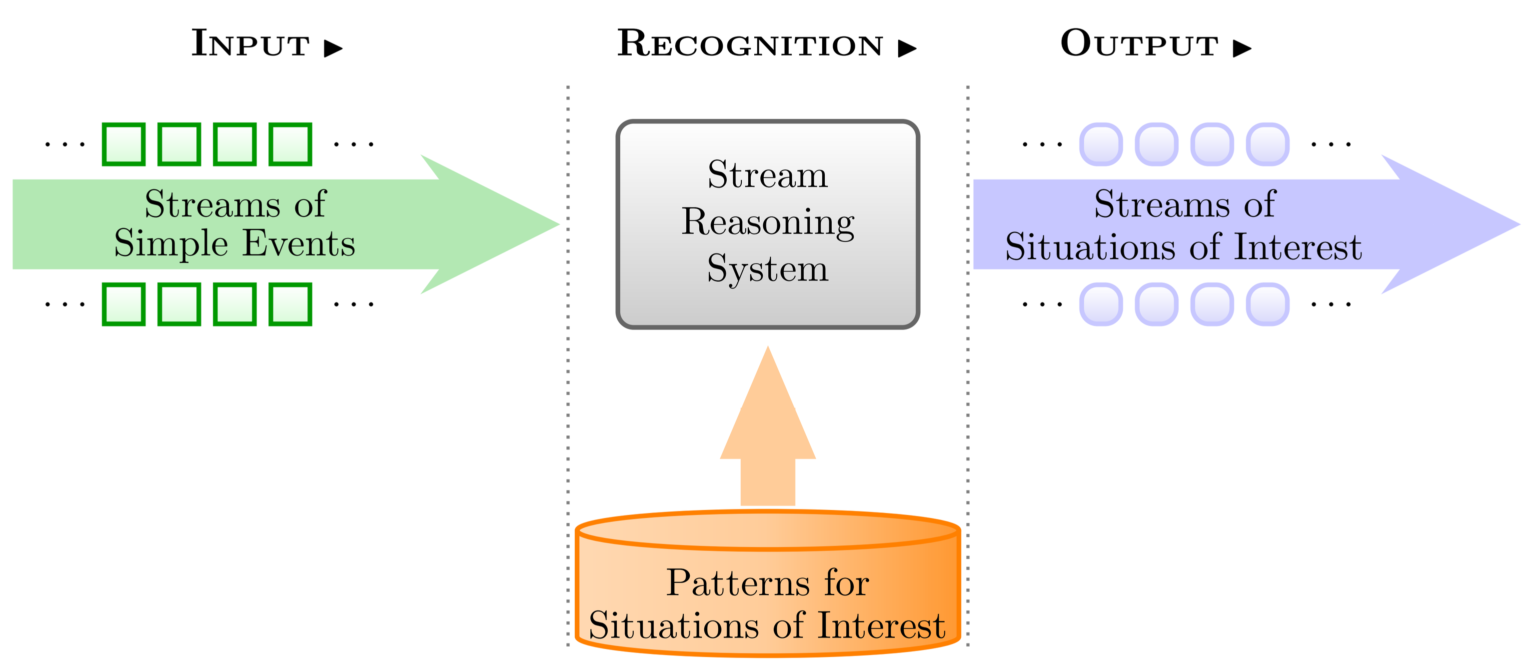
Temporal Specification Optimisation for the Event Calculus

Periklis Mantenoglou [1] Alexander Artikis [1 2]

[1]NCSR "Demokritos", Greece [2]University of Piraeus, Greece

Temporal Pattern Matching over Streams



The Run-Time Event Calculus (RTEC)

The **Event Calculus** is a logic programming formalism for representing and reasoning about the effects of events over time. Key components:

- Linear time-line with integer **time-points**.
- Instantaneous **events**.
- Time-varying properties called **fluents**.

A fluent-value pair (FVP) $F{=}V$ follows the **law of inertia**, i.e., in the absence of information to the contrary, fluent F continues to have value V over time, allowing for succinct and intuitive definitions for FVPs.

RTEC is a formal computational framework that derives FVPs over event streams. In RTEC, an FVP may be **simple** or **statically determined**.

