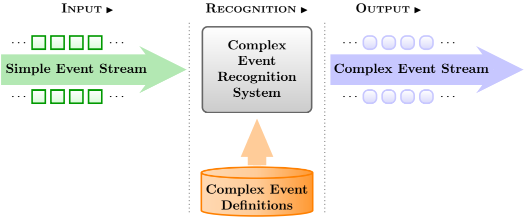
Extending the Range of Temporal Specifications of the Run-Time Event Calculus

Periklis Mantenoglou[1,2] Alexander Artikis[3,1]

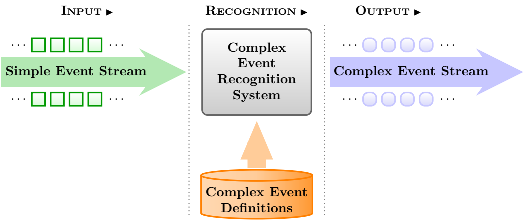
[1]NCSR Demokritos, Greece
[2]National and Kapodistrian University of Athens, Greece
[3]University of Piraeus, Greece

http://cer.iit.demokritos.gr/

Composite Event Recognition



Composite Event Recognition



https://cer.iit.demokritos.gr (activity recognition)

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.

Kowalski R., Sergot M., A Logic-based Calculus of Events. New Gener. Comput. 4(1): 67-95, 1986.

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:
  - $F = V$ 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.

Kowalski R., Sergot M., A Logic-based Calculus of Events. New Gener. Comput. 4(1): 67-95, 1986.

Run-Time Event Calculus (RTEC)

$$\textbf{initiatedAt}(F=V,\ T) \leftarrow$$
$$\quad \textbf{happensAt}(E_{In_1},\ T)[,$$
$$\quad \text{conditions}].$$
$$\ldots$$
$$\textbf{initiatedAt}(F=V,\ T) \leftarrow$$
$$\quad \textbf{happensAt}(E_{In_i},\ T)[,$$
$$\quad \text{conditions}].$$

$$\textbf{terminatedAt}(F=V,\ T) \leftarrow$$
$$\quad \textbf{happensAt}(E_{T_1},\ T)[,$$
$$\quad \text{conditions}].$$
$$\ldots$$
$$\textbf{terminatedAt}(F=V,\ T) \leftarrow$$
$$\quad \textbf{happensAt}(E_{T_j},\ T)[,$$
$$\quad \text{conditions}].$$

where

$$\text{conditions:} \qquad \begin{array}{l} {}^{0-K}\textbf{happensAt}(E_k,\ T), \\ {}^{0-M}\textbf{holdsAt}(F_m = V_m,\ T), \\ {}^{0-N}\text{atemporal-constraint}_n \end{array}$$

Artikis et al., An Event Calculus for Event Recognition. In IEEE Transactions on Knowledge and Data Engineering (TKDE), 27(4), 895–908, 2015.

Activity Recognition

$$
\begin{aligned}
&\textbf{initiatedAt}(\mathit{interaction}(P_1, P_2) = \mathit{greeting},\ T) \leftarrow \\
&\quad \textbf{happensAt}(\mathit{active}(P_1),\ T),\ \textbf{happensAt}(\mathit{active}(P_2),\ T), \\
&\quad \textbf{holdsAt}(\mathit{distance}(P_1, P_2) = \mathit{mid},\ T), \\
&\quad \textbf{holdsAt}(\mathit{orientation}(P_1, P_2) = \mathit{facing},\ T).
\end{aligned}
$$

