Run-Time Composite Event Recognition

Alexander Artikis, Marek Sergot and George Paliouras

Institute of Informatics & Telecommunications, NCSR “Demokritos”, Greece
Department of Computing, Imperial College London, UK

a.artikis@iit.demokritos.gr
m.sergot@imperial.ac.uk
paliourg@iit.demokritos.gr
Event Recognition

Problem:
- Event recognition (event pattern matching):
  - Input: Simple, derived events (SDE) coming from various types of sensor.
  - Output: Composite events (CE), i.e., collections of SDE and CE that satisfy some pattern.

Aim:
- Real-time CE recognition in large-scale DEBS.
- Formal & declarative semantics.

Approach:
- Highly efficient logic programming: Event Calculus.
Event Recognition for City Transport Management
## Event Recognition for City Transport Management

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>scheduled stop enter</td>
</tr>
<tr>
<td>215</td>
<td>late stop leave</td>
</tr>
<tr>
<td>[215, 400]</td>
<td>abrupt acceleration</td>
</tr>
<tr>
<td>[350, 600]</td>
<td>sharp turn</td>
</tr>
<tr>
<td>700</td>
<td>scheduled stop enter</td>
</tr>
<tr>
<td>705</td>
<td>passenger density</td>
</tr>
<tr>
<td>820</td>
<td>scheduled stop leave</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>Input</td>
<td>Output</td>
</tr>
<tr>
<td>-------------</td>
<td>---------------------------------------------</td>
</tr>
<tr>
<td>200</td>
<td>scheduled stop enter</td>
</tr>
<tr>
<td>215</td>
<td>late stop leave</td>
</tr>
<tr>
<td>[215, 400]</td>
<td>abrupt acceleration</td>
</tr>
<tr>
<td>[350, 600]</td>
<td>sharp turn</td>
</tr>
<tr>
<td>700</td>
<td>scheduled stop enter</td>
</tr>
<tr>
<td>705</td>
<td>passenger density change to high</td>
</tr>
<tr>
<td>820</td>
<td>scheduled stop leave</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>Input</td>
<td>Output</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>200</td>
<td>scheduled stop enter</td>
</tr>
<tr>
<td>215</td>
<td>late stop leave</td>
</tr>
<tr>
<td>[215, 400]</td>
<td>abrupt acceleration</td>
</tr>
<tr>
<td>[350, 600]</td>
<td>sharp turn</td>
</tr>
<tr>
<td>700</td>
<td>scheduled stop enter</td>
</tr>
<tr>
<td>705</td>
<td>passenger density change to high</td>
</tr>
<tr>
<td>820</td>
<td>scheduled stop leave</td>
</tr>
<tr>
<td>[215, 820]</td>
<td>non-punctual</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>
Event Calculus

- 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:
  - $F$ holds at a particular time-point if $F$ has been *initiated* by an event at some earlier time-point, and not *terminated* by another event in the meantime.
CE Definitions in the Event Calculus

CE definition:

\[
punctuality(ID) = non\_punctual \text{ initiated} \iff enter\_stop(ID, StopCode, \text{late}) \text{ happens} \text{ or leave\_stop(ID, StopCode, \text{early}) \text{ happens}}
\]

\[
punctuality(ID) = non\_punctual \text{ terminatedAt} \ T \iff enter\_stop(ID, StopCode, \text{scheduled}) \text{ happensAt} \ T', \text{ leave\_stop(ID, StopCode, \text{scheduled}) \text{ happensAt} } T
\]

CE recognition:

▶ \[
punctuality(ID) = non\_punctual \text{ holdsFor } I
\]
CE Definitions in the Event Calculus

CE definition:

\[
\text{driving\_quality}(ID) = \text{low} \iff \\
\text{punctuality}(ID) = \text{non\_punctual} \text{ or} \\
\text{driving\_style}(ID) = \text{unsafe}
\]

Compiled CE definition:

\[
\text{driving\_quality}(ID) = \text{low} \ \text{holdsFor} \ I_1 \cup I_2 \iff \\
\text{punctuality}(ID) = \text{non\_punctual} \ \text{holdsFor} \ I_1, \\
\text{driving\_style}(ID) = \text{unsafe} \ \text{holdsFor} \ I_2
\]
CE Definitions in the Event Calculus

