Event processing under uncertainty

Tutorial

Alexander Artikis¹, Opher Etzion², Zohar Feldman², & Fabiana Fournier² ³

DEBS 2012
July 16, 2012

¹ NCSR Demokritos, Greece
² IBM Haifa Research Lab, Israel
³ We acknowledge the help of Jason Filippou and Anastasios Skarlatidis
Event uncertainty from a petroleum producer point of view

The Two Sides of the Uncertainty

From a Petroleum Producer Point of View:

US$ 18 / bbl
(Expected Value)

Investment: Governed quantitatively by the “Bad News Principle” (fear)
Abandon: Governed quantitatively by the “Good News Principle” (hope)
# Outline

<table>
<thead>
<tr>
<th>Topic</th>
<th>Time</th>
<th>Speaker</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Part I:</strong> Introduction and illustrative example</td>
<td>30 minutes</td>
<td>Opher Etzion</td>
</tr>
<tr>
<td><strong>Part II:</strong> Uncertainty representation</td>
<td>15 minutes</td>
<td>Opher Etzion</td>
</tr>
<tr>
<td><strong>Part III:</strong> Extending event processing to handle uncertainty</td>
<td>45 minutes</td>
<td>Opher Etzion</td>
</tr>
<tr>
<td>Break</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Part IV:</strong> AI based models for Event Recognition under Uncertainty</td>
<td>80 minutes</td>
<td>Alexander Artikis</td>
</tr>
<tr>
<td><strong>Part V:</strong> Open issues and summary</td>
<td>10 minutes</td>
<td>Opher Etzion</td>
</tr>
</tbody>
</table>
Part I: Introduction and illustrative example
By 2015, 80% of all available data will be uncertain.

By 2015 the number of networked devices will be double the entire global population. All sensor data has uncertainty.

The total number of social media accounts exceeds the entire global population. This data is highly uncertain in both its expression and content.

Data quality solutions exist for enterprise data like customer, product, and address data, but this is only a fraction of the total enterprise data.
The big data perspective

Big data is one of the three technology trends at the leading edge a CEO cannot afford to overlook in 2012 [Gartner #228708, January 2012]

Dimensions of big data

- **Velocity**
  - (data in motion)

- **Volume**
  - (data processed in RT)

- **Variety**
  - (data in many forms)

- **Veracity**
  - (data in doubt)
The big data perspective in focus

Big data is one of the three technology trends at the leading edge a CEO cannot afford to overlook in 2012 [Gartner #228708, January 2012]

Dimensions of big data

Velocity
(data processed in RT)

Volume
(data in motion)

Variety
(data in many forms)

Veracity
(data in doubt)
Uncertainty in event processing

State-of-the-art systems assume that data satisfies the “closed world assumption”, being complete and precise as a result of a cleansing process before the data is utilized.

Processing data is deterministic

The problem

In real applications events may be uncertain or have imprecise content for various reasons (missing data, inaccurate/noisy input; e.g. data from sensors or social media)

Often, in real time applications cleansing the data is not feasible due to time constraints

Online data is not leveraged for immediate operational decisions
Mishandling of uncertainty may result in undesirable outcomes

This tutorial presents common types of uncertainty and different ways of handling uncertainty in event processing
Representative sources of uncertainty

- Thermometer
- Human error
- Malicious Source
- Source Malfunction
- Fake tweet
- Sensor disrupter
- Wrong hourly sales summary
- Projection of temporal anomalies
- Source Inaccuracy
- Rumor
- Sampling or approximation
- Inaccuracy
- Propagation of uncertainty
- Inference based on uncertain value
Event processing

Event Sources → Runtime Engine → Detected Situations

Authoring Tool → Rules / Patterns

Build Time

Run Time

Actions

Situation Detection
Types of uncertainty in event processing

- incomplete event streams
- insufficient event dictionary
- erroneous event recognition
- inconsistent event annotation

Uncertainty in the event input, in the composite event pattern, in both...
Illustrative example – Surveillance system for crime detection

A visual surveillance system provides data to an event driven application.

The goal of the event processing application is to detect and alert in real-time possible occurrences of crimes (e.g. drug deal) based on the surveillance system and citizens’ reports inputs.

Real time alerts are posted to human security officers

1. Whenever the same person is observed participating in a potential crime scene more than 5 times in a certain day
2. Identification of locations in which more than 10 suspicious detections were observed in three consecutive days
3. Report of the three most suspicious locations in the last month
Some terminology

Event producer/consumer
Event type
Event stream
Event Processing Agent (EPA)
Event Channel (EC)
Event Processing Network (EPN)
Context (temporal, spatial and segmentation)
Crime observation

Uncertainty
Wrong identification
Missing events
Noisy pictures

Observation

......
Crime Indication
......

Filter

Assertion: *Crime indication* = ‘true’
Context: *Always*

Suspicious observation

......
Crime Indication
......
Identity
Mega Suspect detection

Uncertainty
Wrong identification
Missing events
Wrong threshold

Suspicious observation

| ...... |
| Crime Indication |
| ...... |
| Identity |

Threshold

Assertion: \( \text{Count (Suspicious observation)} > 5 \)
Context: \( \text{Daily per suspect} \)

Mega suspect

| ...... |
| Suspect |
| ...... |
| ...... |
Split Mega Suspect

**Uncertainty**
Wrong identification
Missing events
Wrong threshold

**Known criminal**
- Picture

**Discovered criminal**
- Picture

**Split**
Emit Known criminal when exists (criminal-record)
Emit Discovered criminal otherwise
Context: Always
Crime report matching

**Uncertainty**
- Missed events
- Wrong sequence
- Inaccurate location
- Inaccurate crime type

### Sequence

(Suspicious observation [override],
Crime report)

Context: *Crime type per location*

**Matched crime**

......
Crime Indication
......
Identity
Suspicious location detection

**Threshold**

**Assertion:** \( \text{Count (Suspicious observation)} > 10 \)

**Context:** Location, Sliding 3 day interval

**Uncertainty**

- Missed events
- Wrong timing
- Inaccurate location
Most frequent locations

**Uncertainty**
Derived from uncertainty in Suspicious location events

**Top-K**
Order by: count (Suspicious location)
Context: Month
Part II: Uncertainty representation
Occurrence uncertainty: Uncertainty whether an event has occurred

Example
Event type: Crime observation
Occurrence time: 7/7/12, 12:23
Certainty: 0.73

The event header has an attribute with label “certainty” with values in the range of [0,1]

In raw events:
Given by the source (e.g. sensor accuracy, level of confidence)

In derived events:
Calculated as part of the event derivation
Temporality uncertainty: Uncertainty when an event has occurred

**Example**

Event type: Crime observation
Occurrence time: \( U(7/7/12, 12:00, 7/7/12 \ 13:00) \)
Certainty: 0.9

The event header has an attribute with the label “occurrence time” which may designate: timestamp, interval, or distribution over an interval.

