# Complex Event Processing: Languages, Recognition and Forecasting

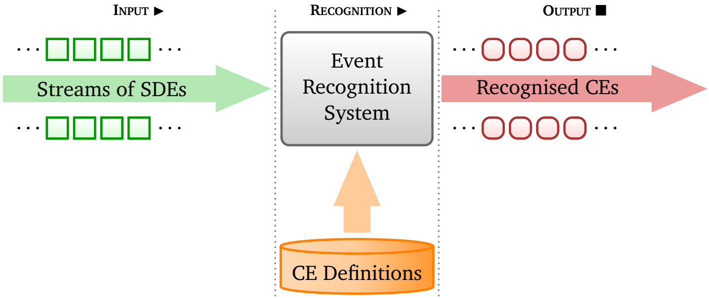
KR 2021 Tutorial

Elias Alevizos[1] and Alexander Artikis[2,1]

[1]NCSR Demokritos, [2]University of Piraeus
alevizos.elias@iit.demokritos.gr, a.artikis@unipi.gr

http://cer.iit.demokritos.gr

# Complex Event Recognition (Event Pattern Matching)



# Human Activity Recognition

# Human Activity Recognition


# Human Activity Recognition

| Input | Output |
| --- | --- |
| 340 $inactive(id_0)$ | |
| 340 $p(id_0) = (20.88, -11.90)$ | |
| 340 $appear(id_0)$ | |
| 340 $walking(id_2)$ | |
| 340 $p(id_2) = (25.88, -19.80)$ | |
| 340 $active(id_1)$ | |
| 340 $p(id_1) = (20.88, -11.90)$ | |
| 340 $walking(id_3)$ | |
| 340 $p(id_3) = (24.78, -18.77)$ | |
| 380 $walking(id_3)$ | |
| 380 $p(id_3) = (27.88, -9.90)$ | |
| 380 $walking(id_2)$ | |
| 380 $p(id_2) = (28.27, -9.66)$ | |

# Human Activity Recognition

| Input | | Output | |
|---|---|---|---|
| 340 | $inactive(id_0)$ | 340 | $left\_object(id_1, id_0)$ |
| 340 | $p(id_0) = (20.88, -11.90)$ | | |
| 340 | $appear(id_0)$ | | |
| 340 | $walking(id_2)$ | | |
| 340 | $p(id_2) = (25.88, -19.80)$ | | |
| 340 | $active(id_1)$ | | |
| 340 | $p(id_1) = (20.88, -11.90)$ | | |
| 340 | $walking(id_3)$ | | |
| 340 | $p(id_3) = (24.78, -18.77)$ | | |
| 380 | $walking(id_3)$ | | |
| 380 | $p(id_3) = (27.88, -9.90)$ | | |
| 380 | $walking(id_2)$ | | |
| 380 | $p(id_2) = (28.27, -9.66)$ | | |

# Human Activity Recognition

| Input | Output |
|---|---|
| 340 $inactive(id_0)$ | 340 $left\_object(id_1, id_0)$ |
| 340 $p(id_0) = (20.88, -11.90)$ | $since(340)$ $moving(id_2, id_3)$ |
| 340 $appear(id_0)$ | |
| 340 $walking(id_2)$ | |
| 340 $p(id_2) = (25.88, -19.80)$ | |
| 340 $active(id_1)$ | |
| 340 $p(id_1) = (20.88, -11.90)$ | |
| 340 $walking(id_3)$ | |
| 340 $p(id_3) = (24.78, -18.77)$ | |
| 380 $walking(id_3)$ | |
| 380 $p(id_3) = (27.88, -9.90)$ | |
| 380 $walking(id_2)$ | |
| 380 $p(id_2) = (28.27, -9.66)$ | |

# Credit Card Fraud Recognition


SDE:

- Credit card transactions from all over the world.

CE:

- Cloned card — a credit card is being used simultaneously in different countries.
- New high use — the card is being frequently used in merchants or countries never used before.
- Potential batch fraud — many transactions from multiple cards in the same point-of-sale terminal in high amounts.

