StreamGlobe: P2P Stream Sharing

Matching and Evaluation of Disjunctive Predicates for Data Stream Sharing

Richard Kuntschke and Alfons Kemper
Lehrstuhl Informatik III: Datenbanksysteme
Technische Universität München – Fakultät für Informatik

Network Basics:

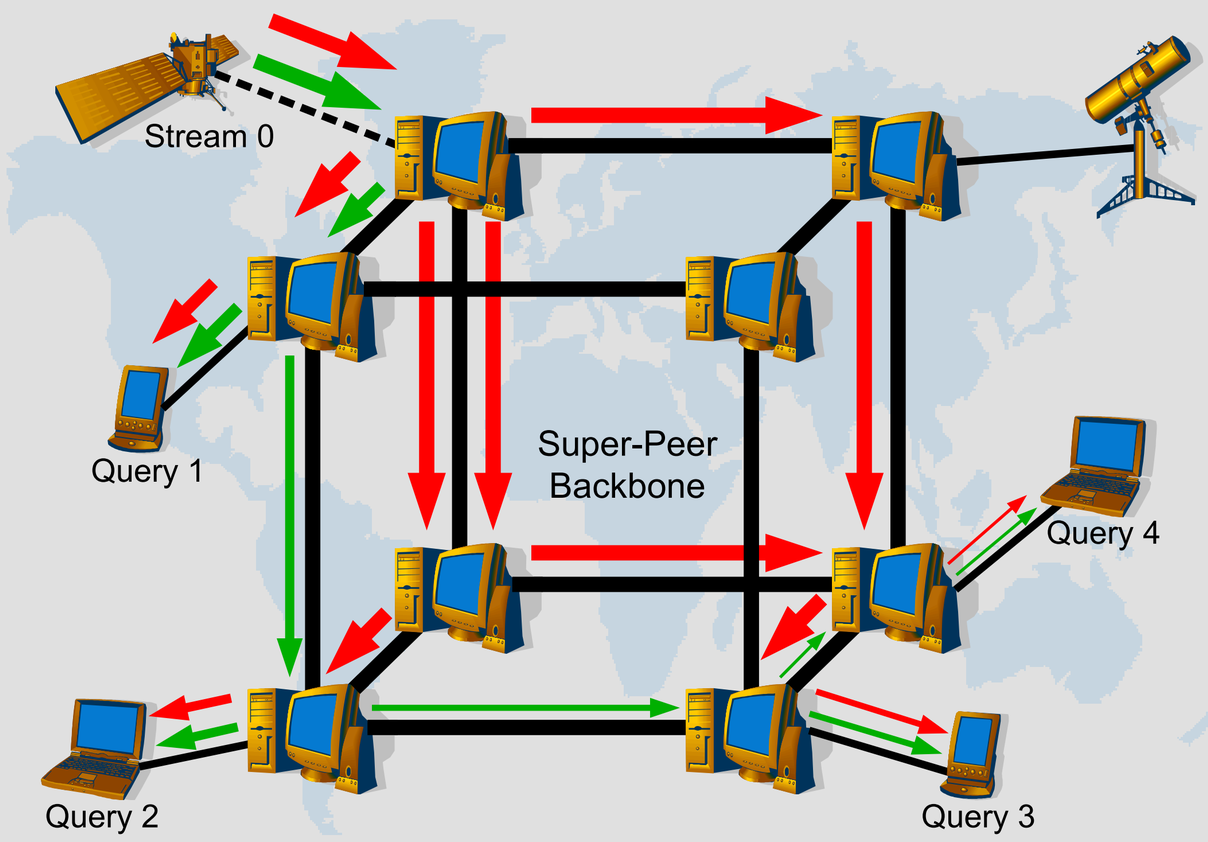
- Grid-based P2P network
- Super-Peer Backbone
- Super-Peers: Powerful stationary servers
- Thin-Peers: Less powerful, possibly mobile peers, sensor devices, etc.

