Towards Scalable Complex Event Processing

Fighting the Exponentiality of Event Pattern Detection

Ilya Kolchinsky
Assaf Schuster
Overview

• Complex event processing (CEP)
• The reason CEP is difficult
• Data-aware CEP
• Lazy evaluation in CEP
• Join methods for CEP
• Adaptive CEP
Complex Event Processing

Traditional DB
- Static, mostly relational data
- “Classic” SQL queries, joins, etc.
- A well-established field since the 70s

Stream Processing
- Data streams instead of tables
- Tight real-time requirements
- Very limited local memory
- Mostly aggregation queries (heavy hitters, distinct items, etc.)

Complex Event Processing
- A data item is viewed as a **primitive event**
- Primitive events are combined into **complex events** which conform to user-defined **patterns**
- The goal is to detect complex event occurrences in the input stream(s)
SEQ(MainLobbyCameraEvent a, CorridorCameraEvent b, RestrictedAreaCameraEvent c) WHERE (a.person_id == b.person_id == c.person_id)
CEP Example 2 – Monitoring Stock Prices

SEQ(GoogleStockPriceUpdate a, MicrosoftStockPriceUpdate b, AppleStockPriceUpdate c)

WHERE

((a.price < b.price) AND (b.price < c.price))

WITHIN 10 minutes
CEP Applications

- Surveillance Systems
- Stock Monitoring
- Internet of Things
- Online Finance
- Fraud Detection
- Network Security Monitoring
- Sensor Networks
- Algorithmic Trading
Pattern Types in CEP

Disjunctions
- at least one of the specified events must be occur

Conjunctions
- all events specified in the pattern must occur

Sequences
- all events must occur in the predefined order
<table>
<thead>
<tr>
<th>Pattern Types in CEP – contd.</th>
</tr>
</thead>
</table>

**Negations**
- some events are prohibited from occurring at the specified positions

**Kleene closure**
- some events may appear an unlimited number of times

**Nested patterns**
- arbitrary combinations of all of the above
CEP Evaluation Mechanisms

• Non-deterministic Finite Automata

• Evaluation Trees
SEQ(GoogleStockPriceUpdate a, MicrosoftStockPriceUpdate b, AppleStockPriceUpdate c)
WHERE ((a.price < b.price) AND (b.price < c.price))
WITHIN 10 minutes
NFA-Based Detection Example

∅

a₁

a₂

a₃

a₁ a₂ a₃

a₁ a₂ a₃

a₁ a₂ a₃

a₁ b₁ b₂ b₃

a₂ b₁ b₂ b₃

a₃ b₁ b₂ b₃

a₁ b₃ c₁

a₂ b₃ c₁

a₃ b₃ c₁

a₁ p=1 a₂ p=1 a₃ p=4 b₁ p=7 b₂ p=9 b₃ p=5 c₁ p=6
Tree Based Evaluation

SEQ(
  GoogleStockPriceUpdate a,
  MicrosoftStockPriceUpdate b,
  AppleStockPriceUpdate c)
WHERE ((a.price < b.price) AND (b.price < c.price))
WITHIN 10 minutes
Tree-Based Detection Example
Complex Event Processing is difficult

- The number of partial matches to be maintained during the detection process is exponential in pattern size!

Traditional human-generated patterns are simple and short...
But recent machine-generated patterns are more sophisticated – an algorithmic challenge!
Data-Aware CEP

• Define multiple evaluation plans for a given pattern
• Leverage the available knowledge on statistical data properties to select the best plan
• Possible for trees 😊
• No such model for NFA 😞
SEQ(A a, B b, C c)
WHERE (a.price < b.price) AND (b.price < c.price)
WITHIN 1 hour

Cost(Plan1) = 228.05
Cost(Plan2) = 300.42.5

rate(A) = rate(B) = rate(C) = 10
sel(a.price < b.price) = 0.25
sel(b.price < c.price) = 0.25
Lazy Evaluation Model for NFA

[DEBS’15 Best Paper Award]

- Process incoming events according to arbitrary order (rather than in order of appearance)
- Keep the unprocessed events in an intermediate storage
- Process the buffered events only when required by the plan in use
Lazy NFA – (order c,a,b)

SEQ(A a, B b, C c)
WHERE (a.price < b.price) AND (b.price < c.price)
WITHIN 1 hour
Lazy NFA – (order \(b,a,c\))

SEQ(A \(a\), B \(b\), C \(c\))
WHERE \((a.\text{price} < b.\text{price})\) AND \((b.\text{price} < c.\text{price})\)
WITHIN 1 hour
What is the best evaluation order?

