Continuous Queries over Data Streams

Shivnath Babu and Jennifer Widom
Stanford University

Presented by Chung Leung, LAM

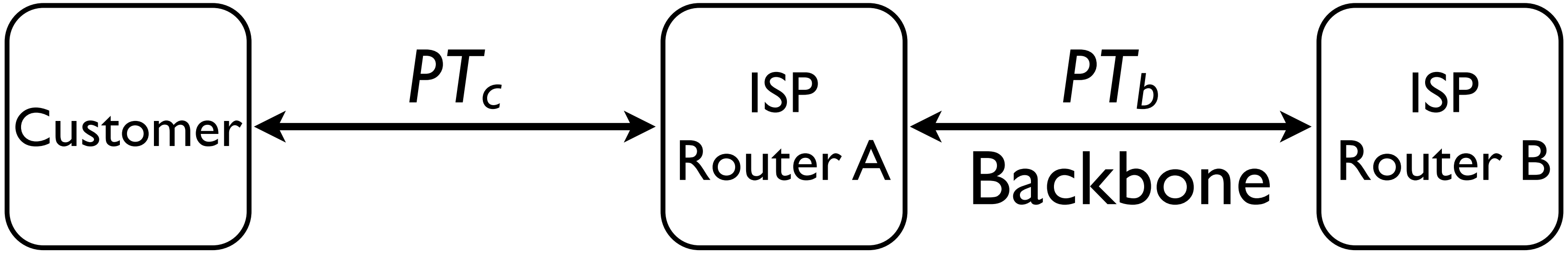
Overview

- Use of continuous data stream
- Survey & New architecture
- Continuous Queries over Data Stream
- The STREAM (STandford stREam datA Management) project

The Survey

- [TGNO92] - Continuous queries
- [JMS95] - Data streams
- [SPAM91] - Triggers
- [GM95] - Materialized views
- [HHW97], [HH99] - Online-processing
- [MRL99], [GK01] - Summarization

A Concrete Example



- An ISP that collects packet trace from two links
- Incoming packets from the link - data stream (unbounded-append only database)
- Collect packet trace - continuous query over data stream
- Conventional DBMS technology is inadequate

With Load As

(Select sadd, daddr, sum(length) as traffic

From PT_b

Group By saddr, daddr)

Select sadd, daddr,, traffic

From Load As L_1

Where (Select count(*)

From Load as L_2

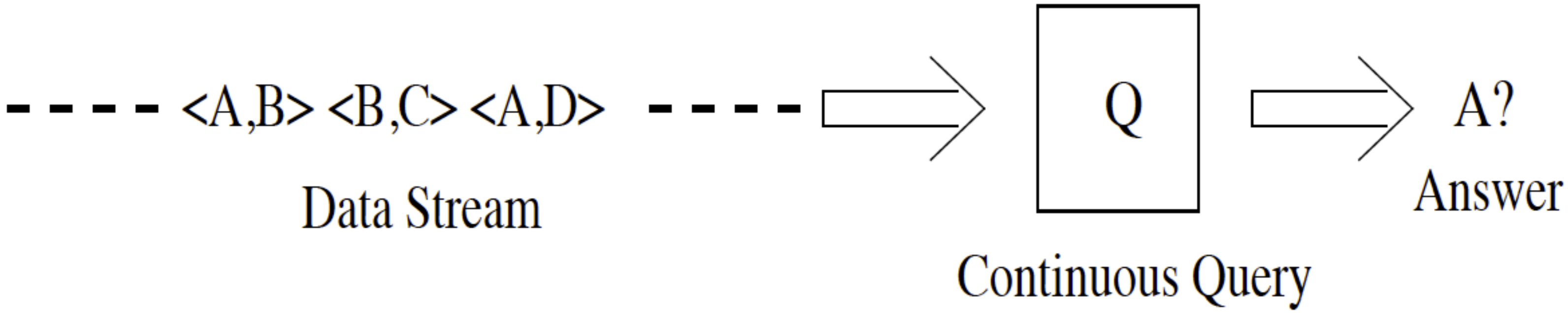
Where L_2.traffic < L_1.traffic) >

(Select 0.95Xcount(*) From Load)