In raw events: Given by the source

In derived events: Calculated as part of the event derivation
The interval semantics

1. The event happens during the entire interval       Occurrence time type: interval

2. The event happens at some unknown time-point within the interval  Occurrence time type: timestamp, instance type: interval

3. The event happens at some time-point within an interval and the distribution of probabilities is known  Occurrence time type: timestamp, instance type: interval + distribution

Example: U(7/7/12, 12:00, 7/7/12 13:00)
Spatial uncertainty: uncertainty where an event has occurred

Example
Event type: Crime observation
Occurrence time: U(7/7/12, 12:00, 7/7/12 13:00)
Certainty: 0.9
Location: near 30 Market Street, Philadelphia

The event header has an attribute with the label “location” which may designate: point, poly-line, area and distribution of points within poly-line and area. Locations can be specified either in coordinates or in labels (e.g. addresses)

In raw events:
Given by the source

In derived events:
Calculated as part of the event derivation
The spatial semantics

A poly-line may designate a road, e.g. M2 in UK

An area may designate a bounded geographical area, e.g. the city of Princeton.

Like temporal interval – the designation of a poly-line or area may be interpreted as: the event happens in the entire area, the event happens at unknown point within an area, the event happens within the area with some distribution of points

Examples:

Location type: area, Location := Princeton

Location type: point Location := M2

Location type: point Location := U (center = city-hall Princeton, radius = 2KM)
Uncertainty about the event content – attribute values (meta data)

Meta-data definition: attribute values may be probabilistic

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Probabilistic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event Name</td>
<td>String</td>
<td>No</td>
</tr>
<tr>
<td>Occurrence Time</td>
<td>Time-stamp</td>
<td>Possibly</td>
</tr>
<tr>
<td>Detection Time</td>
<td>Time-stamp</td>
<td>No</td>
</tr>
<tr>
<td>Certainty</td>
<td>Double (0,1)</td>
<td>No</td>
</tr>
<tr>
<td>Location</td>
<td>Array[2]</td>
<td>Possibly</td>
</tr>
<tr>
<td>Observed-id</td>
<td>Integer</td>
<td>Possibly</td>
</tr>
<tr>
<td>Picture</td>
<td>String</td>
<td>No</td>
</tr>
<tr>
<td>Criminal-id</td>
<td>String</td>
<td>Possibly</td>
</tr>
<tr>
<td>Crime indication</td>
<td>Boolean</td>
<td>Possibly</td>
</tr>
</tbody>
</table>
Uncertainty about the event content – attribute value example

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event Name</td>
<td>Observation</td>
</tr>
<tr>
<td>Occurrence Time</td>
<td>$U(8AM, 9AM)$</td>
</tr>
<tr>
<td>Detection Time</td>
<td>9AM</td>
</tr>
<tr>
<td>Certainty</td>
<td>0.85</td>
</tr>
<tr>
<td>Location</td>
<td>(349845.2, 7654345.4)</td>
</tr>
<tr>
<td>Observed-id</td>
<td>5</td>
</tr>
<tr>
<td>Picture</td>
<td>NA</td>
</tr>
<tr>
<td>Criminal-id</td>
<td>“Jon Doe, 12345678”</td>
</tr>
<tr>
<td>Crime indication</td>
<td>P(“true”, 0.6, “false”, 0.4)</td>
</tr>
</tbody>
</table>

Instance:
“Crime indication” has explicit distribution
Uncertainty in situation detection

Threshold

Assertion: Count (Suspicious observation) > 5
Context: Daily per suspect

Recall the “Mega suspect detection”

Threshold

Assertion: Count (Suspicious observation)
> 5 with probability = 0.9
= 5 with probability = 0.45
= 4 with probability = 0.2
Context: Daily per suspect

The “Mega suspect detection” can have some probability, furthermore there can be some variations (due to possible missing event) when count = 5, count = 4, etc… Thus it may become a collection of assertions with probabilities
Part III: Extending event processing to handle uncertainty
Uncertainty handling

Traditional event processing needs to be enhanced to account for uncertain events

Two main handling methods:

Uncertainty propagation
The uncertainty of input events is propagated to the derived events

Uncertainty flattening
Uncertain values are replaced with deterministic equivalents; events may be ignored.
Topics covered

1. Expressions*
2. Filtering
3. Split
4. Context assignment
5. Pattern matching examples:
   a. Top K
   b. Split

* Note that transformation and aggregations are covered by expressions.
Expressions (1/4)

Expressions are used extensively in event-processing in

*Derivation*: assigning values to derived events

*Assertions*: filter and pattern detection

Expressions comprise a combination of

**Operators**

- Arithmetic functions, e.g. `sum`, `min`
- Logical operators, e.g. `>`, `=`
- Textual operators, e.g. concatenation

**Operands** *(terms)* referencing

- particular event attributes
- general constants
Expressions (2/4)

**Example:** Summing the number of suspicious observations in two locations

\[ \text{location1.num-observations} + \text{location2.num-observations} \]

**Deterministic case:**

\[
12 + 6 = 18
\]

**Stochastic case:**

\[
\begin{array}{c}
11 & 12 & 13 \\
\hline
\end{array} + \begin{array}{c}
4 & 5 & 6 & 7 \\
\hline
\end{array} = ?
\]
Expressions (3/4)

The ‘uncertainty propagation’ approach: Operators overloading

Explicit, e.g. \( \text{Normal}(1,1) + \text{Normal}(3,1) = \text{Normal}(4,2) \)

Numerical convolution (general distributions, probabilistic dependencies)

Monte-Carlo methods: generate samples, evaluate the result and aggregate

The ‘uncertainty flattening’ approach: Replacing the original expression using a predefined policy

Some examples

- Expectation: \( X+Y \) ➞ Expectation\((X+Y)\)
- Confidence value: \( X+Y \) ➞ Percentile\((X+Y,0.9)\)
- \( X>5 \) ➞ ‘true’ if Prob\(\{X>5\}>0.9\); ‘false’ otherwise
## Expressions (4/4)

### Probabilistic operators

<table>
<thead>
<tr>
<th>Function</th>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>expectation (X)</td>
<td>(X)-Stochastic term</td>
<td>The expectation of the stochastic term (X)</td>
</tr>
<tr>
<td>variance (X)</td>
<td>(X)-Stochastic term</td>
<td>The variance of the stochastic term (X)</td>
</tr>
<tr>
<td>(pdf) ((X, x))</td>
<td>(X)-Stochastic Term, (x)-value</td>
<td>The probability of (X) taking value (x)</td>
</tr>
<tr>
<td>(cdf) ((X, x))</td>
<td>(X)-Stochastic Term, (x)-numeric value</td>
<td>The probability that (X) takes a value not larger than (x)</td>
</tr>
<tr>
<td>(percentile) ((X, \alpha))</td>
<td>(X)-Stochastic Term, (\alpha)-numeric value</td>
<td>The smallest number (z) such that (cdf(X, z) \geq \alpha)</td>
</tr>
<tr>
<td>(frequent) ((X))</td>
<td>(X)-Stochastic Term</td>
<td>A value (x) that maximizes (pdf(X, x))</td>
</tr>
</tbody>
</table>
Should the ‘Observation’ event instance pass the filter?

