# Data Mining for Knowledge Management

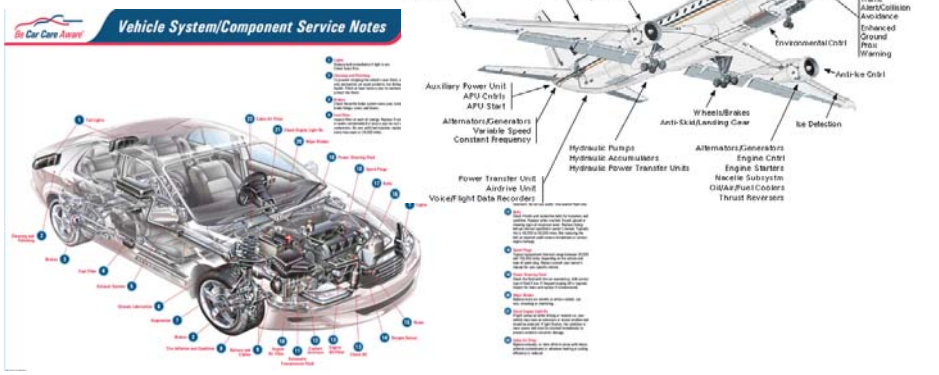
# Mining Data Streams

Themis Palpanas
University of Trento
*http://dit.unitn.it/~themis*



## Motivating Examples: Production Control System

# Motivating Examples: Monitoring Vehicle Operation





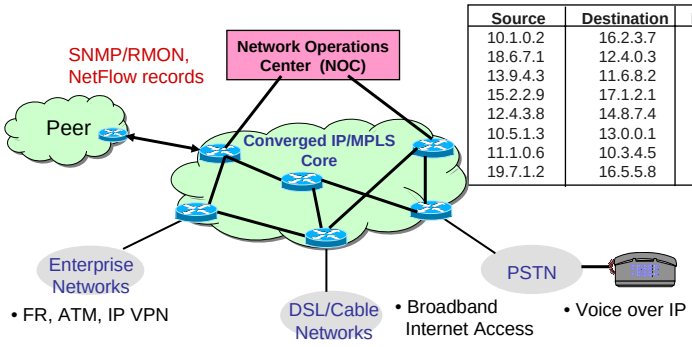

# Motivating Examples: Financial Applications

# Motivating Examples: Web Data Streams

- Mining query streams.
  - Google wants to know what queries are more frequent today than yesterday.
- Mining click streams.
  - Yahoo wants to know which of its pages are getting an unusual number of hits in the past hour.



# Motivating Examples: Network Monitoring

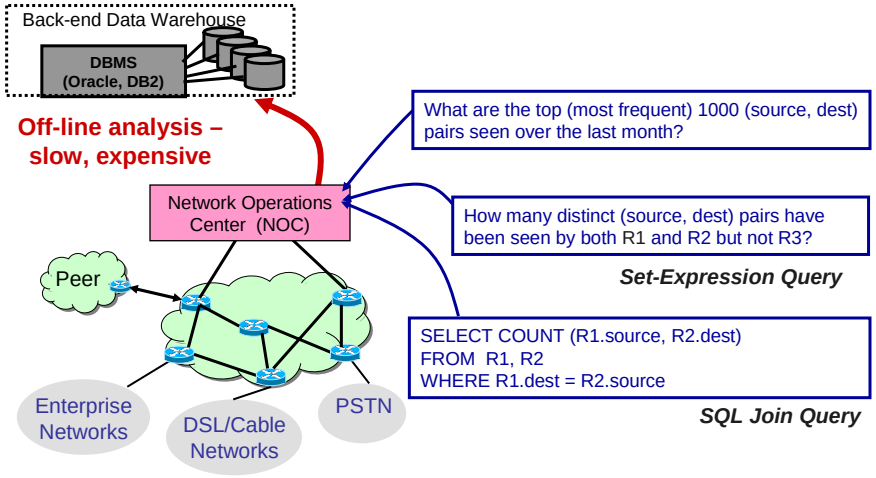



| Source | Destination | Duration | Bytes | Protocol |
|---|---|---|---|---|
| 10.1.0.2 | 16.2.3.7 | 12 | 20K | http |
| 18.6.7.1 | 12.4.0.3 | 16 | 24K | http |
| 13.9.4.3 | 11.6.8.2 | 15 | 20K | http |
| 15.2.2.9 | 17.1.2.1 | 19 | 40K | http |
| 12.4.3.8 | 14.8.7.4 | 26 | 58K | http |
| 10.5.1.3 | 13.0.0.1 | 27 | 100K | ftp |
| 11.1.0.6 | 10.3.4.5 | 32 | 300K | ftp |
| 19.7.1.2 | 16.5.5.8 | 18 | 80K | ftp |