• A simple solution: order the events from the rarest to the most frequent
  – assume the frequencies are given and are not changing
Lazy NFA Pattern Detection Example
(order \(c,b,a\))

- \(\emptyset\)
- \(c_1\)
- \(c_1 b_3\)

Input Buffer:

- \(a_1^{p=1}\)
- \(a_2^{p=1}\)
- \(a_3^{p=4}\)
- \(b_1^{p=7}\)
- \(b_2^{p=9}\)
- \(b_3^{p=5}\)
- \(c_1^{p=6}\)
Join Order Estimation

• A well-known problem since the 80’s
• Given $n$ relations joining on $m$ attributes, what is the most efficient way to perform the join?
• Extensively studied, lots of algorithms published

• Hmmmm…. So…….
• Can an instance of CEP plan generation problem be transformed into this problem?
• [Submission-I 2018]
Join Query Plan Types

Left Deep Tree
(resembles NFA?)

Bushy Tree
(resembles CEP Trees?)
Problem Equivalence

Conjunctive CEP Plan Generation = Join Plan Generation

Non-Conjunctive CEP Plan Generation + Reduction to Conjunctive = Join Plan Generation
Existing Algorithms

- Greedy
- Dynamic Programming
- Local Search
- Hybrid
- Mixed Integer Programming
- Genetic Algorithms
- Graph Theory-based

Join Plan Generation Algorithms
Experimental Results – NFA throughput
Data-Aware CEP in Practice

• Will the data-aware method work in a real-life scenario?
  • Probably not
    – arrival rates / selectivities not known in advance
    – arrival rates / selectivities subject to frequent changes
  • Adaptation is needed
Adaptive CEP

Input Stream(s)

Statistics Collector

• Collects incoming events
• Estimates data statistics on the fly
• Event arrival rates
• Predicate selectivities
• Data distribution

Evaluation Mechanism

• Accepts data statistics from the collector
• Decides whether a reoptimization is to be attempted
• If needed, applied the plan generation algorithm to create a new plan

Pattern Specifications

• Actually handles pattern detection
• Accepts up-to-date evaluation plans from the optimizer
• Responsible for deployment of new plans
• Actually handles pattern detection
• Accepts up-to-date evaluation plans from the optimizer
• Responsible for deployment of new plans
Possible Adaptation Strategies

• Periodically recalculate the evaluation plan based on new statistics
• Monitor all statistic values against a predefined threshold
• Can we do better?
Invariant-based Method [Submission-II 2018]

SEQ(A a, B b, C c)
WHERE (a.price < b.price) AND (b.price < c.price)
WITHIN 1 hour

Recalculating evaluation plans and invariants...

<table>
<thead>
<tr>
<th>Rate</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>100</td>
</tr>
<tr>
<td>B</td>
<td>50</td>
</tr>
<tr>
<td>C</td>
<td>10</td>
</tr>
</tbody>
</table>

Is rate(B) > rate(A)?

Tighten condition found!

rate(A) = 20
rate(B) = 50
rate(C) = 10
Experimental Results - NFA
Summary

• Incorporating data statistics allows a CEP system to achieve a performance boost of orders of magnitude (Still not enough 😞 )

• Detection latency and memory consumption are also significantly improved

• Many open questions remain
QUESTIONS?

ARARAT

PEACE BARREL

Set for ageing in honor of the visit of co-chairs of OSCE Minsk Group and will be opened when Karabakh conflict is resolved.

Yerevan 2001