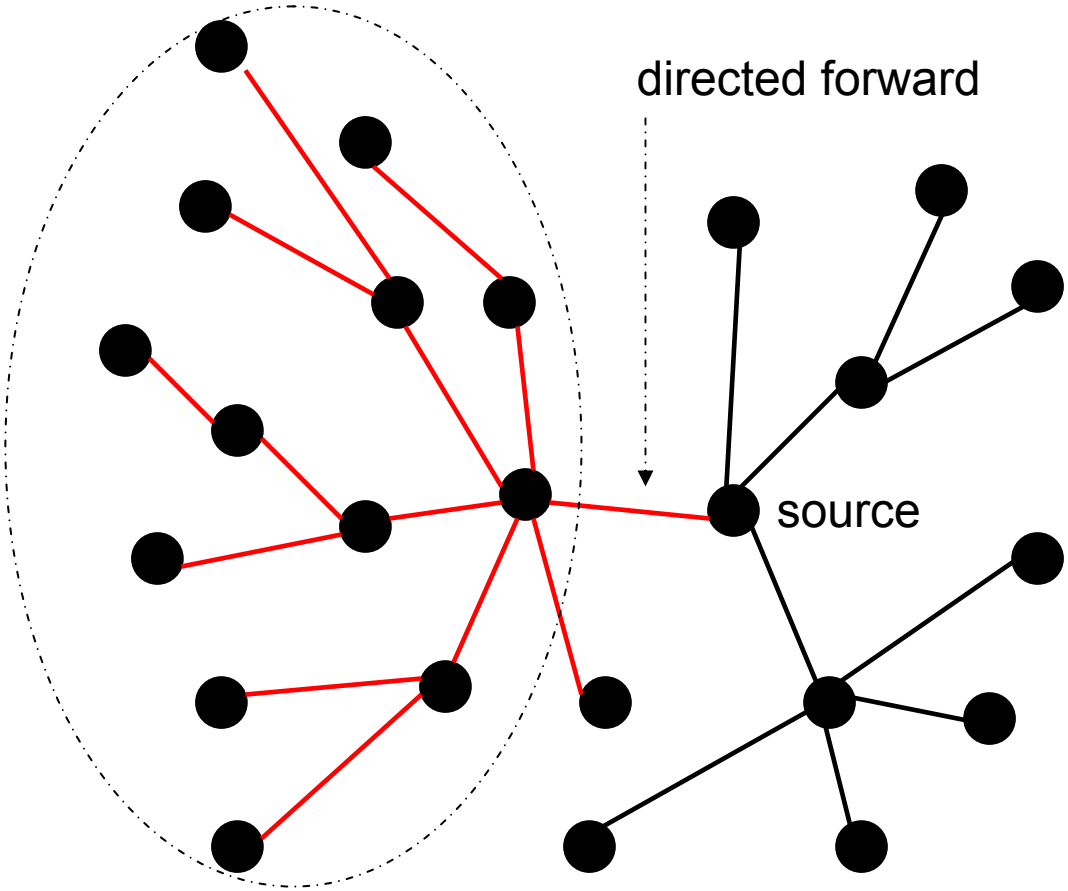
Iterative Deepening (continued)



Directed BFS

- Source selects one neighbor to forward query
- All other nodes continue to forward as in BFS
- Source selects neighbor by collecting statistics from past queries
- Examples:
  - Select neighbor that has returned the highest number of results
  - Select neighbor that has returned messages that have taken the lowest average number of hops
  - Select neighbor that has forwarded the most messages
  - Select neighbor with the shortest message queue

Directed BFS (continued)



breadth first search

red indicates link carrying query

Local Indices

- Each node maintains index over the data of all nodes within r hops of itself
- System-wide policy is used (e.g. P = {1, 5})
- Source node sends Query message to all neighbors
- Nodes not at depth specified in policy simply forward
- Nodes at depth specified in policy process query and return results to source

Local Indices (continued)

- Index maintenance
  - Node join
    - Node X joins by sending Join message with metadata of its collection with TTL = r
    - Any node receiving Join message replies directly to X with metadata of its own collection
  - Node departure or failure
    - Timeout removes old data
  - Node update
    - Small Update message with TTL = r

Search Mechanisms in P2P

- 3 types:
  - No index
    - Simple and robust
    - But enormous cost of flooding
  - Centralized search
    - One hop to answer
    - But vulnerable to attack
  - Indices at each node
    - Answer queries with fewer messages
    - But overhead of maintaining index

