CS 345
Topics in Peer-to-Peer Systems

Mayank Bawa
bawa@cs.stanford.edu

Notes 09(a): Querying the Internet with PIER

Queries on P2P networks

Search queries
“Retrieve documents with specified keywords”

Aggregate queries
“Compute statistics of data-items”

Relational queries
“Return all data-items that match a predicate”
Range predicates: R.K > 20 & R.K < 40
Equi-join predicates: R.K = S.K

Example: Managing/Identifying Spam

Spam(sourceIP, smtpDomain, msgHash, dateSent)
Crawler(clientIP, clientDomain, dateSeen)
Sharing local spam information
Each organization puts its date at a central host;
Each organization participates in a P2P network

Example: Managing/Identifying Spam

Spam(sourceIP, smtpDomain, msgHash, dateSent)
Find popular spam
SELECT S.msgHash, count(*) AS cnt
FROM Spam as S
GROUP BY S.msgHash
HAVING cnt > 100
Crawler(clientIP, clientDomain, dateSeen)
Find if spammed addresses were collected by a crawler:
SELECT S.sourceIP
FROM Spam as S, Crawler as C
WHERE S.smtpDomain = C.clientDomain

How can we enable relational queries?
PIER [HHLSS03]
DHT (Bottom layer)
Routing layer unchanged
Storage manager at each node (Hashtable)

Join strategy: Hash Join

Join strategy: Fetch Matches

Join strategy: Symmetric Semi-Join

Semantics of query

Query is multicast to all hosting nodes
“Best-effort” results are streamed back

- Local snapshot on query delivery
- Soft (leased) state leads to stale results
- Failure at hosting node leads to lost data
- Failure at computing node leads to lost answers

Evaluate quality using “recall”
Experimental Results

Latency (time to 30th tuple)
- Hash join < Fetch join < Symmetric semi-join

Traffic
- Symmetric semi-join < Fetch join < Hash join

Recall increases with refresh frequency