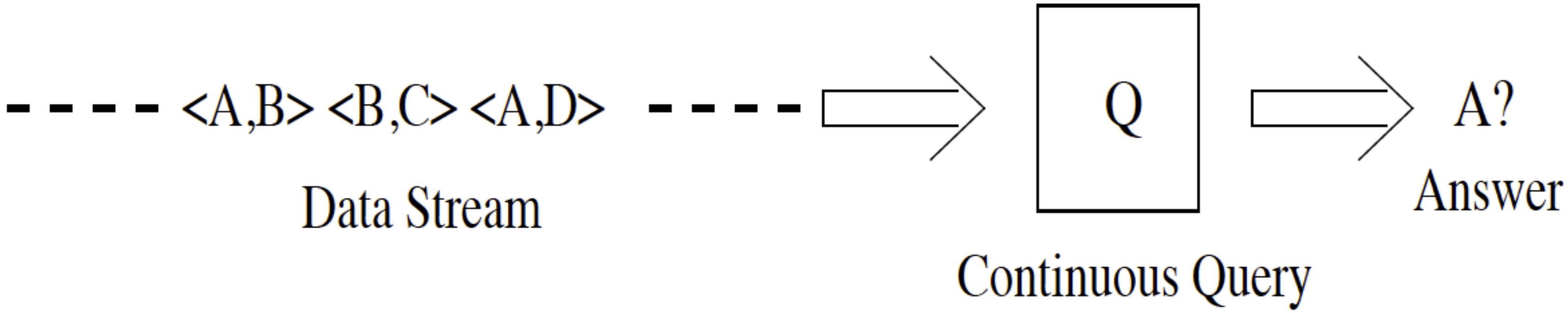
Order By traffic

Field name	Description
saddr	IP address of packet sender
daddr	IP address of packet destination
id	Identification number given by sender so that destination can uniquely identify each packet
length	Length of packet
timestamp	Time when packet header was recorded

