



SmartSeer: Continuous Queries over CiteSeer

Jayanthkumar Kannan, Beverly Yang, Scott Shenker,
Sujata Banerjee, Sung Ju Lee, Puneet Sharma, Sujoy Basu



Motivation

- Problem:
 - Increasing number of research papers
 - No easy way for researchers to keep themselves informed of new papers
 - Increasingly a problem as online preprint libraries etc become more popular
- Our approach:
 - allow users to register continuous queries
 - notify users of incremental results



Requirements

- Donated Infrastructure
 - Single server clearly inadequate
 - Google does it, but how much does it cost them?
 - The success of Planetlab is encouraging
- Rich Continuous Queries
 - Eg: Nested Queries, Joins
 - Citeseer documents have rich structured data: citations, authors etc
 - Allow queries to exploit this structure (follow citations etc)



Sample Queries

- Text matching:
 - `TEXT="database" & TEXT = "networks"`
- Who is citing me?
 - `CITES=(AUTHOR=xxx)`
- Curious Joe
 - `AUTHOR=(AUTHOR=Joe)`



Related Work

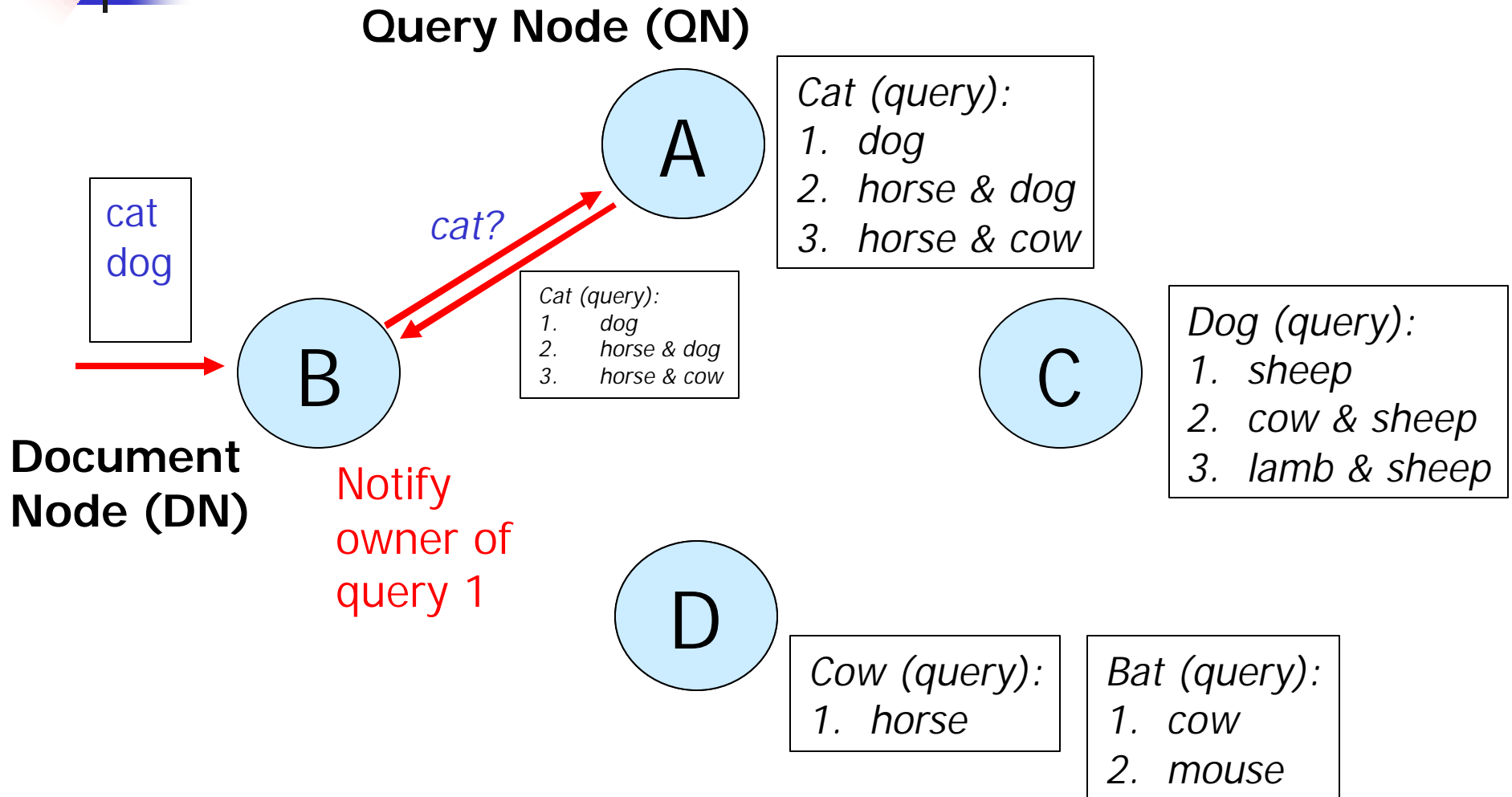
- Telegraph (and others)
 - Allows fully general SQL queries
 - Works on tightly coupled distributed system
- Event notification systems (Scribe etc)
 - Only simple event semantics
- PIER
 - Extends mechanism for instantaneous queries to continuous queries
 - No 'batched' processing of queries



DHT for Continuous Queries

- DHT chosen over replication, partition-by-id approaches
- System model:
 - users register queries
 - matching queries notified on document insertion
- AND query registered at one of its keywords
 - stored at node responsible for hash of keyword
- Document Insertion
 - fetch query list of each word in the document

Fetch Query strategy





Alternate strategies

- Send Document
 - Store entire query at its keyword
 - DN sends entire document to QN
- Term Dialogue
 - QN chooses a term from its set of queries and asks DN
 - DN replies with "yes/no"
 - Recurse
- Bloom Filter
- None of these strategies work for immediate queries



Complex Queries

- Nested Queries
 - Eg: Doc where Author=(Author=xxx)
 - Subquery selects papers written by xxx
- New answer to query if
 - New paper written by some co-author of xxx
 - Or xxx writes a paper with a new co-author
- Register two sub-queries
- Can be extended for more complex queries
 - Fails for all negative predicates



Status

- Implementation in Java
- Will be soon deployed on Planetlab as a public service
- Future work:
 - Relevance Feedback
 - Using semantic vector to reduce bandwidth