Semantically Partitioned Peer to Peer Complex Event Processing Exploiting Information Loss

Filip Nguyen
xnguyen@fi.muni.cz

Masaryk University
Faculty of Informatics
Botanická 68a
602 00 Brno
Czech Republic

CZ.1.07/2.4.00/17.0041
Platform for Industrial Cooperation FI MU

Lab of Software Architectures and Information Systems FI MU

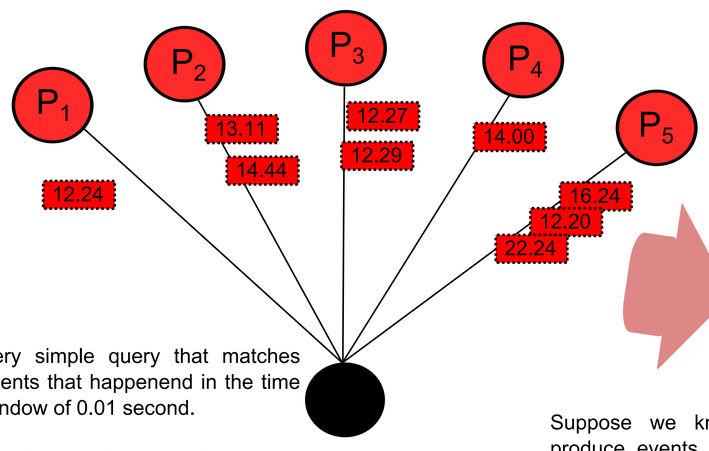
Abstract

Scaling Complex Event Processing [6] (CEP) applications is inherently problematic. Our solution for scaling CEP applications is fully distributed and aspires to scale CEP to the limits of current hardware. Our solution simplifies existent Event Processing Network abstraction and adds features on the level of CEP that change direction of its usage.

Related Work

Complex Event Processing was introduced by David Luckham. We are mainly concerned with subarea of Luckam's work related to distributed CEP [6] (also studied by [6] and [4]).
Motivation of our work stems from our work related to event processing [3]. We have applied our theoretical ideas in concepts introduced in [2] and gave brief introduction to our overall research in [1].

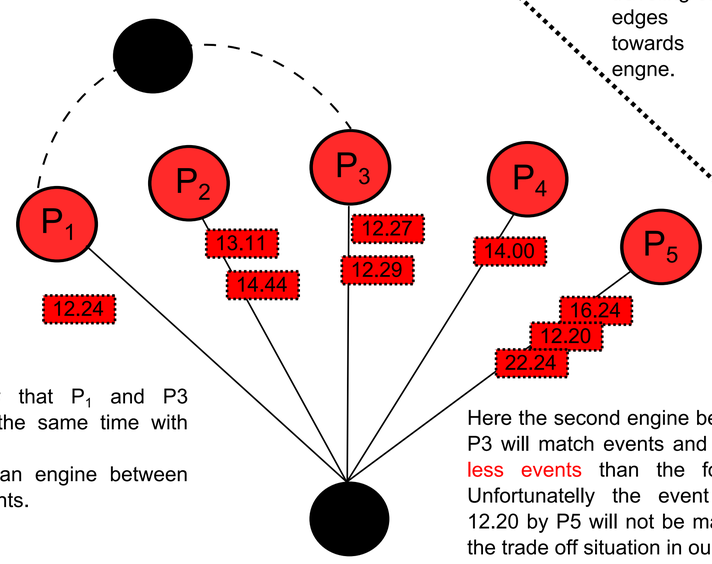
Basic Approach



Very simple query that matches events that happenend in the time window of 0.01 second.