Data Stream VS Traditional Stored Data Sets

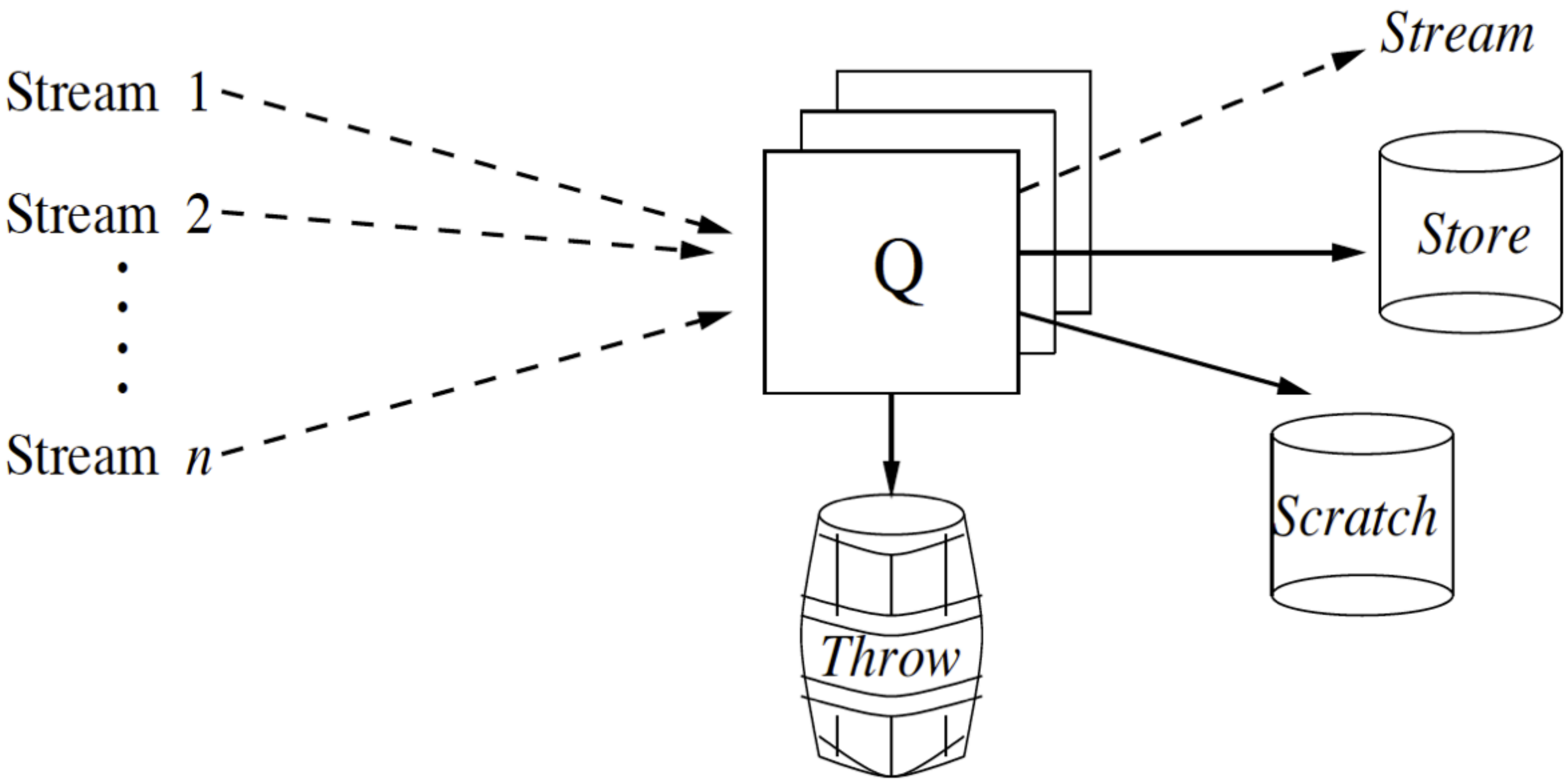


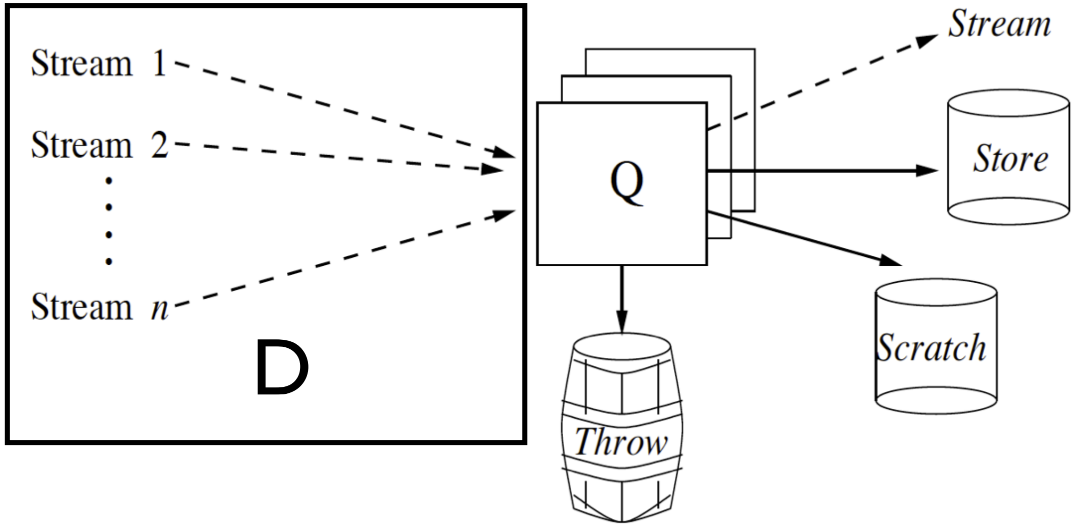
- A single, continuous stream of tuples
- A single continuous query Q
- Data stream as unbounded append-only database D



- Many possible ways to handle Q with ramifications
- E.g. Q is a selection or a group-by query
- Different ways to address such issues
- Suggested to have a new architecture

Architecture

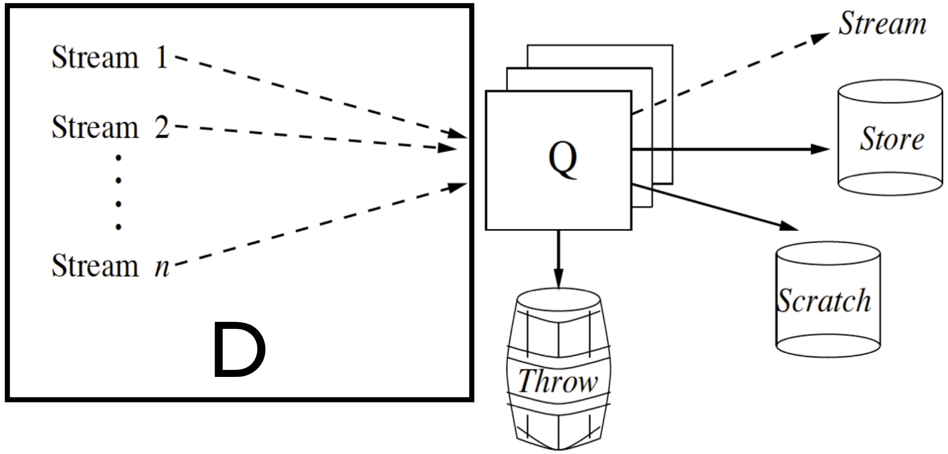




- New tuple ***a*** remain in answer A "forever" because of new tuple ***t*** from stream
  - Send the new tuple ***a*** to the ***Stream***
- New tuple ***t*** cause update or delete of ***Store***
  - Answer tuples moved from Store to Stream
- When ***t*** is not needed now or later
  - ***t*** is sent to ***Throw***

Query Processing Scenarios

- Scenario
  - Always store and make available the current answer to Q



- In terms of the architecture
  - Stream is empty
  - Store always contains A
  - Scratch contains data to keep Store up-to-date

Triggers & Materialized Views

- Triggers
  - Stream and Store may remain empty
  - Scratch store data for monitor complex events or evaluate conditions
- Materialized Views
  - Base data stored in Scratch
  - The view is maintained in Store
  - Updates to the base data represented as data streams

Basic Problems

- Online-processing
  - New tuples arrived in data stream must be "consumed" immediately
  - Some of them may need to be ignored

- Storage constraints
  - Store and/or Scratch may be unbounded size
  - Performance requirements reside in limited amount of main memory

New Techniques

- Summarization
  - Sampling, histograms, wavelets
- Online data structures
  - Data structure designed specifically to handle continuous data flow (e.g. [FW98])
- Adaptivity
  - Long-running query need to consider more parameters (e.g. amount of available memory, stream data flow rate)
  - Adaptive query processing techniques

Data Stream Management System

- Build a complete DSMS
- With similar functionalities and performance with tradition DBMS
- Build from scratch
- Complete prototype - STREAM
  - A flexible interface
  - A processor
  - A client API

Summary

- Focused on continuous queries over data stream
- Survey on previous related work
- Proposed a new architecture
- Discussed related issues and research problems
- Introduce the STREAM project

Questions?