Deficiencies of traditional query evaluation:

- Redundant transmission of data streams
- Redundant execution of stream transforming operators
- Transmission of unnecessary data

⇨ Increased network traffic
⇨ Increased peer load

Publish & Subscribe in a P2P Network



StreamGlobe Basics:

- StreamGlobe: Distributed Data Stream Management System (DSMS)
- Super-Peers process and route data streams
- Thin-Peers publish and subscribe to data streams

Benefits of Data Stream Sharing in StreamGlobe:

- Stream sharing avoids redundant stream transmission
- Sharing computational results avoids redundant computation
- Early filtering and aggregation avoid unnecessary data transmission

⇨ Reduced network traffic
⇨ Reduced peer load

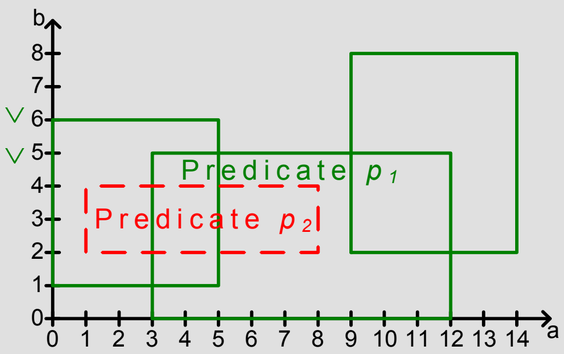
Predicate Matching (1)

Problem:

- Data stream sharing requires identifying shareable data streams
- Identification process involves matching (selection/join) predicates:
  Given a predicate p_1 and a (query) predicate p_2, does p_2 imply p_1?
  If not, how can we alter p_1 for the implication to become valid?

Example Predicates:

Stream Predicate *p1*:
$(a \geq 3) \wedge (a \leq 12) \wedge (b \geq 0) \wedge (b \leq 5) \vee$
$(a \geq 9) \wedge (a \leq 14) \wedge (b \geq 2) \wedge (b \leq 8) \vee$
$(a \geq 0) \wedge (a \leq 5) \wedge (b \geq 1) \wedge (b \leq 6)$
Query Predicate *p2*:
$(a \geq 1) \wedge (a \leq 8) \wedge (b \geq 2) \wedge (b \leq 4)$



Quick Check (QC):

- Checks conjunctive subpredicates for containment
- Can be combined with any of the other matching algorithms

Heuristics with Simple Relaxation (HSR):

- Disjunctively adds conjunctive subpredicates of p_2 to p_1
- ☺ Fast and easy to implement
- ☹ Misses matches in general causing unnecessary predicate relaxations
- ☹ Generally increases number of disjunctions in p_1

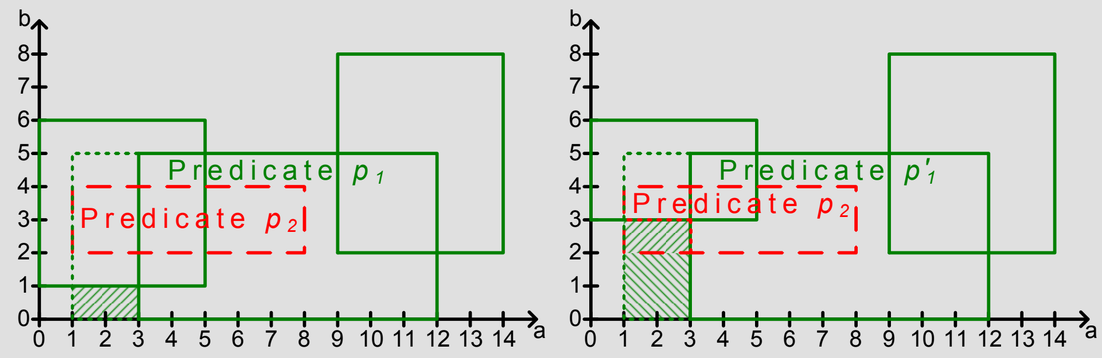
Predicate Matching (2)

Heuristics with Complex Relaxation (HCR):

- Relaxes subpredicates of p_1 to contain subpredicates of p_2
- ☺ Relatively fast and easy to implement
- ☺ Does not add any disjunctions to p_1
- ☹ Misses matches in general
- ☹ Might add unnecessary parts of the data space to p_1
  (leads to approximate results during predicate evaluation)

Exact Matching (EM):

- Split algorithm
- ☺ Exactly identifies matches, mismatches, and non-matching parts
- ☹ Exponential complexity in number of subpredicates
  ⇒ inapplicable for larger problem sizes
  ⇒ use heuristics instead



Predicate Evaluation

Problem:

- Given a predicate *p* and a data item *i*, does *i* satisfy *p*?
- Efficiently evaluate disjunctive predicates with potentially many disjunctions

Standard Evaluation (SE):