<table>
<thead>
<tr>
<th>Observation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Certainty</td>
</tr>
<tr>
<td>Crime-Indication</td>
</tr>
</tbody>
</table>

Crime observation

Assertion: crime-indication = ‘true’
Context: Always
The ‘uncertainty propagation’ approach

observation.certainty * \text{Prob}\{\text{assertion}\}
The ‘uncertainty flattening’ approach

Assertion: crime-indication = ‘true’

Context: Always

observation.certainty * Prob{assertion} > 0.5
Split (1/3)

Split Payments

Sender

PayPal

Recipient

Recipient

Known criminal

Certainty 0.75

Id ['John Doe1', 0.3; NA, 0.7]

Discovered criminal

Certainty 0.75

Id ['John Doe1', 0.3; NA, 0.7]

Mega suspect

Certainty 0.75

Id ['John Doe1', 0.3; NA, 0.7]

Split mega-suspect

Pattern: Split

Context: Always

Id is null

Id is not null
Split (2/3)

The ‘uncertainty propagation’ approach

<table>
<thead>
<tr>
<th>Mega suspect</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Certainty</strong></td>
</tr>
<tr>
<td><strong>Id</strong></td>
</tr>
</tbody>
</table>

**Split mega-suspect**

Pattern: Split
Context: Always

**Known criminal**

<table>
<thead>
<tr>
<th>Certainty</th>
<th>0.75*0.3</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Id</strong></td>
<td>‘John Doe1’</td>
</tr>
</tbody>
</table>

**Discovered criminal**

<table>
<thead>
<tr>
<th>Certainty</th>
<th>0.75*0.7</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Id</strong></td>
<td>NA</td>
</tr>
</tbody>
</table>
The ‘uncertainty flattening’ approach

\[ Id \rightarrow \text{frequent}(Id) \]
Context assignment – temporal
Example 1 (1/3)

Should the ‘Suspicious observation’ event be assigned to the context segment ‘July 16, 2012, 10AM-11AM’?
The ‘uncertainty propagation’ approach

\[
\text{Prob}\{\text{occurrence time } \in [10AM, 11AM]\} \times \text{certainty}
\]

**Mega suspect detection**

Context segment: ‘John Doe’,

‘July 16, 2012, 10AM-11AM’
Context assignment – temporal Example 1 (3/3)

The ‘uncertainty flattening’ approach

\[ \text{Prob}\{\text{occurrence time } \in [10AM, 11AM]\} > 0.5 \]

Mega suspect detection

Context assignment – temporal
Example 2

Should the *uncertain* ‘Suspicious observation’ event initiate a new context segment? Participate in an existing context segmentation?

```
Suspicious Observation
Certainty: 0.3
Occurrence time: Uni(9:45AM, 10:05AM)
Id: ‘John Doe’

Suspicious observation detection
Context: ‘John Doe’
State: 1 instance

Suspicious observation detection
Context: ‘John Doe’
```
Which of the context segmentations should the ‘Suspicious observation’ event be assigned to?

**Mega suspect detection**

Context segment: ‘John Doe1’,

‘July 16, 2012, 10AM-11AM’

**Mega suspect detection**

Context segment: ‘John Doe2’,

‘July 16, 2012, 10AM-11AM’
The ‘uncertainty propagation’ approach

observation.certainty \cdot \text{Prob}\{\text{‘John Doe1’}\}

Mega suspect detection
Context segment: ‘John Doe1’,
‘July 16, 2012, 10AM-11AM’

observation.certainty \cdot \text{Prob}\{\text{‘John Doe2’}\}

Mega suspect detection
Context segment: ‘John Doe2’,
‘July 16, 2012, 10AM-11AM’
Context assignment – Segmentation-oriented (3/3)

The ‘uncertainty flattening’ approach

Associate event to context segmentation frequent(Id)

Mega suspect detection
Context segment: ‘John Doe1’,
‘July 16, 2012, 10AM-11AM’

Mega suspect detection
Context segment: ‘John Doe2’,
‘July 16, 2012, 10AM-11AM’
Pattern matching: Top-K (1/3)
(aka event recognition)

What is the location with the most crime indications?

Most frequent locations
Pattern: Top-K (K=1)
Order by: count (suspicious location)
Context: Month
The ‘uncertainty propagation’ approach

Most frequent locations

Pattern: Top-K (K=1)
Order by: count (Suspicious location)
Context: Month

\[
\begin{align*}
\text{Count}_1 &= 11, 12, 13 \\
\text{Count}_2 &= 10, 11, 12, 13, 14 \\
\end{align*}
\]

\[
\begin{align*}
\text{Order} &= (\text{‘downtown’, ‘suburbs’}) \text{ w.p. Prob}\{\text{count}_1 > \text{count}_2\} \\
&\quad (\text{‘downtown’, ‘suburbs’}) \text{ w.p. Prob}\{\text{count}_1 > \text{count}_2\} \\
\end{align*}
\]
The ‘uncertainty flattening’ approach

**Most frequent locations**

Pattern: Top-K (K=1)
Order by: \( f(\text{count(suspicious location)}) \)
Context: Month

e.g. expectation, cdf, …
Pattern matching: Sequence (1/3)

Should the following events instances be matched as a sequence?