# Credit Card Fraud Recognition


SPEEDD

- Fraud must be detected within 25 milliseconds.

# Credit Card Fraud Recognition


SPEEDD

- Fraud must be detected within 25 milliseconds.
- Fraudulent transactions: 0.1% of the total number of transactions.

# Credit Card Fraud Recognition


SPEEDD

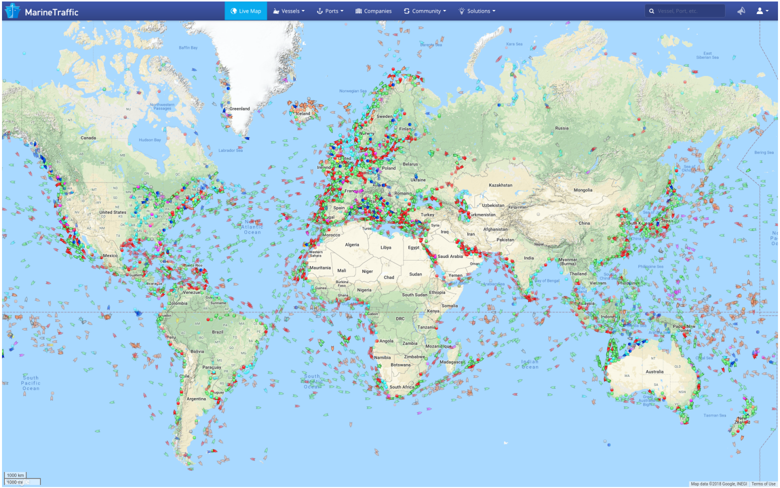
- Fraud must be detected within 25 milliseconds.
- Fraudulent transactions: 0.1% of the total number of transactions.
- Fraud is constantly evolving.

# Credit Card Fraud Recognition


SPEEDD

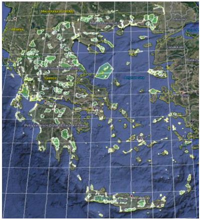
- Fraud must be detected within 25 milliseconds.
- Fraudulent transactions: 0.1% of the total number of transactions.
- Fraud is constantly evolving.
- Erroneous transactions, missing fields.

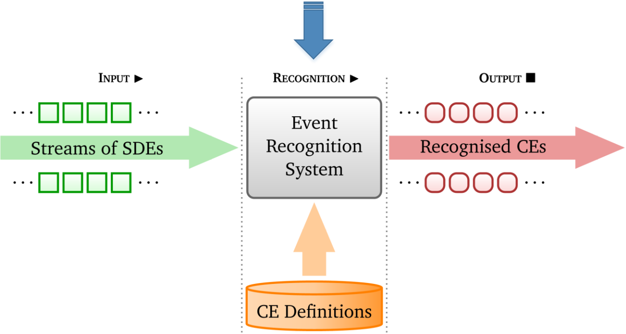
# Maritime Situational Awareness




Source: http://www.marinetraffic.com

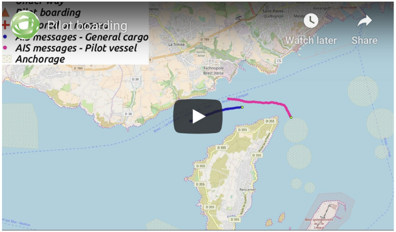
# Maritime Situational Awareness





# Maritime Situational Awareness



# Big Data Problem

- Velocity, Volume: 19,000,000 position signals/day at European scale.

# Big Data Problem

- Velocity, Volume: 19,000,000 position signals/day at European scale.
- Variety: Position signals need to be combined with other data streams
  - Weather forecasts, sea currents, etc.
- … and static information
  - NATURA areas, shallow waters areas, coastlines, etc.

# Big Data Problem

- Velocity, Volume: 19,000,000 position signals/day at European scale.
- Variety: Position signals need to be combined with other data streams
  - Weather forecasts, sea currents, etc.
- ... and static information
  - NATURA areas, shallow waters areas, coastlines, etc.
- Lack of Veracity: GPS manipulation, vessels reporting false identity, communication gaps.