CE definition:

\[
\text{driving\_quality}(ID) = \text{medium} \iff \\
\text{punctuality}(ID) = \text{punctual}, \\
\text{driving\_style}(ID) = \text{uncomfortable}
\]

Compiled CE definition:

\[
\text{driving\_quality}(ID) = \text{medium} \text{ holdsFor } I_1 \cap I_2 \iff \\
\text{punctuality}(ID) = \text{punctual} \text{ holdsFor } I_1, \\
\text{driving\_style}(ID) = \text{uncomfortable} \text{ holdsFor } I_2
\]
CE Definitions in the Event Calculus

CE definition:

\[
\text{driving}_\text{quality}(ID) = \text{high} \iff \\
\text{punctuality}(ID) = \text{punctual}, \\
\text{driving}_\text{style}(ID) \neq \text{unsafe}, \\
\text{driving}_\text{style}(ID) \neq \text{uncomfortable}
\]

Compiled CE definition:

\[
\text{driving}_\text{quality}(ID) = \text{high} \ \text{holdsFor} \ I_1 \ \setminus \ I_2 \cup I_3 \ \iff \\\n\text{punctuality}(ID) = \text{punctual} \ \text{holdsFor} \ I_1, \\
\text{driving}_\text{style}(ID) = \text{unsafe} \ \text{holdsFor} \ I_2, \\
\text{driving}_\text{style}(ID) = \text{uncomfortable} \ \text{holdsFor} \ I_3
\]
Run-Time Event Recognition

Real-time decision-support in the presence of:

- Very large SDE streams.
- Non-sorted SDE streams.
- SDE revision.
- Very large CE numbers.
Event Calculus: Run-Time Event Recognition
Event Calculus: Run-Time Event Recognition
Event Calculus: Run-Time Event Recognition

![Diagram of time line with working memory and transitions](image-url)
Event Calculus: Run-Time Event Recognition
Event Calculus: Run-Time Event Recognition
Event Calculus: Run-Time Event Recognition

Diagram showing a timeline labeled with time (t) and various segments labeled as $Q_1$, $Q_2$, $Q_3$, $Q_4$, and $Q_5$. There is a section labeled 'Working Memory' and some arrows indicating upward and downward transitions.
Event Calculus: Run-Time Event Recognition
Event Calculus: Run-Time Event Recognition
City Transport Management in Helsinki

- 7 sec = 2450 SDE
- 13 sec = 4550 SDE
- 19 sec = 6650 SDE
- 25 sec = 8750 SDE
- 31 sec = 10850 SDE
- 37 sec = 12950 SDE
- 43 sec = 15050 SDE

Working Memory

Time (ms)

1 processor
8 processors
City Transport Management in Very Big Cities

![Graph showing the relationship between time and working memory for a computational task with 8 processors.](image)

- 7 sec = 23331 SDE
- 13 sec = 43329 SDE
- 19 sec = 63327 SDE
- 25 sec = 83325 SDE
- 31 sec = 103323 SDE
- 37 sec = 123321 SDE
- 43 sec = 143319 SDE

Time (ms) vs. Working Memory for 8 processors.
Summary

Event Calculus for real-time CE recognition:

▶ ‘Windowing’ mechanism.
▶ A simple indexing mechanism means that we do not have to rely on SDE filtering modules.
▶ A form of caching stores the results of sub-computations in order to avoid unnecessary recomputations.
▶ A set of interval manipulation constructs simplify CE definitions and improve reasoning efficiency.
Summary

- Complex temporal representation:
  - Succinct representation $\rightarrow$ code maintenance.
  - Intuitive representation $\rightarrow$ facilitates interaction with domain experts unfamiliar with programming.
- Formal & declarative semantics.
Further Work

- Event recognition under uncertainty in the Event Calculus:
  - Erroneous SDE detection.
  - Incomplete SDE stream.
  - Imprecise CE definition.
- Machine learning in the Event Calculus:
  - Automated generation of CE definitions.