**Suspicious Observation**
- Certainty: 0.8
- Occurrence time: Uni(9:45AM, 10:05AM)
- Id: ‘John Doe’

**Crime report matching**
- Pattern: Sequence [Suspicious observation, Crime report]
- Context: Location, Crime type

**Crime report**
- Certainty: 0.9
- Occurrence time: 10:02AM
- Id: NA
Pattern matching: Sequence (2/3)

The ‘uncertainty propagation’ approach

**Suspicious Observation**
- **Certainty**: 0.8
- **Occurrence time**: Uni(9:45AM, 10:05AM)
- **Id**: ‘John Doe’

**Crime report matching**
- **Pattern**: Sequence
- **Context**: Location, Crime type

**Crime report**
- **Certainty**: 0.9
- **Occurrence time**: 10:02AM
- **Id**: NA

**Matched crime**
- **Certainty**: 0.612
- **Occurrence time**: Uni(9:45AM, 10:02AM)
- **Id**: ‘John Doe’

\[
\text{obs.certainty} \cdot \text{crime.certainty} \cdot \text{Prob}\{\text{obs.time}<\text{crime.time}\}
\]
Pattern matching: Sequence (3/3)

The ‘uncertainty flattening’ approach

Suspicious Observation

<table>
<thead>
<tr>
<th>Certainty</th>
<th>Occurrence time</th>
<th>Id</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.8</td>
<td>Uni(9:45AM,10:05AM)</td>
<td>‘John Doe’</td>
</tr>
</tbody>
</table>

......

......

Crime report

<table>
<thead>
<tr>
<th>Certainty</th>
<th>Occurrence time</th>
<th>Id</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.9</td>
<td>10:02AM</td>
<td>NA</td>
</tr>
</tbody>
</table>

......

......

Matched crime

<table>
<thead>
<tr>
<th>Certainty</th>
<th>Occurrence time</th>
<th>Id</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.72</td>
<td>9:55AM</td>
<td>‘John Doe’</td>
</tr>
</tbody>
</table>

......

......

Crime report matching

Pattern: Sequence

Context: Location, Crime type

Occurrence time → percentile(occurrence time, 0.5)
Part IV: Event Recognition under Uncertainty in Artificial Intelligence
Event Recognition under Uncertainty in Artificial Intelligence

AI-based systems already deal with various types of uncertainty:
- Erroneous input data detection.
- Imperfect composite event definitions.

Overview:
- Logic programming.
- Markov logic networks.
Human Recognition from Video: Bilattice-based Reasoning

Aim:
▶ Recognise humans given input data from part-based classifiers operating on video.

Uncertainty:
▶ Erroneous input data detection.
▶ Imperfect composite event definitions.

Approach:
▶ Extend logic programming by incorporating a bilattice:
  ▶ Consider both supportive and contradicting information about a given human hypothesis.
  ▶ For every human hypothesis, provide a list of justifications (proofs) that support or contradict the hypothesis.
Human Recognition from Video
Human Recognition from Video

- Evidence FOR hypothesis “Entity1 is a human”:
  - Head visible.
  - Torso visible.
  - Scene is consistent at 1’s position.
Evidence AGAINST hypothesis “Entity1 is a human”:
- Legs occluded.
- However, the missing legs can be explained because of the occlusion by the image boundary.
- If occluded body parts can be explained by a rule, the hypothesis is further supported.
Evidence FOR hypothesis “Entity A is a human”:
- A is on the ground plane.
- Scene is consistent at A’s position.
Evidence AGAINST hypothesis “Entity A is a human”:
  ▶ No head detected.
  ▶ No torso detected.
  ▶ No legs detected.

Evidence against the hypothesis is overwhelming, so the system infers that A is unlikely to be a human.
Human Recognition from Video

- Recognising humans given three different sources of information, supporting or contradicting human hypotheses:

  \[
  \text{human}(X, Y, S) \leftarrow \text{head}(X, Y, S) \\
  \text{human}(X, Y, S) \leftarrow \text{torso}(X, Y, S) \\
  \neg\text{human}(X, Y, S) \leftarrow \neg\text{scene\_consistent}(X, Y, S)
  \]

- Supporting information sources are provided by head and torso detectors.

- Contradicting information is provided by the violation of geometric constraints (scene\_consistent).
We are able to encode information for and against:
- Input data (output of part-based detectors)
- Rules (typically they are less-than-perfect)
- Hypotheses (inferrable atoms)

We do this by defining a *truth assignment* function:

\[ \varphi : L \rightarrow B \]

where \( L \) is a declarative language and \( B \) is a bilattice.

Some points in the bilattice:
- \( < 0, 0 > \) agnostic
- \( < 1, 0 > \) absolute certainty about truth
- \( < 0, 1 > \) absolute certainty about falsity
- \( < 1, 1 > \) complete contradiction
A perfect part-based detector would detect body parts and geometrical constraints with absolute certainty.

\[
\begin{align*}
  head(25, 95, 0.9) \\
  torso(25, 95, 0.9) \\
  \neg scene\_consistent(25, 95, 0.9)
\end{align*}
\]
However, in realistic applications, every low-level information is detected with a degree of certainty, usually normalized to a probability.

\[
\varphi(\text{head}(25, 95, 0.9)) = \langle 0.90, 0.10 \rangle \\
\varphi(\text{torso}(25, 95, 0.9)) = \langle 0.70, 0.30 \rangle \\
\varphi(\neg \text{scene\_consistent}(25, 95, 0.9)) = \langle 0.80, 0.20 \rangle 
\]

For simple detectors, if the probability of the detection is \( p \), the following holds:

\[
\varphi(\text{detection}) = \langle p, 1-p \rangle 
\]
Traditional logic programming rules are binary:

\[
\begin{align*}
\text{human}(X, Y, S) & \leftarrow \text{head}(X, Y, S) \\
\text{human}(X, Y, S) & \leftarrow \text{torso}(X, Y, S) \\
\neg \text{human}(X, Y, S) & \leftarrow \neg \text{scene}\_\text{consistent}(X, Y, S)
\end{align*}
\]
But in a realistic setting, we’d like to have a measure of their reliability:

\[
\varphi(human(X, Y, S) \leftarrow head(X, Y, S)) = <0.40, 0.60>
\]
\[
\varphi(human(X, Y, S) \leftarrow torso(X, Y, S)) = <0.30, 0.70>
\]
\[
\varphi(\neg human(X, Y, S) \leftarrow \neg scene\_consistent(X, Y, S)) = <0.90, 0.10>
\]

Rules are specified manually.