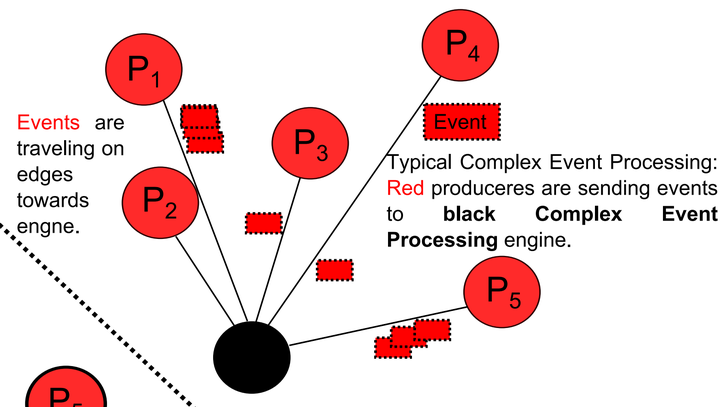
select name(E_A), name(E_B)
where abs(time(E_A) - time(E_B))
< 0.01s
&& E_A!=E_B

This query needs all the produced events.



Suppose we know that P_1 and P3 produce events at the same time with high probability
Then we can add an engine between them and match events.

Here the second engine between P1 and P3 will match events and is loaded with less events than the former engine. Unfortunatelly the event produced at 12.20 by P5 will not be matched. This is the trade off situation in our solution.



Events are traveling on edges towards engne.

Typical Complex Event Processing: Red produceres are sending events to **black Complex Event Processing** engine.

Event

The definition of an event varies based on context of CEP. However some parameters are always the same. Each event has defined *time of creation* and *producer.*
Events should be as fine grained as possible - to allow effective CEP. That means thousands, even milions of events per second are desirable. This is not uncommon thing today with advent of social networks, faster networking hardware and computer driven high frequency trading.

Distributed CEP

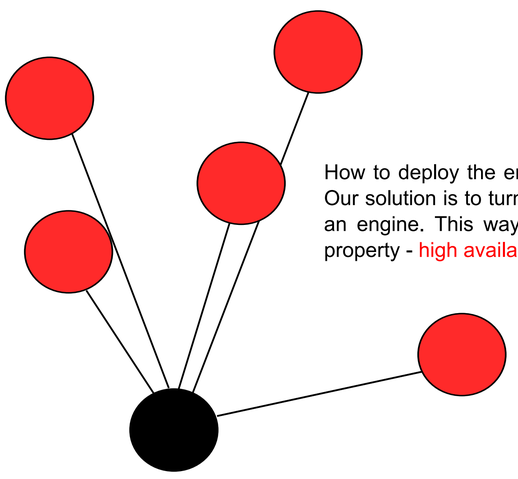
There is ongoing research to distribute CEP. Every author makes his own definition of distributed CEP. Usually, it refers to a use of filters on producers or parallelizing existing CEP operators. We see distributed CEP differently. We aim to distribute the processing at semantic level. We do not want to just filter unknown events. We allow users to leverage standard operators and give them framework to easily trade off processing power with matching precision.

Partitioning Algorithm

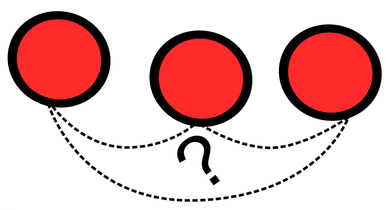
Partitioning algorithm is used to select sets of producers among which the engine is placed. This is equivallent to creating channels between peers.

Of course, the number of these sets is exponential, thus it is necessary to select such amount that will balance the trade off.

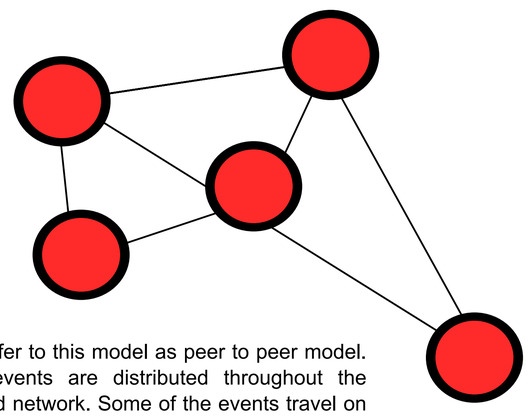
Peer Network



How to deploy the engines dynamically? Our solution is to turn each producer into an engine. This way we gain additional property - high availability.



A node is said to be a peer.