**Example NetFlow IP Session Data**

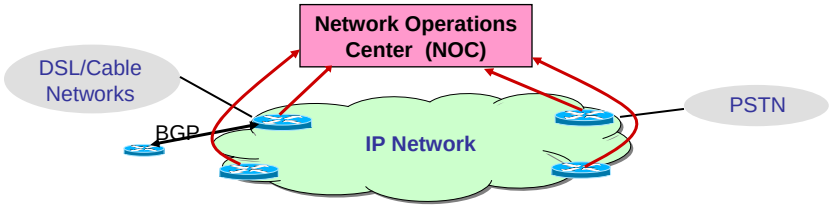
- 24x7 IP packet/flow data-streams at network elements
- Truly massive streams arriving at rapid rates
  - AT&T collects 600-800 Gigabytes of NetFlow data each day.
- Often shipped off-site to data warehouse for off-line analysis

# Motivating Examples: Network Monitoring





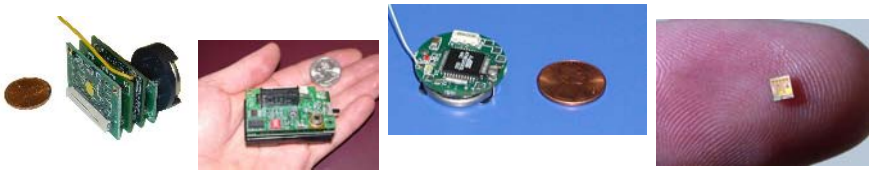

# Motivating Examples: Network Monitoring




- Must process network streams in *real-time* and *one pass*
- Critical NM tasks: fraud, DoS attacks, SLA violations
  - Real-time traffic engineering to improve utilization
- Tradeoff communication and computation to reduce load
  - Make responses fast, minimize use of network resources
  - Secondarily, minimize space and processing cost at nodes

# Motivating Examples: Sensor Networks

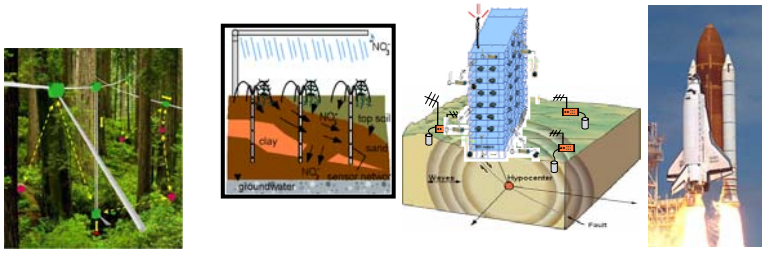
- the sensors era
  - ubiquitous, small, inexpensive sensors
  - applications that bridge physical world to information technology





# Motivating Examples: Sensor Networks

- the sensors era
  - ubiquitous, small, inexpensive sensors
  - applications that bridge physical world to information technology
- sensors unveil previously unobservable phenomena



# Requirements

- develop efficient streaming algorithms
  - need to process this data online
  - allow approximate answers
  - operate in a distributed fashion (network as distributed database)
  - can also be used as one-pass algorithms for massive datasets