There exists an elementary weight learning technique: The value \(x\) in \(<x, y>\) is the fraction of the times that the head of a rule is true when the body is true.
Human Recognition from Video: Uncertainty Handling

- It is now possible to compute the belief for and against a given human hypothesis, by aggregating through all the possible rules and input data.
- This is done by computing the closure over $\phi$ of multiple sources of information.
- E.g. if we have a knowledge base $S$ with only one rule and one fact that together entail $human$

$$S = \{ human(X, Y, S) \leftarrow head(X, Y, S), \ head(25, 95, 0.9) \}$$

the degree of belief for and against the human hypothesis would be:

$$cl(\phi)(human(X, Y, S)) = \bigwedge_{p \in S} cl(\phi)(p) =$$

$$cl(\phi)(human(X, Y, S) \leftarrow head(X, Y, S)) \land$$

$$cl(\phi)(head(25, 95, 0.9)) = \cdots = \langle x_h, y_h \rangle$$
Because we have a variety of information sources to entail a hypothesis from (or the negation of a hypothesis), the inference procedure needs to take into account all possible information sources.

The operator $\oplus$ is used for this.

The final form of the equation that computes the closure over the truth assignment of a hypothesis $q$ is the following:

$$cl(\varphi)(q) = \bigoplus_{S|q} \left[ \bigwedge_{p \in S} cl(\varphi)(p) \right] \oplus \neg \bigoplus_{S|\neg q} \left[ \bigwedge_{p \in S} cl(\varphi)(p) \right]$$

- Aggregating supporting information
- Negating contradicting information
Human Recognition from Video: Uncertainty Handling

- Given this uncertain input
  \[ \varphi(\text{head}(25, 95, 0.9)) = <0.90, 0.10> \]
  \[ \varphi(\text{torso}(25, 95, 0.9)) = <0.70, 0.30> \]
  \[ \varphi(\neg \text{scene\_consistent}(25, 95, 0.9)) = <0.80, 0.20> \]

- and these uncertain rules
  \[ \varphi(\text{human}(X, Y, S) \leftarrow \text{head}(X, Y, S)) = <0.40, 0.60> \]
  \[ \varphi(\text{human}(X, Y, S) \leftarrow \text{torso}(X, Y, S)) = <0.30, 0.70> \]
  \[ \varphi(\neg \text{human}(X, Y, S) \leftarrow \neg \text{scene\_consistent}(X, Y, S)) = <0.90, 0.10> \]

- we would like to calculate our degrees of belief for and against the hypothesis \text{human}(25, 95, 0.9).
**Human Recognition from Video: Uncertainty Handling**

\[
cl(\varphi)(\text{human}(25, 95, 0.9)) = \left[ < 0.4, 0.6 > \land < 0.9, 0.1 > \right] \oplus \\
\left[ < 0.3, 0.7 > \land < 0.7, 0.3 > \right] \oplus \neg \left[ < 0.9, 0.1 > \land < 0.8, 0.2 > \right] = \\
< 0.36, 0 > \oplus < 0.21, 0 > \oplus \neg < 0.72, 0 > = \\
< 0.4944, 0 > \oplus < 0, 0.72 > = < 0.4944, 0.72 >
\]

- Evidence against the hypothesis exceeds evidence for the hypothesis.
- Justifications for the hypothesis:
  - \( \varphi(\text{head}(25, 95, 0.9)) = < 0.90, 0.10 > \)
  - \( \varphi(\text{torso}(25, 95, 0.9)) = < 0.70, 0.30 > \)
- Justifications against the hypothesis:
  - \( \varphi(\neg \text{scene\_consistent}(25, 95, 0.9)) = < 0.80, 0.20 > \)
Human Recognition from Video: Summary

Uncertainty:
▶ Erroneous input data detection.
▶ Imperfect composite event definitions.

Features:
▶ Consider both supportive and contradicting information about a given hypothesis.
▶ For every hypothesis, provide a list of justifications (proofs) that support or contradict the hypothesis.

Note:
▶ Human detection is not a typical event recognition problem.
Public Space Surveillance: VidMAP

Aim:
- Continuously monitor an area and report suspicious activity.

Uncertainty:
- Erroneous input data detection.

Approach:
- Multi-threaded layered system combining computer vision and logic programming.
VidMAP: Architecture

High-Level Module
Standard logic programming reasoning

Mid-Level Module
Uncertainty elimination

Low-Level Module
Background Subtraction, Tracking and Appearance Matching on video content
VidMAP: Architecture

Monitored Area

Cameras (1 for each connected camera)

Camera Threads

Prolog Knowledge Base

Reasoning Thread (invoked every 5 seconds)

Complex Events

human(obj_0_2).
object(obj_0_5).
dropoff(obj_0_2, obj_0_5, 15800).

theft(obj_0_1, obj_0_5, 14740).
entry_violation(obj_0_2, 1600).
Filter out noisy observations (false alarms) by first determining whether the observation corresponds to people or objects that have been *persistently* tracked.

A tracked object is considered to be a human if:
- It is ‘tall’.
- It has exhibited some movement in the past.

A tracked object is considered to be a ‘package’ if:
- It does not move on its own.
- At some point in time, it was attached to a human.
The following rule defines theft:

\[
\text{theft}(H, \text{Obj}, T) \leftarrow \\
\text{human}(H), \\
\text{package}(\text{Obj}), \\
\text{possess}(H, \text{Obj}, T), \\
\text{not belongs}(\text{Obj}, H, T)
\]

- A human possesses an object if he carries it.
- An object belongs to a human if he was seen possessing it before anyone else.
The following rule defines entry violation:

\[
\text{entry\_violation}(H) \leftarrow \\
\quad \text{human}(H), \\
\quad \text{appear}(H, \text{scene}, T1), \\
\quad \text{enter}(H, \text{building\_door}, T2), \\
\quad \text{not privileged\_to\_enter}(H, T1, T2)
\]

- ‘building\_door’ and ‘scene’ correspond to video areas that can be hard-coded by the user at system start-up.
- An individual is privileged to enter a building if he swipes his ID card in the ‘building\_door’ area of ‘scene’ or if he is escorted into the building by a friend who swipes his ID card.
VidMAP: Summary

Uncertainty:
▶ Erroneous input data detection.

Features:
▶ Uncertainty elimination.
▶ Intuitive composite event definitions that are easy to be understood by domain experts.

Note:
▶ Crude temporal representation.
Probabilistic Logic Programming: Event Calculus

Aim:
- General-purpose event recognition system.

Uncertainty:
- Erroneous input data detection.

Approach:
- Express the Event Calculus in ProbLog.
The Event Calculus (EC)

- A Logic Programming language for representing and reasoning about events and their effects.

- Key Components:
  - event (typically instantaneous)
  - fluent: a property that may have different values at different points in time.