Simple FVP (SF):

initiatedAt($F{=}V$, T) ←
  happensAt(E_{In_1}, T)[,
  conditions].
  ⋮
terminatedAt($F{=}V$, T) ←
  happensAt(E_{T_1}, T)[,
  conditions].
  ⋮
where conditions:
$^{0-K}$[not] happensAt(E_k, T),
$^{0-M}$[not] holdsAt($F_m{=}V_m$, T),
$^{0-N}$atemporal-constraint$_n$

Statically Determined FVP (SDF):

holdsFor($F{=}V$, I) ←
  holdsFor($F_1{=}V_1$, I_1)[,
  holdsFor($F_2{=}V_2$, I_2), ...
  holdsFor($F_n{=}V_n$, I_n),
  intervalConstruct(L_1, I_{n+1}), ...
  intervalConstruct(L_m, I)].

where intervalConstruct:
 union_all or
 intersect_all or
 relative_complement_all

We may use **FVPs to model situations of interest**.

Problem Statement & Proposed Solution

Challenges:

- Most Event Calculus specifications contain **only SFs**.
- The knowledge engineer may **detect only a portion of SFs** that can be re-written as **equivalent SDFs**.

Our Approach:

- Formal characterisation of the class of **SFs that are translatable into equivalent SDFs**.
- **Compiler** that identifies and re-writes them as SDFs.
- **Reproducible empirical evaluation** on numerous **real domain specifications**.

Example of a Translatable SF

In human activity recognition, we may use the FVP $meeting(P_1, P_2){=}interact$ in order to monitor when two people, P_1 and P_2, are having a meeting.

SF:

initiatedAt($meeting(P_1, P_2){=}interact$, T) ←
  happensAt(start($active(P_1)$=true), T),
  holdsAt($close(P_1, P_2)$=true, T),
  not happensAt(end($close(P_1, P_2)$=true), T).
initiatedAt($meeting(P_1, P_2){=}interact$, T) ←
  happensAt(start($close(P_1, P_2)$=true), T),
  holdsAt($active(P_1)$=true, T),
  not happensAt(end($active(P_1)$=true), T).
initiatedAt($meeting(P_1, P_2){=}interact$, T) ←
  happensAt(start($active(P_1)$=true), T),
  happensAt(start($close(P_1, P_2)$=true), T).
terminatedAt($meeting(P_1, P_2){=}interact$, T) ←
  happensAt(end($active(P_1)$=true), T).
terminatedAt($meeting(P_1, P_2){=}interact$, T) ←
  happensAt(end($close(P_1, P_2)$=true), T).

SDF:

holdsFor($meeting(P_1, P_2){=}interact$, I) ←
  holdsFor($active(P_1)$=true, I_a),
  holdsFor($close(P_1, P_2)$=true, I_c),
  intersect_all([I_a, I_c], I).

Boolean definition:

$meeting(P_1, P_2){=}interact \leftrightarrow$
$active(P_1)$=true∧$close(P_1, P_2)$=true

The above definitions for FVP $meeting(P_1, P_2){=}interact$ are **equivalent**, i.e., they lead to the same holdsAt($meeting(P_1, P_2){=}interact$, T) atoms, for every time-point T.

Key observation: The SF definition of $meeting(P_1, P_2){=}interact$ includes one initiation (termination) rule for each one of the possible ways of changing the truth value of its Boolean definition to true (false).

Theoretical Results

An SF is translatable to an SDF iff it is:

- **inertial condition symmetric**,
- **guard condition symmetric** and
- **Boolean representation symmetric**.

We have devised and implemented an algorithm that:

- identifies the SFs that are **translatable**, and
- maps them into **equivalent SDFs**.

Experimental Analysis

We evaluated our approach on Event Calculus rule-sets formalising:

- human activity recognition ($\mathcal{E}_h^i$).
- maritime situational awareness ($\mathcal{E}_m^i$).
- city transport management ($\mathcal{E}_t^i$).
- legal contract verification ($\mathcal{E}_l^i$).
- clinical guideline monitoring ($\mathcal{E}_g^i$).
- authorisation policy conflicts ($\mathcal{E}_c^i$).
- redundant authorisation policies ($\mathcal{E}_r^i$).

$\mathcal{E}_x^i$ were hand-crafted and contain only SFs. $\mathcal{E}_x^o$ is an optimised rule-set.