Routing Indices for P2P Systems

- 3 techniques
  - Compound
  - Hop-count
  - Exponential
- Compared against flooding and random forwarding

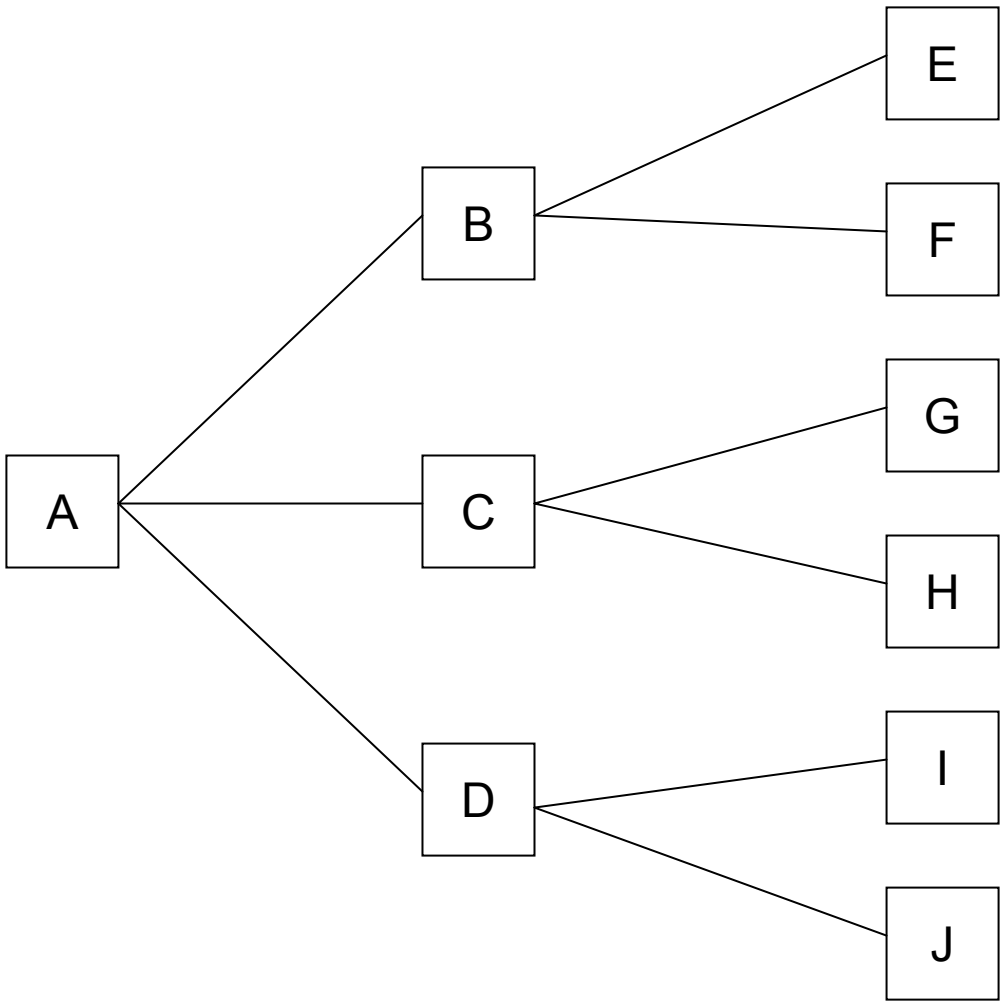
Routing Indices

- Routing indices are small tables that give a “direction” towards a document rather than its actual location
- Tables proportional to number of neighbors, not number of documents
- Assume that queries are on content of documents, not document identifiers
- Assume users submit queries to node with a stop condition (e.g. the desired number of results). If stop condition not reached after doing local search, forward to one or more neighbors with updated stop condition

Routing Indices (continued)

- A routing index (RI) is a data structure that, when given a query, returns a list of neighbors ranked according to their "goodness"

Compound Routing Indices (CRI)



Compound Routing Indices (CRI)

Routing Index for node A

Path	docs	Documents with topics: DB	N	T	L
B	100	20	0	10	30
C	1000	0	300	0	50
D	200	100	0	100	150

Compound Routing Indices (CRI)