- Built-in representation of inertia:
  - $F = V$ holds at a particular time-point if $F = V$ has been *initiated* by an event at some earlier time-point, and not *terminated* by another event in the meantime.
ProbLog

- A Probabilistic Logic Programming language.
- Allows for independent ‘probabilistic facts’ $\text{prob}::\text{fact}$.
- $\text{Prob}$ indicates the probability that $\text{fact}$ is part of a possible world.
- Rules are written as in classic Prolog.
- The probability of a query $q$ imposed on a ProbLog database (success probability) is computed by the following formula:

\[
P_s(q) = P( \bigvee_{e \in \text{Proofs}(q)} \bigwedge_{f_i \in e} f_i )
\]
### ProbLog-EC: Input & Output

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>340 0.45 :: <em>inactive</em>(id&lt;sub&gt;0&lt;/sub&gt;)</td>
<td>340 0.41 :: <em>leaving_object</em>(id&lt;sub&gt;1&lt;/sub&gt;, id&lt;sub&gt;0&lt;/sub&gt;)</td>
</tr>
<tr>
<td>340 0.80 :: p(id&lt;sub&gt;0&lt;/sub&gt;) = (20.88, −11.90)</td>
<td>340 0.55 :: <em>moving</em>(id&lt;sub&gt;2&lt;/sub&gt;, id&lt;sub&gt;3&lt;/sub&gt;)</td>
</tr>
<tr>
<td>340 0.55 :: <em>appear</em>(id&lt;sub&gt;0&lt;/sub&gt;)</td>
<td></td>
</tr>
<tr>
<td>340 0.15 :: <em>walking</em>(id&lt;sub&gt;2&lt;/sub&gt;)</td>
<td></td>
</tr>
<tr>
<td>340 0.80 :: p(id&lt;sub&gt;2&lt;/sub&gt;) = (25.88, −19.80)</td>
<td></td>
</tr>
<tr>
<td>340 0.25 :: <em>active</em>(id&lt;sub&gt;1&lt;/sub&gt;)</td>
<td></td>
</tr>
<tr>
<td>340 0.66 :: p(id&lt;sub&gt;1&lt;/sub&gt;) = (20.88, −11.90)</td>
<td></td>
</tr>
<tr>
<td>340 0.70 :: <em>walking</em>(id&lt;sub&gt;3&lt;/sub&gt;)</td>
<td></td>
</tr>
<tr>
<td>340 0.46 :: p(id&lt;sub&gt;3&lt;/sub&gt;) = (24.78, −18.77)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>...</td>
</tr>
</tbody>
</table>
ProbLog-EC: Event Recognition
Uncertainty Elimination vs Uncertainty Propagation

Crisp-EC (uncertainty elimination) vs ProbLog-EC (uncertainty propagation):

- Crisp-EC: input data with probability \(< 0.5\) are discarded.
- ProbLog-EC: all input data are kept with their probabilities.
- ProbLog-EC: accept as recognised the composite events that have probability \(> 0.5\).
Crisp-EC: Uncertainty Elimination

The graph shows the relationship between SDE occurrences and noise (Gamma distribution mean). As the noise increases, the SDE occurrences decrease exponentially.
Uncertainty Elimination vs Uncertainty Propagation

![Graph showing F measure vs Gamma Distribution Mean for Crisp-EC and ProbLog-EC]

\[ \text{meeting}(ld_1, ld_2) = \text{true initiated if} \]
\[ \text{active}(ld_1) \text{ happens}, \]
\[ \text{close}(ld_1, ld_2) = \text{true holds}, \]
\[ \text{person}(ld_2) = \text{true holds}, \]
\[ \text{not running}(ld_2) \text{ happens} \]
Uncertainty Elimination vs Uncertainty Propagation

$\text{meeting}(ld_1, ld_2) = \text{true initiated}$ if

$\text{active}(ld_1)$ happens,

$\text{close}(ld_1, ld_2) = \text{true holds},$

$\text{person}(ld_2) = \text{true holds},$

not $\text{running}(ld_2)$ happens
Moving

\text{moving}(ld1, ld2) = \text{true} \text{ initiated iff}
\text{walking}(ld1) \text{ happens,}
\text{walking}(ld2) \text{ happens,}
\text{close}(ld1, ld2) = \text{true} \text{ holds,}
\text{orientation}(ld1) = O1 \text{ holds,}
\text{orientation}(ld2) = O2 \text{ holds,}
|O1 - O2| < threshold
Uncertainty Elimination vs Uncertainty Propagation

\[ \text{moving}(ld1, ld2) = \text{true initiated iff} \]
\[ \text{walking}(ld1) \text{ happens,} \]
\[ \text{walking}(ld2) \text{ happens,} \]
\[ \text{close}(ld1, ld2) = \text{true holds,} \]
\[ \text{orientation}(ld1) = O1 \text{ holds,} \]
\[ \text{orientation}(ld2) = O2 \text{ holds,} \]
\[ |O1 - O2| < \text{threshold} \]
Uncertainty Elimination vs Uncertainty Propagation

ProbLog-EC clearly outperforms Crisp-EC when:

▶ The environment is highly noisy (input data probability < 0.5) — realistic assumption in many domains,
▶ there are successive initiations that allow the composite event’s probability to increase and eventually exceed the specified (0.5) threshold, and
▶ the amount of probabilistic conjuncts in an initiation condition is limited.
ProbLog-EC: Summary

Uncertainty:
- Erroneous input data detection.

Features:
- Built-in rules for complex temporal representation.

Note:
- Independence assumption on input data is not always desirable.
Probabilistic Graphical Models

▶ E.g. Hidden Markov Models, Dynamic Bayesian Networks and Conditional Random Fields.
▶ Can naturally handle uncertainty:
  ▶ Erroneous input data detection.
  ▶ Imperfect composite event definitions.
▶ Limited representation capabilities.
▶ Composite events with complex relations create models with prohibitively large and complex structure.
▶ Various extensions to reduce the complexity.
▶ Lack of a formal representation language.
▶ Difficult to incorporate background knowledge.
Markov Logic Networks (MLN)

- Unify first-order logic with graphical models.
  - Compactly represent event relations.
  - Handle uncertainty.
- Syntactically: weighted first-order logic formulas \((F_i, w_i)\).
- Semantically: \((F_i, w_i)\) represents a probability distribution over possible worlds.

\[
P(\text{world}) \propto \exp \left( \sum (\text{weights of formulas it satisfies}) \right)
\]

A world violating formulas becomes less probable, but not impossible!
Event Recognition using MLN

Input (evidence)

SDE

Grounding

Markov Logic Networks

Markov Network
(probabilistic inference)

CE
Knowledge Base

Output
P(CE = True | SDE)