# Big Data Problem

- Velocity, Volume: 19,000,000 position signals/day at European scale.
- Variety: Position signals need to be combined with other data streams
  - Weather forecasts, sea currents, etc.
- ... and static information
  - NATURA areas, shallow waters areas, coastlines, etc.
- Lack of Veracity: GPS manipulation, vessels reporting false identity, communication gaps.
- Distribution: Vessels operating across the globe.

# Many Other (Big Data) Applications

- Cardiac arrhythmia recognition.
- Identification of opportunities for refueling in fleet management.
- Identification of shipment irregularities inventory management.
- Intrusion detection in computer networks.
- Traffic congestion recognition and forecasting in smart cities.

# Requirements

- Expressive representation
  - to capture complex relationships between the events that stream into the system.

# Requirements

- Expressive representation
  - to capture complex relationships between the events that stream into the system.
- Efficient reasoning
  - to support real-time decision-making in large-scale, (geographically) distributed applications.

# Requirements

- Expressive representation
  - to capture complex relationships between the events that stream into the system.
- Efficient reasoning
  - to support real-time decision-making in large-scale, (geographically) distributed applications.
- Automated knowledge construction
  - to avoid the time-consuming, error-prone manual CE definition development.

# Requirements

- Expressive representation
  - to capture complex relationships between the events that stream into the system.
- Efficient reasoning
  - to support real-time decision-making in large-scale, (geographically) distributed applications.
- Automated knowledge construction
  - to avoid the time-consuming, error-prone manual CE definition development.
- Reasoning under uncertainty
  - to deal with various types of noise.

# Requirements

- Expressive representation
  - to capture complex relationships between the events that stream into the system.
- Efficient reasoning
  - to support real-time decision-making in large-scale, (geographically) distributed applications.
- Automated knowledge construction
  - to avoid the time-consuming, error-prone manual CE definition development.
- Reasoning under uncertainty
  - to deal with various types of noise.
- Complex event forecasting
  - to support proactive decision-making.

# Tutorial Structure

- Part I: Introduction
- Part II: Logic-based complex event recognition
- Part III: Automata-based complex event recognition
- Part IV: Complex event forecasting
- Part V: Topics not covered

# CER vs DBMSs

Traditional Database Management Systems (DBMS)s:

- Store data before processing.
- Data updates are relatively infrequent.
- Typically sort data.
- Process data only when explicitly asked by the user.

# CER vs DBMSs

Complex event recognition (CER) systems:

- Process data without storing them.
- Data are continuously updated.
  - Data stream into the system in high velocity.
  - Data streams are large (usually unbounded).
- No assumption can be made on data arrival order.
- Users install standing/continuous queries:
  - Queries deployed once and execute continuously until removed.
  - Online reasoning.
- Latency requirements are very strict.

# Models of CER Systems

- Data model.
- Time model.
- Pattern language model.
- Processing model.
- Deployment model.

# Pattern Language Model

- CER refers to matching patterns among the incoming streams of Simple, Derived Events (SDE)s.
- Thus, we need a language for expressing such patterns.
- We present a simple event algebra with common operators.
- Some systems extend this algebra with additional operators.

# A Simple Unifying Event Algebra

$$
\begin{array}{rlll}
ce ::= & sde & | & \\
& ce_1 \; ; \; ce_2 & | & \textit{Sequence} \\
& ce_1 \; \vee \; ce_2 & | & \textit{Disjunction} \\
& ce^* & | & \textit{Iteration} \\
& \neg \; ce & | & \textit{Negation} \\
& \sigma_\theta(ce) & | & \textit{Selection} \\
& \pi_m(ce) & | & \textit{Projection} \\
& [ce]_{T_1}^{T_2} & & \textit{Windowing} \text{ (from } T_1 \text{ to } T_2\text{)}
\end{array}
$$

- *Sequence*: Two events following each other in time.