- Summarization into topics may introduce overcounts or undercounts in the RI
  - Overcounts: Documents on “indices”, “recovery”, and “SQL” are included in “databases” topic. Then a query on “SQL” is converted to one on “databases” which implies there are many documents on “SQL” when in fact there may be few or none
  - Undercounts: If we use a frequency threshold to throw away topics with few documents, then we’ll miss documents that actually exist

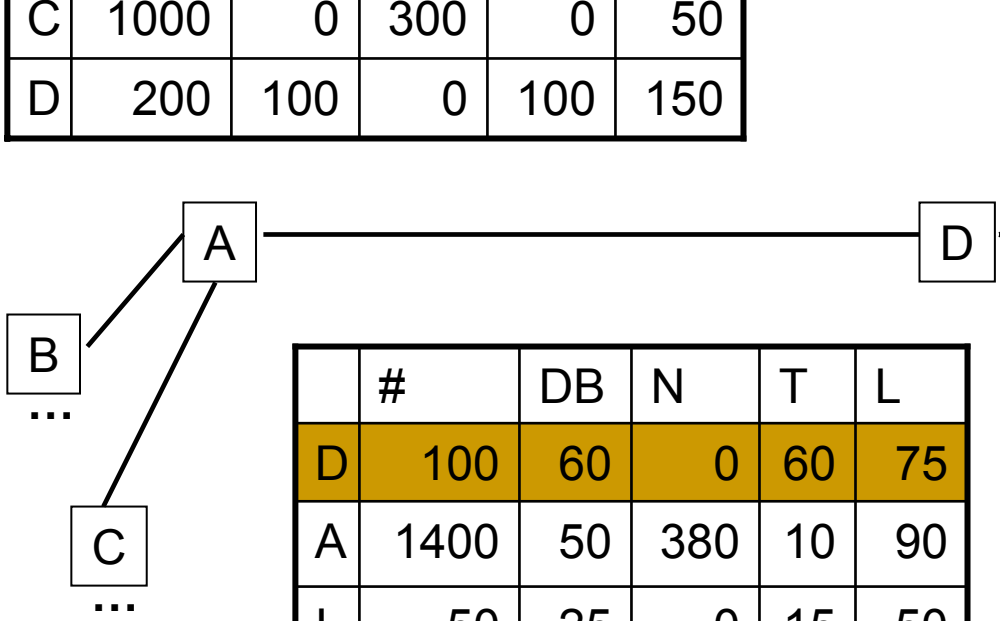
Compound Routing Indices (CRI)

- Assume simplified model where queries are conjunctions of topics, documents can have more than one topic, and document topics are independent
- Number of results in a path = NumberOfDocs x Π_i CRI(s_i)/NumberOfDocs
- Savings in number of messages over flooding and random forwarding comes from fact that query only forwarded to nodes that have high potential for having results
- RIs require more total space than centralized index but the cost is shared among the network nodes