# Requirements

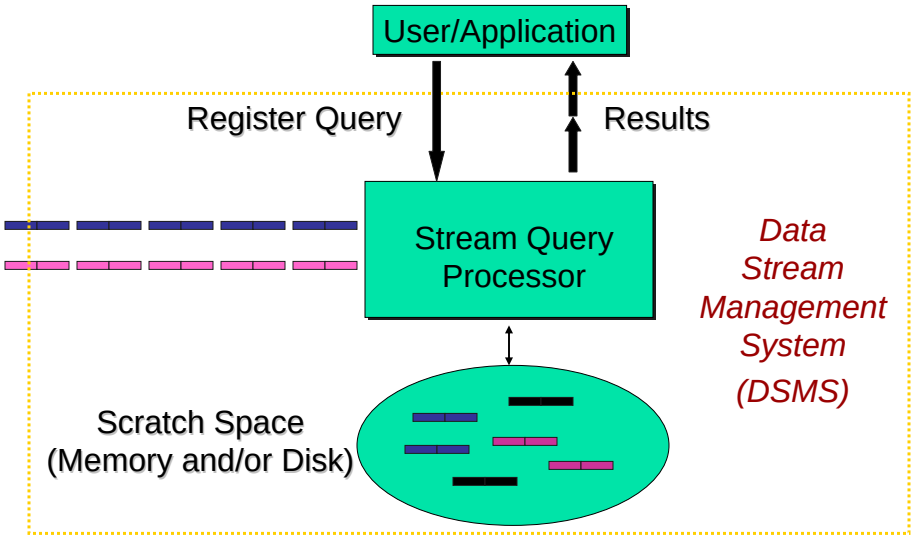
- develop efficient streaming algorithms
  - need to process this data online
  - allow approximate answers
  - operate in a distributed fashion (network as distributed database)
  - can also be used as one-pass algorithms for massive datasets
- propose new data mining algorithms
  - help in data analysis in the above setting

## Data Stream Management System?

- Traditional DBMS – data stored in finite, persistent data sets
- New Applications – data input as continuous, ordered data streams
  - Network monitoring and traffic engineering
  - Telecom call records
  - Network security
  - Financial applications
  - Sensor networks
  - Manufacturing processes
  - Web logs and clickstreams
  - Massive data sets



## Data Stream Management System!



# Meta-Questions

- Killer-apps
  - Application stream rates exceed DBMS capacity?
  - Can DSMS handle high rates anyway?
- Motivation
  - Need for general-purpose DSMS?
  - Not ad-hoc, application-specific systems?
- Non-Trivial
  - DSMS = merely DBMS with enhanced support for triggers, temporal constructs, data rate mgmt?

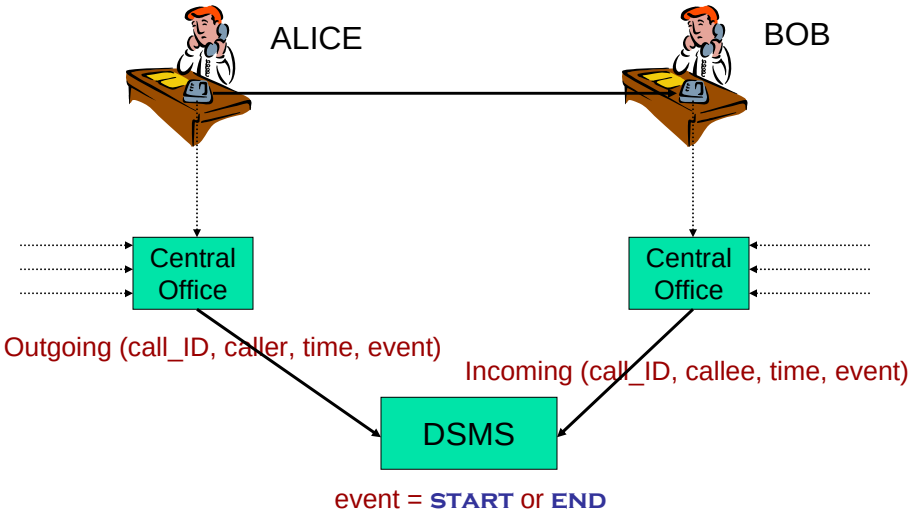


# DBMS versus DSMS

- Persistent relations
- One-time queries
- Random access
- "Unbounded" disk store
- Only current state matters
- Passive repository
- Relatively low update rate
- No real-time services
- Precise answers
- Access plan determined by query processor, physical DB design

- Transient streams
- Continuous queries
- Sequential access
- Bounded main memory
- History/arrival-order is critical
- Active stores
- Possibly multi-GB arrival rate
- Real-time requirements
- Imprecise/approximate answers
- Access plan dependent on variable data arrival and data characteristics