# A Simple Unifying Event Algebra

$$
\begin{array}{rlll}
ce ::= & sde & | & \\
& ce_1 \; ; \; ce_2 & | & \textit{Sequence} \\
& ce_1 \; \vee \; ce_2 & | & \textit{Disjunction} \\
& ce^* & | & \textit{Iteration} \\
& \neg \; ce & | & \textit{Negation} \\
& \sigma_\theta(ce) & | & \textit{Selection} \\
& \pi_m(ce) & | & \textit{Projection} \\
& [ce]_{T_1}^{T_2} & & \textit{Windowing} \text{ (from } T_1 \text{ to } T_2\text{)}
\end{array}
$$

- *Sequence*: Two events following each other in time.
- *Disjunction*: Either of two events occurring, regardless of temporal relations.

# A Simple Unifying Event Algebra

$$
\begin{array}{rlll}
ce ::= & sde & | & \\
& ce_1 \; ; \; ce_2 & | & \textit{Sequence} \\
& ce_1 \; \vee \; ce_2 & | & \textit{Disjunction} \\
& ce^* & | & \textit{Iteration} \\
& \neg \; ce & | & \textit{Negation} \\
& \sigma_\theta(ce) & | & \textit{Selection} \\
& \pi_m(ce) & | & \textit{Projection} \\
& [ce]_{T_1}^{T_2} & & \textit{Windowing} \text{ (from } T_1 \text{ to } T_2\text{)}
\end{array}
$$

- *Sequence*: Two events following each other in time.
- *Disjunction*: Either of two events occurring, regardless of temporal relations.
- The combination of *Sequence* and *Disjunction* expresses Conjunction (both events occurring).

# A Simple Unifying Event Algebra

$$
\begin{array}{rll}
ce ::= & sde & | \\
& ce_1 \; ; \; ce_2 & | \; \textit{Sequence} \\
& ce_1 \; \vee \; ce_2 & | \; \textit{Disjunction} \\
& ce^* & | \; \textit{Iteration} \\
& \neg \; ce & | \; \textit{Negation} \\
& \sigma_\theta(ce) & | \; \textit{Selection} \\
& \pi_m(ce) & | \; \textit{Projection} \\
& [ce]_{T_1}^{T_2} & \;\; \textit{Windowing} \text{ (from } T_1 \text{ to } T_2\text{)}
\end{array}
$$

- *Iteration*: An event occurring $N$ times in sequence, where $N \geq 0$. This operation is similar to the *Kleene star* operation in regular expressions, the difference being that *Kleene star* is unbounded.

# A Simple Unifying Event Algebra

$$
\begin{array}{rll}
ce ::= & sde & | \\
& ce_1 \; ; \; ce_2 & | \; \textit{Sequence} \\
& ce_1 \; \vee \; ce_2 & | \; \textit{Disjunction} \\
& ce^* & | \; \textit{Iteration} \\
& \neg \; ce & | \; \textit{Negation} \\
& \sigma_\theta(ce) & | \; \textit{Selection} \\
& \pi_m(ce) & | \; \textit{Projection} \\
& [ce]_{T_1}^{T_2} & \textit{Windowing} \text{ (from } T_1 \text{ to } T_2\text{)}
\end{array}
$$

- *Negation*: Absence of event occurrence.

# A Simple Unifying Event Algebra

$$
\begin{array}{rll}
ce ::= & sde & | \\
& ce_1 \; ; \; ce_2 & | \; \textit{Sequence} \\
& ce_1 \; \vee \; ce_2 & | \; \textit{Disjunction} \\
& ce^* & | \; \textit{Iteration} \\
& \neg \; ce & | \; \textit{Negation} \\
& \sigma_\theta(ce) & | \; \textit{Selection} \\
& \pi_m(ce) & | \; \textit{Projection} \\
& [ce]_{T_1}^{T_2} & \textit{Windowing} \text{ (from } T_1 \text{ to } T_2\text{)}
\end{array}
$$

- *Negation*: Absence of event occurrence.
- *Selection*: Select those events whose attributes satisfy a set of predicates/relations $\theta$, temporal or otherwise.