CRI Example

	#	DB	N	T	L
A	300	30	80	0	10
B	100	20	0	10	30
C	1000	0	300	0	50
D	200	100	0	100	150

	#	DB	N	T	L
I	50	25	0	15	50
D	1550	125	380	95	190



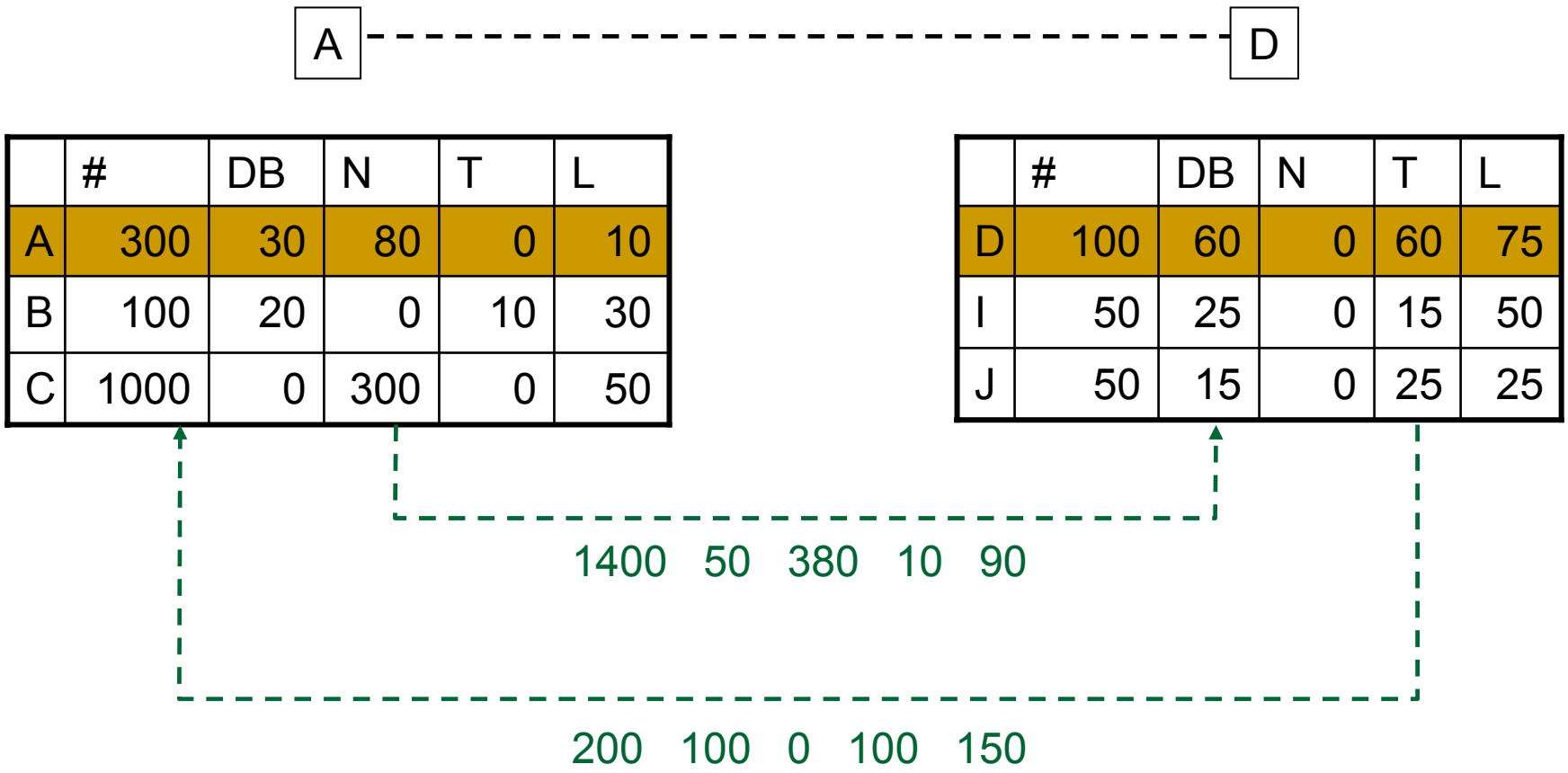
	#	DB	N	T	L
D	100	60	0	60	75
A	1400	50	380	10	90
I	50	25	0	15	50
J	50	15	0	25	25

	#	DB	N	T	L
J	50	15	0	25	25
D	1550	135	380	85	215

Table Maintenance

- Create and Update
  - When a connection is made between nodes, they swap aggregated RI information. This information must propagate to all nodes in order to keep up-to-date.
- Node departure
  - Update local RI and send new aggregate information to all neighbors. This again must occur between every pair of nodes.

Creating CRIs Example (1)



