

Query Processing over Data Streams

Joint project with Prof. Rajeev Motwani
and a group of graduate students

stanfordstreamdatamanager

Formula for a Database Research Project

- Pick a simple but fundamental assumption underlying traditional database systems
 - Drop it
- Reconsider all aspects of data management and query processing
 - Many Ph.D. theses
 - Prototype from scratch

stanfordstreamdatamanager

2

Following the Formula

- We followed this formula once before
 - The LORE project
 - Dropped assumption:
Data has a fixed schema declared in advance
 - XML
- The STREAM Project
 - Dropped assumption:
First load data, then index it, then run queries
 - Continuous data streams (+ continuous queries)

stanfordstreamdatamanager

3

Data Streams

- Continuous, unbounded, rapid, time-varying streams of data elements
- Occur in a variety of modern applications
 - Network monitoring and traffic engineering
 - Sensor networks
 - Telecom call records
 - Financial applications
 - Web logs and click-streams
 - Manufacturing processes
- DSMS = Data Stream Management System

stanfordstreamdatamanager

4

The STREAM System

- Data streams and stored relations
- Declarative language for registering continuous queries
- Designed to cope with high data rates and query workloads
 - Graceful approximation when needed
 - Careful resource allocation and usage

stanfordstreamdatamanager

5

The STREAM System

- Designed to cope with bursty streams and changing workloads
 - Continuous monitoring of data and system behavior
 - Continuous reoptimization
- Relational, centralized (for now)

stanfordstreamdatamanager

6

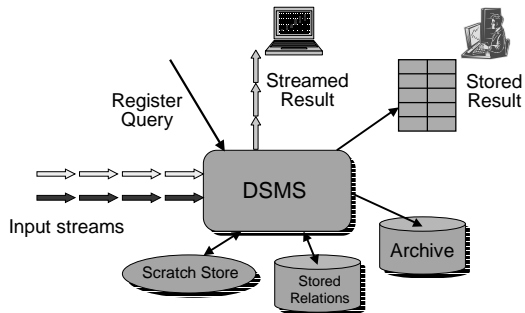
Contributions to Date

- Language for continuous queries
- Query plans
- Query optimization and dynamic reoptimization
- Exploiting stream constraints
- Operator scheduling
- Approximation techniques
- Resource allocation to maximize precision
- Initial running prototype

stanfordstreamdatamanager

7

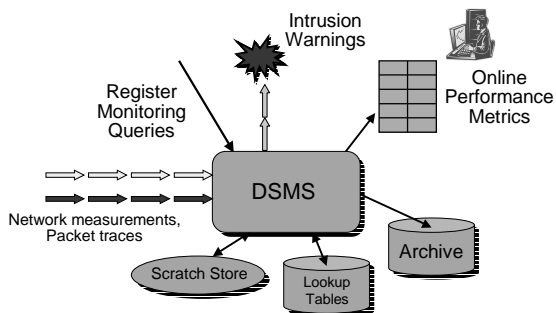
The (Simplified) Big Picture



stanfordstreamdatamanager

8

(Simplified) Network Monitoring



stanfordstreamdatamanager

9

Language for Continuous Queries

GOALS

- Continuous queries over multiple streams and relations
- Exploit relational semantics to the extent possible
- Easy queries should be easy to write
- Queries should do what you expect

stanfordstreamdatamanager

10

Example Query 1

Two streams, contrived for ease of examples:

Orders (orderID, customer, cost)

Fulfillments (orderID, clerk)

Total cost of orders fulfilled over the last day by clerk "Sue" for customer "Joe"

```
Select Sum(O.cost)
From Orders O, Fulfillments F [Range 1 Day]
Where O.orderID = F.orderID And F.clerk = "Sue"
And O.customer = "Joe"
```

stanfordstreamdatamanager

11

Example Query 2

Using a 10% sample of the Fulfillments stream, take the 5 most recent fulfillments for each clerk and return the maximum cost

```
Select F.clerk, Max(O.cost)
From Orders O,
      Fulfillments F [Partition By clerk Rows 5] 10% Sample
Where O.orderID = F.orderID
Group By F.clerk
```

stanfordstreamdatamanager

12

Abstract Semantics and Concrete Language

- Abstract: window specification language and relational query language as “black boxes”
- Concrete: SQL-based instantiation for our system; includes syntactic shortcuts, defaults, equivalences