# A Simple Unifying Event Algebra

$$
\begin{array}{rlll}
ce ::= & sde & | & \\
& ce_1 \; ; \; ce_2 & | & \textit{Sequence} \\
& ce_1 \; \vee \; ce_2 & | & \textit{Disjunction} \\
& ce^* & | & \textit{Iteration} \\
& \neg \; ce & | & \textit{Negation} \\
& \sigma_\theta(ce) & | & \textit{Selection} \\
& \pi_m(ce) & | & \textit{Projection} \\
& [ce]_{T_1}^{T_2} & & \textit{Windowing} \text{ (from } T_1 \text{ to } T_2\text{)}
\end{array}
$$

- *Projection*: Return an event whose attribute values are a possibly transformed subset of the attribute values of its sub-events.

# A Simple Unifying Event Algebra

$$
\begin{array}{rlll}
ce ::= & sde & | & \\
& ce_1 \; ; \; ce_2 & | & \textit{Sequence} \\
& ce_1 \; \vee \; ce_2 & | & \textit{Disjunction} \\
& ce^* & | & \textit{Iteration} \\
& \neg \; ce & | & \textit{Negation} \\
& \sigma_\theta(ce) & | & \textit{Selection} \\
& \pi_m(ce) & | & \textit{Projection} \\
& [ce]_{T_1}^{T_2} & & \textit{Windowing} \text{ (from } T_1 \text{ to } T_2\text{)}
\end{array}
$$

- *Projection*: Return an event whose attribute values are a possibly transformed subset of the attribute values of its sub-events.
- *Windowing*: Evaluate the conditions of an event pattern within a specified time window.

# Types of Window

- Logical (time-based) windows: bounds are defined as a function of time.
  - Example: Match a pattern only on the events received in the last 10 minutes.
- Physical (count-based) windows: bounds depend on the number of items included in the window.
  - Example: Match a pattern only on the last 10 received events.

# Types of Window

Orthogonal classification based on the way bounds move:

- Fixed windows do not move!
  - Process the events received between 7/11/2012 and 12/6/2015.

# Types of Window

Orthogonal classification based on the way bounds move:

- Fixed windows do not move!
  - Process the events received between 7/11/2012 and 12/6/2015.
- Landmark windows have a fixed lower bound, while the upper bound advances every time a new event enters the system.
  - Process the events received since 12/6/2015.

# Types of Window

Orthogonal classification based on the way bounds move:

- Fixed windows do not move!
  - Process the events received between 7/11/2012 and 12/6/2015.
- Landmark windows have a fixed lower bound, while the upper bound advances every time a new event enters the system.
  - Process the events received since 12/6/2015.
- Sliding windows have a fixed size, i.e. both bounds advance with a pre-defined logical or physical step.
  - Process the last 10 received events.
  - Process the events received in the last 10'.

# Types of Window

Orthogonal classification based on the way bounds move:

- Fixed windows do not move!
  - Process the events received between 7/11/2012 and 12/6/2015.
- Landmark windows have a fixed lower bound, while the upper bound advances every time a new event enters the system.
  - Process the events received since 12/6/2015.
- Sliding windows have a fixed size, i.e. both bounds advance with a pre-defined logical or physical step.
  - Process the last 10 received events.
  - Process the events received in the last 10'.
- Pane windows: overlapping sliding windows.
- Tumble windows: non-overlapping sliding windows.

# Processing Model

Selection strategies filter the set of matched patterns.

- Assume the pattern $\alpha;\beta$ and the stream $(\alpha, 1),\ (\alpha, 2),\ (\beta, 3)$.

# Processing Model

Selection strategies filter the set of matched patterns.

- Assume the pattern $\alpha ; \beta$ and the stream $(\alpha, 1),\ (\alpha, 2),\ (\beta, 3)$.
- The multiple selection strategy produces $(\alpha, 1),\ (\beta, 3)$ and $(\alpha, 2),\ (\beta, 3)$.

