Scalable Pattern Sharing over Event Streams

Medhabi Ray (Microsoft)
Chuan Lei (NEC Labs America)*
Elke A. Rundensteiner (WPI)

06/28/2016
Motivation – CEP

Event Producers

Event Processing Engine

Event Pattern Consumers

Event Streams

Event Patterns
Motivation – Multiple CEP

Traffic Event Stream

Multiple Pattern Queries

Q1: SEQ (CarLoc c (c.pos = tollA), CarLoc c1 (c1.pos = WorkZone, c1.id = c.id), CarLoc c2 (c2.pos = exitA & c.id = c2.id))

Q2: SEQ (CarLoc c (c.pos = tollA), Accident a (a.pos = tollA + 5 miles), CarLoc c1 (c1.pos = exitA & c.id = c1.id))

Q3: SEQ (CarLoc c (c.pos = tollA), CarLoc c1 (c1.pos = c.pos & c1.type = EMS & c1.state = Parked), CarLoc c2 (c2.pos = exitA & c.id = c2.id))

Q4: SEQ (CarLoc c (c.pos = tollA), CarLoc c1 (c1.pos = WorkZone, c1.id = c.id), CarLoc c2 (c2.pos = exitB & c.id = c2.id))

Observation

Multiple Pattern queries share common sub-patterns and are often run concurrently.
Challenges – Time-Ordered Sequence Correlation

Q1: SEQ (CarLoc c (c.pos = tollA),
       CarLoc c1 (c1.pos = WorkZone, c1.id=c.id),
       CarLoc c2 (c2.pos = exitA & c.id=c2.id))

Q2: SEQ (CarLoc c (c.pos = tollA),
        Accident a (a.pos = tollA+5 miles),
        CarLoc c1 (c1.pos = exitA & c.id=c1.id))

Q3: SEQ (CarLoc c (c.pos = tollA),
        CarLoc c1 (c1.pos = c.pos & c1.type= EMS & c1.state = Parked),
        CarLoc c2 (c2.pos = exitA & c.id=c2.id))

Q4: SEQ (CarLoc c (c.pos = tollA),
        CarLoc c1 (c1.pos = WorkZone, c1.id=c.id),
        CarLoc c2 (c2.pos = exitB & c.id=c2.id))

$T_1 = 4:00 \text{ PM} \quad T_2 = 4:10 \text{ PM} \quad T_3 = 4:15 \text{ PM}$
Challenges – Inter-Query Event Correlation

Q1: SEQ (CarLoc c (c.pos = tollA),
    CarLoc c1 (c1.pos = WorkZone, c1.id=c.id),
    CarLoc c2 (c2.pos = exitA & c.id=c2.id))

Q2: SEQ (CarLoc c (c.pos = tollA),
    Accident a (a.pos = tollA+5 miles),
    CarLoc c1 (c1.pos = exitA & c.id=c1.id))

Q3: SEQ (CarLoc c (c.pos = tollA),
    CarLoc c1 (c1.pos = c.pos & c1.type= EMS & c1.state = Parked),
    CarLoc c2 (c2.pos = exitA & c.id=c2.id))

Q4: SEQ (CarLoc c (c.pos = tollA),
    CarLoc c1 (c1.pos = WorkZone, c1.id=c.id),
    CarLoc c2 (c2.pos = exitB & c.id=c2.id))
Challenges – Inter-Query Event Correlation

Q1: SEQ (CarLoc c (c.pos = tollA),
    CarLoc c1 (c1.pos = WorkZone, c1.id=c.id),
    CarLoc c2 (c2.pos = exitA & c.id=c2.id))

Q2: SEQ (CarLoc c (c.pos = tollA),
    Accident a (a.pos = tollA+5 miles),
    CarLoc c1 (c1.pos = exitA & c.id=c1.id))

Q3: SEQ (CarLoc c (c.pos = tollA),
    CarLoc c1 (c1.pos = c.pos & c1.type= EMS & c1.state = Parked),
    CarLoc c2 (c2.pos = exitA & c.id=c2.id))

Q4: SEQ (CarLoc c (c.pos = tollA),
    CarLoc c1 (c1.pos = WorkZone, c1.id=c.id),
    CarLoc c2 (c2.pos = exitB & c.id=c2.id))
Challenges – Intractable Search Space

Q1: SEQ (A, B, C, D, E)
Q2: SEQ (A, B, M, D, P)
Q3: SEQ (X, B, M, N, Y)
Q4: SEQ (X, B, C, D, P)

Q1: SEQ (A, B, C, D, E)
Q2: SEQ (A, B, M, D, P)
Q3: SEQ (X, B, M, N, Y)
Q4: SEQ (X, B, C, D, P)

Shared Plan

Q1: SEQ (A, B, C, D, E)
Q2: SEQ (A, B, M, D, P)
Q3: SEQ (X, B, M, N, Y)
Q4: SEQ (X, B, C, D, P)
SPASS Architecture

Output Pattern Matches

SPASS Runtime
- Shared Continuous Sliding Views
- Pattern Executor
- Statistics Collector
- View Manager

SPASS Optimizer
- RR Estimator
- Sub-Pattern Selector
- Shared Pattern Plan Generator

Pattern Workload

Event Stream

SPASS – Scalable Pattern Sharing on event Streams

Worcester Polytechnic Institute
SPASS Optimizer

- **Intra-query event correlation**
  - Number of event instances of $E_j$ follow an event of $E_i$
  - Estimate # sub-pattern matches formed in a time interval

- **Inter-query event correlation**
  - Ratio between # of matches computed with sharing and # of matches computed separately
  - Estimate the degree of sharing across multiple patterns

- **Redundancy Ratio (RR)**
  - Estimate the degree of the redundant computation of matches using both Intra- and Inter-Query Event Correlations
Workload
- Q1: SEQ(A,B,C)
- Q2: SEQ(B,C,D)
- Q3: SEQ(A,B,C,D)
- Q4: SEQ(B,C)

Query Map
- Q1: <A, BC>
- Q2: <BC, D>
- Q3: <A, BC, D>
- Q4: <BC>

Shared Pattern Plan
- SEQ(A,B,C) | {Q1}
- SEQ(A,B,C,D) | {Q2}
- SEQ(B,C,D) | {Q3}
- SEQ(D) | {Q2, Q3}

Identify shared sub-patterns
Build shared pattern plan
SPASS Runtime

- Challenge – concurrent access to shared sub-pattern matches
- Solution – shared continuous sliding views store intermediate results of sub-patterns for subsequent reuse
  - View Validity Interval (VVI) timestamp-based indicators associated with materialized views
  - View Lookup Interval (VLI) a time interval to look up pattern matches
Experimental Results – Optimizer

• SPASS optimizer efficiently produces a nearly-optimal solution
  – Length of queries (5 – 20 event types)

Vary the length of queries

![Graphs showing accuracy and CPU time for different maximum lengths of queries.](image-url)

Worcester Polytechnic Institute
Experimental Results – Runtime

• Average execution time
  – SPASS outperforms prefix-share, suffix-share and random sub-expression sharing approaches
  – SPASS statistics collection introduces negligible overheads
Conclusions

• Establish a model for sharing benefit estimation
  – Capturing overlapping event ordering among pattern queries
  – Capturing inter-query event correlation

• SPASS optimizer identifies shared sub-patterns in polynomial-time with proven bounds on optimality

• SPASS runtime facilitates shared processing of common sub-patterns with shared continuous sliding views
Questions?

Contact: Chuan Lei (chuan@nec-labs.com)