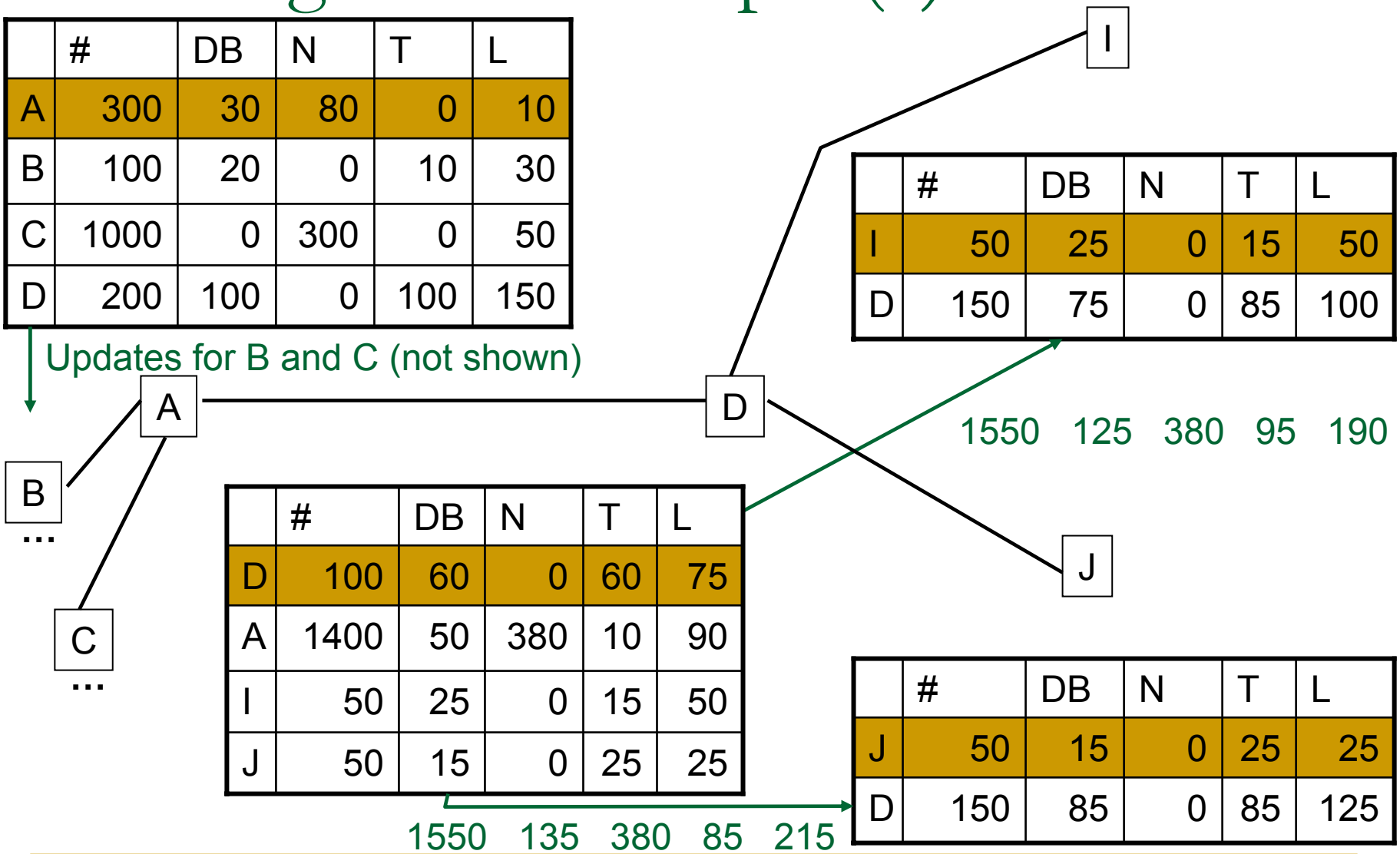
	#	DB	N	T	L
A	300	30	80	0	10
B	100	20	0	10	30
C	1000	0	300	0	50

	#	DB	N	T	L
D	100	60	0	60	75
I	50	25	0	15	50
J	50	15	0	25	25

Creating CRIs Example (2)

	#	DB	N	T	L
A	300	30	80	0	10
B	100	20	0	10	30
C	1000	0	300	0	50
D	200	100	0	100	150

Updates for B and C (not shown)