Recognised

CE
Markov Logic: Representation

\[
\text{suspect}(Id) \leftarrow \\
w_1 \quad \text{criminal\_record}(Id) \lor \\
\text{citizen\_report}(Id) \lor \\
\text{holds\_Dangerous\_Object}(Id)
\]

\[
\text{suspicious\_transaction}(Id_1) \leftarrow \\
w_2 \quad \text{suspect}(Id_1) \land \\
\text{mega\_suspect}(Id_2) \land \\
\text{deliver}(Id_1, Id_2)
\]
Markov Logic: Representation

- **Weight**: is a real-valued number.
- **Higher weight** $\rightarrow$ Stronger constraint.
- **Hard constraints**:
  - Must be satisfied in all possible worlds.
  - Infinite weight values.
  - Background knowledge.
- **Soft constraints**:
  - May not be satisfied in some possible worlds.
  - Strong weight values: almost always true.
  - Weak weight values: describe exceptions.
Sensors detect events with:

▶ Absolute certainty
▶ A degree of confidence

Example:

▶ \( holds\_Dangerous\_Object(Id) \) detected with probability 0.8.
▶ It can be represented with an auxiliary formula.
▶ The value of the weight of the formula is the log-odds of the probability.

\[ 1.38629 \quad obs\_holds\_Dangerous\_Object(Id) \Rightarrow holds\_Dangerous\_Object(Id) \]
Formulas are translated into clausal form.

Weights are divided equally among the clauses of a formula.

Given a set of constants from the input data, ground all clauses.

Ground predicates are Boolean nodes in the network.

Each ground clause:
  - Forms a clique in the network,
  - and is associated with $w_i$ and a Boolean feature.

\[
P(X = x) = \frac{1}{Z} \exp \left( \sum_i w_i n_i(x) \right)
\]

\[
Z = \sum_{x' \in \mathcal{X}} \exp \left( \sum_i w_i n_i(x') \right)
\]
Markov Logic: Network Construction

\[\text{suspect}(id) \iff \]

\[w_1 \quad \text{criminal}_\text{record}(id) \lor \]
\[\text{citizen}_\text{report}(id) \lor \]
\[\text{holds}_\text{Dangerous}_\text{Object}(id)\]

\[\text{suspicious}_\text{transaction}(id_1) \iff \]

\[w_2 \quad \text{suspect}(id_1) \land \]
\[\text{mega}_\text{suspect}(id_2) \land \]
\[\text{deliver}(id_1, id_2)\]

\[\frac{1}{3} w_1 \quad \neg \text{criminal}_\text{record}(id) \lor \text{suspect}(id)\]

\[\frac{1}{3} w_1 \quad \neg \text{citizen}_\text{report}(id) \lor \text{suspect}(id)\]

\[\frac{1}{3} w_1 \quad \neg \text{holds}_\text{Dangerous}_\text{Object}(id) \lor \text{suspect}(id)\]

\[w_2 \quad \neg \text{suspect}(id_1) \lor \neg \text{mega}_\text{suspect}(id_2) \lor \]
\[\neg \text{deliver}(id_1, id_2) \lor \text{suspicious}_\text{transaction}(id_1)\]
Markov Logic: Network Construction

For example, the clause:
\[ w_2 \neg suspect(id_1) \lor \neg mega\_suspect(id_2) \lor \neg deliver(id_1, id_2) \lor suspicious\_transaction(id_1) \]

produces the following groundings
\[ w_2 \neg suspect(alex) \lor \neg mega\_suspect(alex) \lor \neg deliver(alex, alex) \lor suspicious\_transaction(alex) \]
\[ w_2 \neg suspect(alex) \lor \neg mega\_suspect(nick) \lor \neg deliver(alex, nick) \lor suspicious\_transaction(alex) \]
\[ w_2 \neg suspect(nick) \lor \neg mega\_suspect(alex) \lor \neg deliver(nick, alex) \lor suspicious\_transaction(nick) \]
\[ w_2 \neg suspect(nick) \lor \neg mega\_suspect(nick) \lor \neg deliver(nick, nick) \lor suspicious\_transaction(nick) \]
Markov Logic: Network Construction

deliver (alex, alex)
deliver (nick, nick)
suspicious_transaction (nick)
suspect (nick)
mega_suspect (alex)

deliver (nick, alex)
suspicious_transaction (nick)
suspect (nick)
mega_suspect (alex)

deliver (alex, nick)

citizen_report (alex)
criminal_record (alex)
holds_Dangerous_Object (alex)

citizen_report (nick)
criminal_record (nick)
deliver (nick, nick)

holds_Dangerous_Object (nick)
criminal_record (nick)
citizen_report (nick)
Markov Logic: World state discrimination

State 1: \( P(X = x_1) = \frac{1}{Z} \exp \left( \frac{1}{3} w_1 \cdot 2 + \frac{1}{3} w_1 \cdot 2 + \frac{1}{3} w_1 \cdot 2 + w_2 \cdot 4 \right) = \frac{1}{Z} e^{2w_1+4w_2} \)
Markov Logic: World state discrimination

State 1: \( P(X = x_1) = \frac{1}{Z} \exp\left( \frac{1}{3} w_1 \cdot 2 + \frac{1}{3} w_1 \cdot 2 + \frac{1}{3} w_1 \cdot 2 + w_2 \cdot 4 \right) = \frac{1}{Z} e^{2w_1+4w_2} \)

State 2: \( P(X = x_2) = \frac{1}{Z} \exp\left( \frac{1}{3} w_1 \cdot 2 + \frac{1}{3} w_1 \cdot 2 + \frac{1}{3} w_1 \cdot 2 + w_2 \cdot 3 \right) = \frac{1}{Z} e^{2w_1+3w_2} \)
Markov Logic: Inference

- Event recognition: querying about composite events.
- Large network with complex structure.
- Exact inference is infeasible.
- MLN combine logical and probabilistic inference methods.
- Given some evidence $E = e$, there are two types of inference:
  1. Most Probable Explanation:
     - $\text{argmax}_q (P(Q = q|E = e))$
     - e.g. Maximum Satisfiability Solving.
  2. Marginal:
     - $P(Q = \text{true}|E = e)$
     - e.g. Markov Chain Monte Carlo sampling.
Training dataset: input data annotated with composite events.