stanfordstreamdatamanager

13

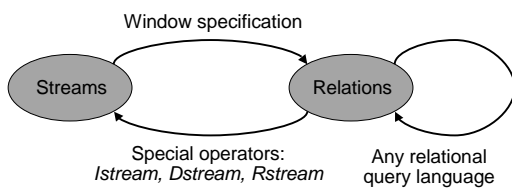
Relations and Streams

- Assume global, discrete, ordered time domain
- Relation
 - Maps time T to set-of-tuples R
- Stream
 - Set of $(tuple, timestamp)$ elements

stanfordstreamdatamanager

14

Conversions



stanfordstreamdatamanager

15

Conversion Definitions

- Stream-to-relation
 - $S[W]$ is a relation — at time T it contains all tuples in window W applied to stream S up to T
 - When $W = \infty$, contains all tuples in stream S up to T
- Relation-to-stream
 - $Istream(R)$ contains all (r, T) where $r \in R$ at time T but $r \notin R$ at time $T-1$
 - $Dstream(R)$ contains all (r, T) where $r \in R$ at time $T-1$ but $r \notin R$ at time T
 - $Rstream(R)$ contains all (r, T) where $r \in R$ at time T

stanfordstreamdatamanager 16

Abstract Semantics

- Take any relational query language
- Can reference streams in place of relations
 - But must convert to relations using any window specification language (default window = $[\infty]$)
- Can convert relations to streams
 - For streamed results
 - For windows over relations (note: converts back to relation)

stanfordstreamdatamanager 17

Query Result at Time T

- Use all relations at time T
- Use all streams up to T , converted to relations
- Compute relational result
- Convert result to streams if desired

stanfordstreamdatamanager 18

Abstract Semantics – Example 1

```
Select F.clerk, Max(O.cost)
From O [ $\infty$ ], F [Rows 1000]
Where O.orderID = F.orderID
Group By F.clerk
```

- Maximum-cost order fulfilled by each clerk in last 1000 fulfillments

stanfordstreamdatamanager

19

Abstract Semantics – Example 1

```
Select F.clerk, Max(O.cost)
From O [ $\infty$ ], F [Rows 1000]
Where O.orderID = F.orderID
Group By F.clerk
```

- At time T : entire stream O and last 1000 tuples of F as relations
- Evaluate query, update result relation at T

stanfordstreamdatamanager

20

Abstract Semantics – Example 1

```
Select Istream(F.clerk, Max(O.cost))
From O [ $\infty$ ], F [Rows 1000]
Where O.orderID = F.orderID
Group By F.clerk
```

- At time T : entire stream O and last 1000 tuples of F as relations
- Evaluate query, update result relation at T
- Streamed result: New element $\langle \text{clerk}, \text{max} \rangle, T$ whenever $\langle \text{clerk}, \text{max} \rangle$ changes from $T-1$

stanfordstreamdatamanager

21

Abstract Semantics – Example 2

Relation CurPrice(stock, price)

Select stock, Avg(price)

From Istream(CurPrice) [Range 1 Day]

Group By stock

- Average price over last day for each stock

stanfordstreamdatamanager

22

Abstract Semantics – Example 2

Relation CurPrice(stock, price)

Select stock, Avg(price)

From Istream(CurPrice) [Range 1 Day]

Group By stock

- *Istream* provides history of *CurPrice*
- Window on history, back to relation, group and aggregate

stanfordstreamdatamanager

23

Concrete Language – CQL

- Relational query language: SQL
- Window spec. language derived from SQL-99
 - Tuple-based, time-based, partitioned
- Syntactic shortcuts and defaults
 - *So easy queries are easy to write and simple queries do what you expect*
- Equivalences
 - Basis for query-rewrite optimizations
 - Includes all relational equivalences, plus new stream-based ones

stanfordstreamdatamanager

24

Two Extremely Simple CQL Examples

Select * From Strm

- Had better return *Strm* (It does)
 - Default ∞ window for *Strm*
 - Default *Istream* for result

Select * From Strm, Rel Where Strm.A = Rel.B

- Often want "NOW" window for *Strm*
- But may not want as default

stanfordstreamdatamanager 25

Stream Systems

- (At least) three general-purpose DSMS prototypes underway
 - STREAM (Stanford)
 - Aurora (MIT, Brown, Brandeis)
 - TelegraphCQ (Berkeley)
- All will be demo'd at conference next week
- Cooperating to develop stream system benchmark
 - Goal: demonstrate that conventional systems are far inferior for data stream applications

stanfordstreamdatamanager 26

Contributions to Date

- Language for continuous queries
- Query plans
- Query optimization and dynamic reoptimization
- Exploiting stream constraints
- Operator scheduling
- Approximation techniques
- Resource allocation to maximize precision
- Initial running prototype

stanfordstreamdatamanager 27

<http://www-db.stanford.edu/stream>

stanfordstreamdatamanager 28