# Making Things Concrete






# Query 1 (SELF-JOIN)

- Find all outgoing calls longer than 2 minutes

```
SELECT   O1.call_ID, O1.caller
FROM     Outgoing O1, Outgoing O2
WHERE    (O2.time – O1.time > 2
            AND   O1.call_ID = O2.call_ID
            AND   O1.event = START
          AND   O2.event = END)
```

- Result requires unbounded storage
- Can provide result as data stream
- Can output after 2 min, without seeing END

# Query 2 (JOIN)

- Pair up callers and callees

```
SELECT   O.caller, I.callee
FROM     Outgoing O, Incoming I
WHERE    O.call_ID = I.call_ID
```

- Can still provide result as data stream
- Requires unbounded temporary storage ...
- ... unless streams are near-synchronized



# Query 3 (group-by aggregation)

- Total connection time for each caller

```
SELECT      O1.caller, sum(O2.time – O1.time)
FROM        Outgoing O1, Outgoing O2
WHERE       (O1.call_ID = O2.call_ID
              AND O1.event = START
            AND  O2.event = END)
GROUP BY    O1.caller
```

- Cannot provide result in (append-only) stream
  - Output updates?
  - Provide current value on demand?
  - Memory?

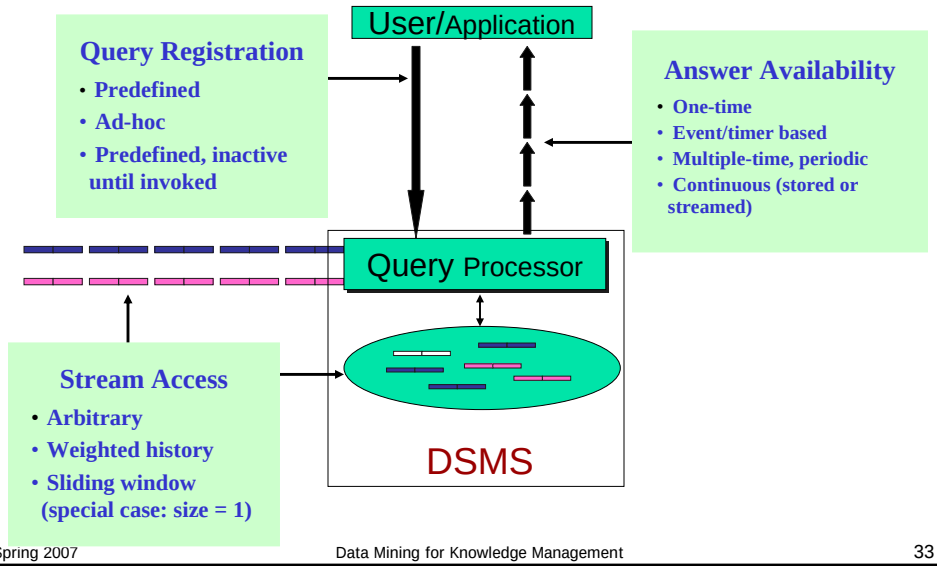
# Data Model

- Append-only
  - Call records
- Updates
  - Stock tickers
- Deletes
  - Transactional data
- Meta-Data
  - Control signals, punctuations

System Internals – probably need all above



# Query Model



## Related Database Technology

- DSMS must use ideas, but none is substitute
  - Triggers, Materialized Views in Conventional DBMS
  - Main-Memory Databases
  - Distributed Databases
  - Pub/Sub Systems
  - Active Databases
  - Sequence/Temporal/Timeseries Databases
  - Realtime Databases
  - Adaptive, Online, Partial Results
- Novelty in DSMS
  - Semantics: input ordering, streaming output, ...
  - State: cannot store unending streams, yet need history
  - Performance: rate, variability, imprecision, ...



## Stream Projects

- Amazon/Cougar (Cornell) – sensors
- Borealis (Brown/MIT) – sensor monitoring, dataflow
- Hancock (AT&T) – telecom streams
- Niagara (OGI/Wisconsin) – Internet XML databases
- OpenCQ (Georgia) – triggers, incr. view maintenance
- Stream (Stanford) – general-purpose DSMS
- Tapestry (Xerox) – pub/sub content-based filtering
- Telegraph (Berkeley) – adaptive engine for sensors
- Tribeca (Bellcore) – network monitoring