	#	DB	N	T	L
I	50	25	0	15	50
D	150	75	0	85	100

	#	DB	N	T	L
D	100	60	0	60	75
A	1400	50	380	10	90
I	50	25	0	15	50
J	50	15	0	25	25

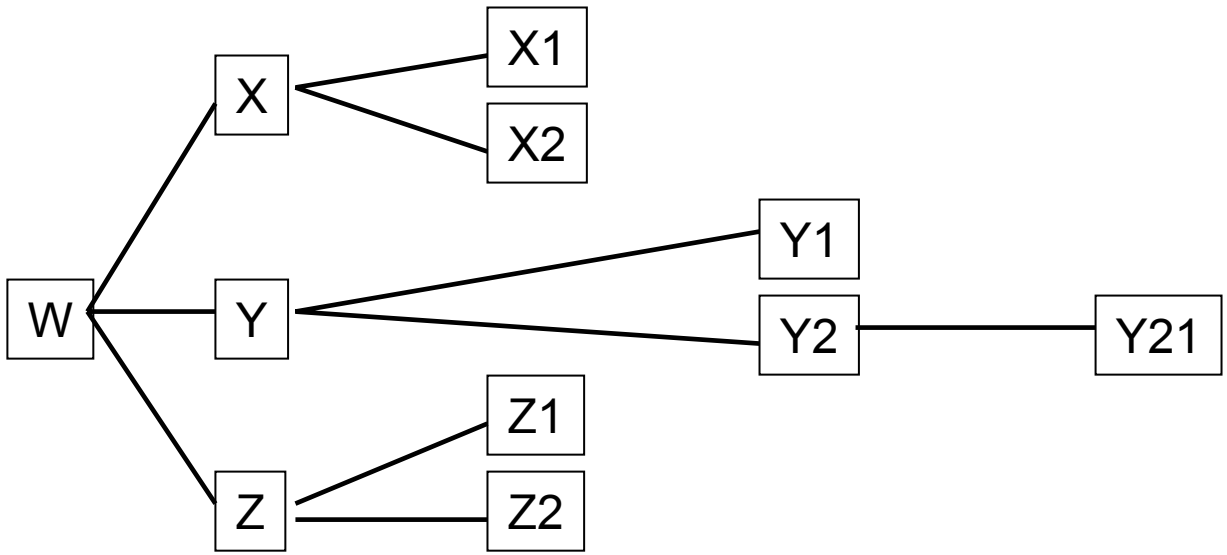
	#	DB	N	T	L
J	50	15	0	25	25
D	150	85	0	85	125

Hop-count RI (HRI)

- Takes into account the number of hops required to find documents (reasoning: a path which has more documents but requires many more hops is not as desirable as one that has fewer documents but much fewer hops)
- In table, aggregated RIs for each hop are stored up to some maximum called the horizon

Hop-count Example

	1 Hop					2 Hops				
	#	DB	N	T	L	#	DB	N	T	L
X	60	13	2	5	10	20	10	10	4	17
Y	30	0	3	15	12	50	31	0	15	20
Z	5	2	0	3	3	70	10	40	20	50



Exponentially Aggregated RI (ERI)

- Hop-count RI comes at the cost of higher storage and transmission cost; also its performance is negatively affected by the lack of information beyond the horizon
- ERI overcomes these disadvantages at the cost of some potential loss in accuracy
- ERI outperforms HRI in most cases

Bid Trading

- Digital archives protect information through multiple copies
- A local site that wants to replicate information announces its storage requirement and accepts bids from other sites
- The local site “pays” for the remote storage by offering storage space of its own for use by the bid winner
- Space traded need not be equal
- This technique has higher reliability than when sites trade equal space without bidding

Bid Trading

- This system preserves site autonomy and decision making while providing more storage than the site can possibly afford to buy and maintain
- “collective benefit from individual action”
- An accepted bid leads to the swapping of deeds where a deed is the right of one site to use space on another

Bid Trading

- Auction calling policy: guides decision of when to call an auction (e.g. when it needs to replicate collections)
- Space management policy: guides decision of how much local storage to make available for public use (e.g. to reserve some space for future use)
- Bidding policy: guides construction of an appropriate bid (e.g. how urgently a site needs to replicate its own collections)

Bid Policy Examples

- FreeSpace
  - Site bids more when it has more free space.
  - Site tends to win more bids when its space is scarce since it bids low.
  - Scarce space makes trading difficult so bid low to win as many bids as possible.
- UsedSpace
  - Site bids more when more of its space is used.
  - Site bids low and wins when space is abundant.
  - Allows sites to hoard local space when scarce.

References

- The Gnutella Protocol Specification v0.4. http://www9.limewire.com/developer/gnutella_protocol_0.4.pdf
- I. Clarke, S. G. Miller, T. W. Hong, O. Sandberg, and B. Wiley. Protecting Free Expression Online with Freenet. IEEE Internet Computing. Jan-Feb 2002.
- B. Yang and H. Garcia-Molina. Improving Search in Peer-to-Peer Networks. Proceedings of the 22nd International Conference on Distributed Computing Systems.
- A. Crespo and H. Garcia-Molina. Routing Indices For Peer-to-Peer Systems. Proceedings of the 22nd International Conference on Distributed Computing Systems.
- B. F. Cooper and H. Garcia-Molina. Bidding for storage space in a peer-to-peer data preservation system. Proceedings of the 22nd International Conference on Distributed Computing Systems.