<table>
<thead>
<tr>
<th>Input Data</th>
<th>Composite Events</th>
</tr>
</thead>
<tbody>
<tr>
<td>happensAt(\textit{active}(id_1), 101)</td>
<td>...</td>
</tr>
<tr>
<td>happensAt(\textit{walking}(id_2), 101)</td>
<td>\textbf{holdsAt}(\textit{meeting}(id_1, id_2), 101)</td>
</tr>
<tr>
<td>orientationMove(id_1, id_2, 101)</td>
<td>\textbf{holdsAt}(\textit{meeting}(id_2, id_1), 101)</td>
</tr>
<tr>
<td>¬close(id_1, id_2, 24, 101)</td>
<td>...</td>
</tr>
</tbody>
</table>

| ... | ... |
| happensAt(\textit{walking}(id_1), 200) | ... |
| happensAt(\textit{running}(id_2), 200) | ¬\textbf{holdsAt}(\textit{meeting}(id_1, id_2), 200) |
| ¬orientationMove(id_1, id_2, 200) | ¬\textbf{holdsAt}(\textit{meeting}(id_2, id_1), 200) |
| ¬close(id_1, id_2, 24, 200) | ... |
Markov Logic: Machine Learning

- Structure learning:
  - First-order logic formulas.
  - Based on Inductive Logic Programming techniques.

- Weight estimation:
  - Structure is known.
  - Find weight values that maximise the likelihood function.
  - Likelihood function: how well our model fits to the data.
  - Generative learning.
  - Discriminative learning.
Markov Logic: Discriminative Weight Learning

In event recognition we know a priori:

- Evidence variables: input data, eg simple events
- Query variables: composite events
- Recognise composite events given the input data.

Conditional log-likelihood function:

\[
\log P_w(Q = q \mid E = e) = \sum_i w_i n_i(e, q) - \log Z_e
\]

- Joint distribution of query Q variables given the evidence variables E.
- Conditioning on evidence reduces the likely states.
- Inference takes place on a simpler model.
- Can exploit information from long-range dependencies.
MLN-based Approaches

Most of the approaches:

- Knowledge base of domain-depended composite event definitions.
- Weighted definitions.
- Input uncertainty expressed using auxiliary formulas.
- Temporal constraints over successive time-points.

More expressive methods:

- Based on Allen’s interval logic.
- Event Calculus.
Event Calculus in MLN

Aim:
- General-purpose event recognition system.

Uncertainty:
- Imperfect composite event definitions.

Approach:
- Express the Event Calculus in Markov logic.
The Event Calculus

- A logic programming language for representing and reasoning about events and their effects.
  - Translation to Markov Logic is therefore straightforward.
- Key Components:
  - event (typically instantaneous)
  - fluent: a property that may have different values at different points in time.
- Built-in representation of inertia:
  - \( F = V \) holds at a particular time-point if \( F = V \) has been initiated by an event at some earlier time-point, and not terminated by another event in the meantime.
Event Calculus

Initiated at 3

Initiated at 10

Terminated at 20

Time
Event Calculus in MLN

Hard-constrained inertia rules:

2.3 \( CE \) \text{ initiatedAt } T \text{ if [Conditions]}

\[ \neg (CE \text{ holdsAt } T) \text{ iff } \neg (CE \text{ holdsAt } T-1), \neg (CE \text{ initiatedAt } T-1) \]

2.5 \( CE \) \text{ terminatedAt } T \text{ if [Conditions]}

\[ CE \text{ holdsAt } T-1, \neg (CE \text{ terminatedAt } T-1) \]
Event Calculus in MLN

Soft-constrained initiation inertia rules:

2.3 \( CE \) \text{ initiatedAt} \ T \text{ if [Conditions]}

2.5 \( CE \) \text{ terminatedAt} \ T \text{ if [Conditions]}

2.8 \( \neg(CE \text{ holdsAt} \ T) \text{ iff [Conditions]} \)

\( CE \text{ holdsAt} \ T \text{ iff [Conditions]} \)

\( CE \text{ holdsAt} \ T−1, \neg(CE \text{ terminatedAt} \ T−1) \)
Event Calculus in MLN

Soft-constrained termination inertia rules:

2.3 \( CE \) \( \text{initiatedAt} \ T \) if

[Conditions]

\( \neg(CE \ \text{holdsAt} \ T) \) iff

\( \neg(CE \ \text{holdsAt} \ T−1), \)

\( \neg(CE \ \text{initiatedAt} \ T−1) \)

2.5 \( CE \) \( \text{terminatedAt} \ T \) if

[Conditions]

2.8 \( CE \) \( \text{holdsAt} \ T \) iff

\( CE \ \text{holdsAt} \ T−1, \)

\( \neg(CE \ \text{terminatedAt} \ T−1) \)
Event Calculus in MLN

Soft-constrained termination inertia rules:

2.3 \( CE \) \text{ initiatedAt } T \text{ if}  
\[ \text{[Conditions]} \]
\[ \neg(CE \text{ holdsAt } T) \text{ iff} \]
\[ \neg(CE \text{ holdsAt } T-1), \]
\[ \neg(CE \text{ initiatedAt } T-1) \]

2.5 \( CE \) \text{ terminatedAt } T \text{ if}  
\[ \text{[Conditions]} \]

0.8 \( CE \) \text{ holdsAt } T \text{ iff}  
\( CE \text{ holdsAt } T-1, \)
\[ \neg(CE \text{ terminatedAt } T-1) \]
Crisp-EC (uncertainty elimination) is compared to various MLN-EC dialects (uncertainty propagation):

- MLN-EC_{HI}: hard-constrained inertia.
- MLN-EC_{SI(h)}: soft-constrained termination inertia rules.
- MLN-EC_{SI}: soft-constrained inertia.
Uncertainty Elimination vs Uncertainty Propagation

![Graph showing F-measure vs Threshold for EC_crisp, MLN-EC_HI, MLN-EC_SI, and MLN-EC_SI(h) methods.](image)
Event Calculus in MLN

Uncertainty:
- Imperfect composite event definitions.

Features:
- Customisable inertia behaviour to meet the requirements of the application under consideration.
- Weight learning is effective.

Note:
- Lack of explicit representation of intervals.
Part V: Open issues and summary
Open issues

Being a relative new area there exist many open issues related to events uncertainty both research and practical

In this tutorial, we showed a sample of several such open issues
Open issue: Assignment

How the assignment of uncertainty values is done?

Can people assign probabilities? Maybe fuzzy values?

Can probabilities be inferred statistically?
Open issue: real-time processing

Back to the “big data” applications: uncertainty is one dimension, scalability and real-time processing are other dimensions.

The handling of uncertain events in a scale and in real-time requires novel algorithmic approach.
Open issue: Forecasting

Applications employing derived events that are forecasted to occur in the (relatively near) future have inherent uncertainty associated with this forecasting.

The forecasting model should support various level of uncertainty (occurrence uncertainty, temporal uncertainty, location uncertainty) built in the forecasting model.
This tutorial dealt with various aspects of event uncertainty: motivation, representation, AI based techniques, and event processing handling techniques.

Uncertainty will become gradually part of event-based applications.