# Processing Model

Selection strategies filter the set of matched patterns.

- Assume the pattern $\alpha;\beta$ and the stream $(\alpha, 1),\ (\alpha, 2),\ (\beta, 3)$.
- The multiple selection strategy produces $(\alpha, 1),\ (\beta, 3)$ and $(\alpha, 2),\ (\beta, 3)$.
- The single selection strategy produces either $(\alpha, 1),\ (\beta, 3)$ or $(\alpha, 2),\ (\beta, 3)$.
- The single selection strategy represents a family of strategies, depending on the matches actually chosen among all possible ones.

# Processing Model

Consumption policies place constraints on the use of events.

- Assume the pattern $\alpha; \beta$ and the stream $(\alpha, 1),\ (\beta, 2),\ (\beta, 3)$.

# Processing Model

Consumption policies place constraints on the use of events.

- Assume the pattern $\alpha; \beta$ and the stream $(\alpha, 1),\ (\beta, 2),\ (\beta, 3)$.
- The zero consumption policy produces $(\alpha, 1),\ (\beta, 2)$ and $(\alpha, 1),\ (\beta, 3)$.
  - ... assuming a multiple selection strategy.

# Processing Model

Consumption policies place constraints on the use of events.

- Assume the pattern $\alpha;\beta$ and the stream $(\alpha, 1),\ (\beta, 2),\ (\beta, 3)$.
- The zero consumption policy produces $(\alpha, 1),\ (\beta, 2)$ and $(\alpha, 1),\ (\beta, 3)$.
  - ... assuming a multiple selection strategy.
- The selected consumption policy produces $(\alpha, 1),\ (\beta, 2)$.
  - $(\alpha, 1)$ is consumed when the pattern is matched (at the arrival of $(\beta, 2)$), and thus no longer available when $(\beta, 3)$ arrives.
  - Once $(\alpha, 1)$ is consumed, it is not considered in ANY other pattern!

# Literature/Sources


Surveys:

- G. Cugola, A. Margara: Processing flows of information: From data stream to complex event processing. ACM Comput. Surv. 44(3): 15:1-15:62, 2012.
- E. Alevizos et al: Probabilistic Complex Event Recognition: A Survey. ACM Comput. Surv. 50(5): 71:1-71:31, 2017.
- M. Dayarathna, S. Perera: Recent Advancements in Event Processing. ACM Comput. Surv. 51(2): 33:1-33:36, 2018.
- N. Giatrakos et al: Complex Event Recognition in the Big Data Era: A Survey. The VLDB Journal, 29, pp. 313–352, 2020.

Books:

- D. C. Luckham: The power of events—an introduction to complex event processing in distributed enterprise systems. ACM 2005, ISBN 978-0-201-72789-0.
- O. Etzion, P. Niblett: Event Processing in Action. Manning Publications Company 2010, ISBN 978-1-935182-21-4.

Conferences/Proceedings:

- International Conference on Distributed and Event-Based Systems (DEBS). (2021), (2020), (2019), ...
- SIGMOD, VLDB, ICDT, PODS, EDBT, IJCAI, ECAI.

# Literature/Sources


Foundations:

- A. Grez et al: A Formal Framework for Complex Event Processing. ICDT 2019: 5:1-5:18.
- A. Grez et al: On the Expressiveness of Languages for Complex Event Recognition. ICDT 2020: 15:1-15:17.
- A. Artikis et al: Dagstuhl Seminar on the Foundations of Composite Event Recognition. SIGMOD Rec. 49(4): 24-27, 2020.

Applications:

- N. P. Schultz-Møller et al: Distributed complex event processing with query rewriting. DEBS 2009.
- A. Artikis et al: Heterogeneous Stream Processing and Crowdsourcing for Urban Traffic Management. EDBT 2014: 712-723.
- M. Pitsikalis et al: Composite Event Recognition for Maritime Monitoring. DEBS 2019, pp. 163–174, 2019.

Public datasets:

- DEBS challenges.
- Maritime situational awareness.
- Human activity recognition.