We refer to this model as peer to peer model. The events are distributed throughout the formed network. Some of the events travel on dedicated paths, some are broadcasted. This behavior is based on the result of partitioning algorithm.

CEP Based and Monte Carlo Algorithm

There are many possible partitioning algorithms. Beginning with the manual, CEP based partitioning algorithm. This algorithm is dynamic (selects subsets of peers at runtime of matching). Input for this algorithm is CEP rule provided by user. For example:

subselect originator(CE_A)
where abs(time(CE_A) - time(CE_B)) < 1 minute

The CEP query above will ensure that channel will be created among those peers that generated coarse grained events in window of 1 minute.

Another, fully automated, partitioning algorithm is Monte Carlo partitioning algorithm. This algorithm is based on statistical analysis of events firing and producers that fired the event. This way, the peer network automatically makes itself more effective.

Coarse Grained Event

Event is relatively small, high frequency event, e. g. scanning of barcode in supermarket.
Coarse Event is a notion used to describe event that happens much less frequently than Event, e.g. payment by credit card in supermarket.

Results

We believe that fully distributed peer to peer CEP is inevitable solution to high volume event streams. Our implementation of presented concept is called peer CEP (PCEP). The main property of PCEP is semantic scaling. The scaling is not done by brute force or by exploiting specific feature of specific event context, but it is done by exploiting partitioning of peers according to their's affiliation to matching rules.
The developed distributed engine is written in Java and thus runs on heterogenous platforms. In the implementation we leverage distributed algorithms developed in theirs natural form - not optimized to the state of being obfuscated code.
In theoretical point of view, our solution introduces rigorously defined trade off between matching capabilities and throughoutput of the events. In the future we plan to extend this knowledge by revealing statistical properties of mentioned trade off situation.

There is another result we present - partitioning algorithms. We believe those algorithms may be extended and generalized to be used in other fields for set partitioning and analysis of data sets. These algorithms join several Distributed algorithms, Statistics and Complex Event Processing. We theorize, that the partitioning may be done in a distributed fashion.
We also believe in adoption by users. We strive robust architecture. Our solution is Open Source and we plan to apply for Apache Incubation. We believe the science should be done for greater good and sharing the code will improve the implementation.
Lastly, our solution is not mutually exclusive with recent research in the area of CEP. It will be possible to use standard CEP engines on the peer nodes and thus augmenting existing tools with PCEP.

esf european social fund in the czech republic
EUROPEAN UNION
MINISTRY OF EDUCATION, YOUTH AND SPORTS
OP Education for Competitiveness
INVESTMENTS IN EDUCATION DEVELOPMENT

References

[1] Nguyen F., Pitner T. 2012. *Information system monitoring and notifications using complex event processing*.
In Proceedings of the Fifth Balkan Conference in Informatics (BCI '12). ACM, New York, NY, USA, 211-216.
[2] Kunc P., Nguyen F., Pitner T. 2013. *Towards Effective Social Network System Implementation*.
New Trends in Databases and Information Systems Advances in Intelligent Systems and Computing. Springer Berlin Heidelberg, 327-336.
[3] Nguyen F., Škrabálek J. 2011 *NotX service oriented multi-platform notification system*.
In Computer Science and Information Systems (FedCSIS). Szczecin, Poland, 313-316.
[4] Wu, E., Diao, Y., Rizvi, S. 2006. *High-performance complex event processing over streams*.
In Proceedings of the 2006 ACM SIGMOD international conference on Management of data - SIGMOD '06.
[5] Akram, S., Marazakis, M., Bilas, A. 2012. *Understanding and improving the cost of scaling distributed event processing*.
In Proceedings of the 6th ACM International Conference on Distributed Event-Based Systems (DEBS '12). ACM, New York, NY, USA, 290-301.
[6] Luckham, D. C., Frasca, B. 1998. *Complex Event Processing in Distributed Systems*.
In Standford University, Vol 28.

Links

Github: https://github.com/nguyenfilip/pcep
LaSArIS: http://lasaris.fi.muni.cz/
LinkedIn: http://www.linkedin.com/pub/filip-nguyen/27/60/5b4
University: http://www.fi.muni.cz