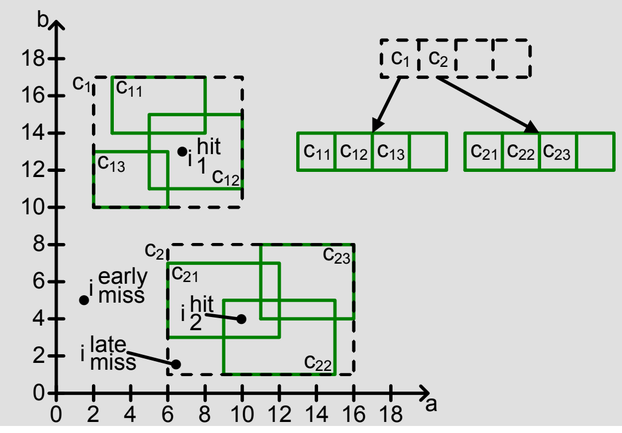
- Sequential scan
- Early exit when a match occurs

Index-based Evaluation (IE):

- Multi-dimensional index
- Early exit when a mismatch occurs

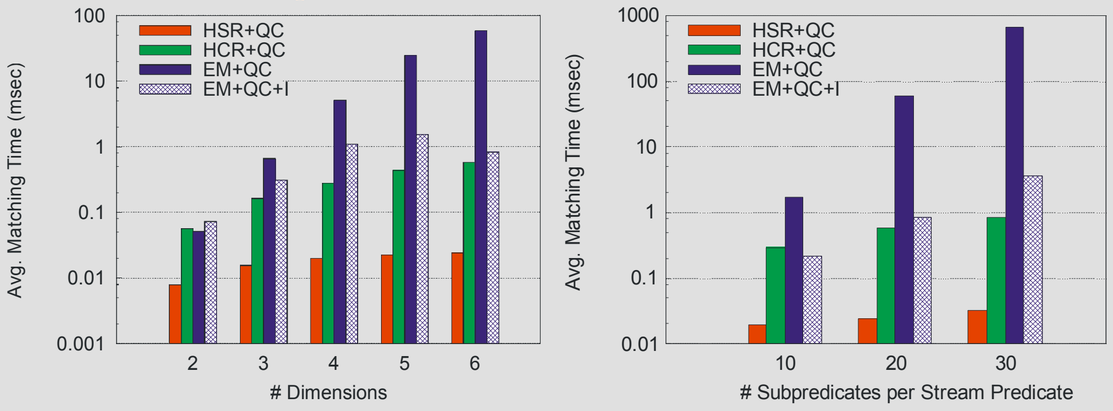
Optimization:

- Multi-dimensional index support (I) for predicate matching and evaluation
- Improve performance of evaluation index through short-circuiting (SC)

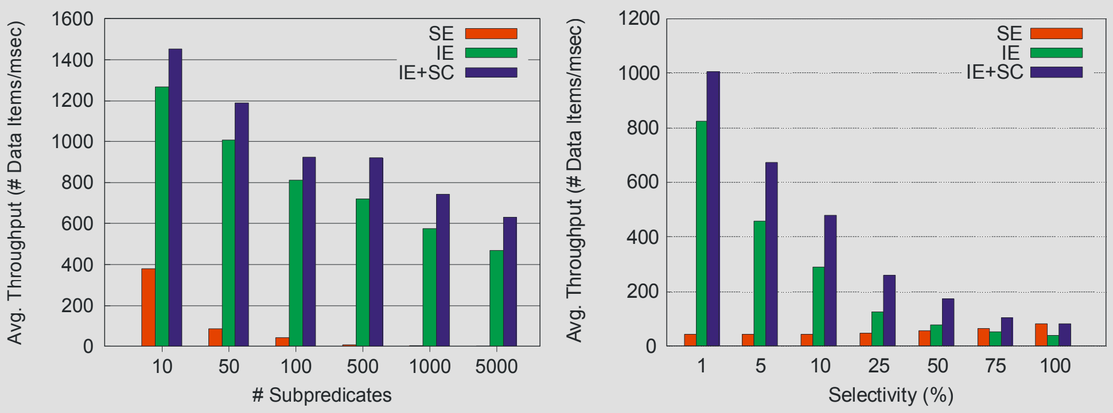


Benchmark Results

Predicate Matching:



Predicate Evaluation:



http://www-db.in.tum.de/research/projects/StreamGlobe