Activity Recognition

$$
\begin{aligned}
&\textbf{initiatedAt}(\mathit{interaction}(P_1, P_2) = \mathit{greeting},\ T) \leftarrow \\
&\quad \textbf{happensAt}(\mathit{active}(P_1),\ T),\ \textbf{happensAt}(\mathit{active}(P_2),\ T), \\
&\quad \textbf{holdsAt}(\mathit{distance}(P_1, P_2) = \mathit{mid},\ T), \\
&\quad \textbf{holdsAt}(\mathit{orientation}(P_1, P_2) = \mathit{facing},\ T).
\end{aligned}
$$

$$
\begin{aligned}
&\textbf{terminatedAt}(\mathit{interaction}(P_1, P_2) = \mathit{greeting},\ T) \leftarrow \\
&\quad \textbf{happensAt}(\mathit{walking}(P_1),\ T), \\
&\quad \text{not } \textbf{holdsAt}(\mathit{orientation}(P_1, P_2) = \mathit{facing},\ T).
\end{aligned}
$$

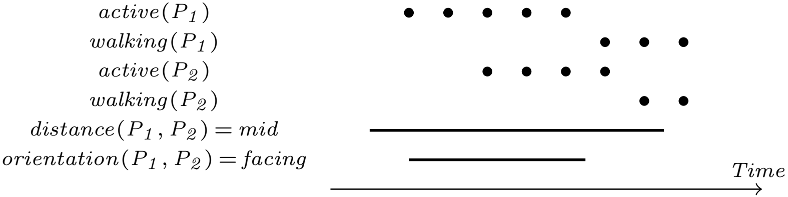
$$
\begin{aligned}
&\textbf{terminatedAt}(\mathit{interaction}(P_1, P_2) = \mathit{greeting},\ T) \leftarrow \\
&\quad \textbf{happensAt}(\mathit{walking}(P_2),\ T), \\
&\quad \text{not } \textbf{holdsAt}(\mathit{orientation}(P_1, P_2) = \mathit{facing},\ T).
\end{aligned}
$$

Activity Recognition

$\mathbf{initiatedAt}(\mathit{interaction}(P_1, P_2) = \mathit{greeting},\ T) \leftarrow$
$\quad \mathbf{happensAt}(\mathit{active}(P_1),\ T),\ \mathbf{happensAt}(\mathit{active}(P_2),\ T),$
$\quad \mathbf{holdsAt}(\mathit{distance}(P_1, P_2) = \mathit{mid},\ T),$
$\quad \mathbf{holdsAt}(\mathit{orientation}(P_1, P_2) = \mathit{facing},\ T).$

$\mathbf{terminatedAt}(\mathit{interaction}(P_1, P_2) = \mathit{greeting},\ T) \leftarrow$
$\quad \mathbf{happensAt}(\mathit{walking}(P_1),\ T),$
$\quad \text{not } \mathbf{holdsAt}(\mathit{orientation}(P_1, P_2) = \mathit{facing},\ T).$

$\mathbf{terminatedAt}(\mathit{interaction}(P_1, P_2) = \mathit{greeting},\ T) \leftarrow$
$\quad \mathbf{happensAt}(\mathit{walking}(P_2),\ T),$
$\quad \text{not } \mathbf{holdsAt}(\mathit{orientation}(P_1, P_2) = \mathit{facing},\ T).$

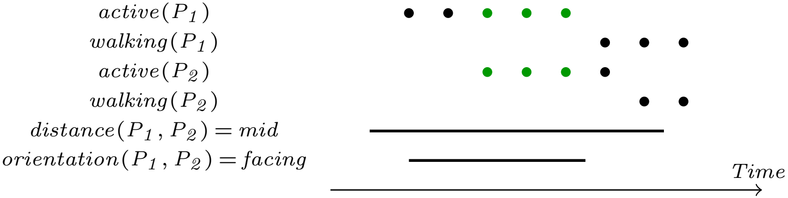


Activity Recognition

$$
\begin{aligned}
&\textbf{initiatedAt}(\mathit{interaction}(P_1, P_2) = \mathit{greeting},\ T) \leftarrow \\
&\quad \textbf{happensAt}(\mathit{active}(P_1),\ T),\ \textbf{happensAt}(\mathit{active}(P_2),\ T), \\
&\quad \textbf{holdsAt}(\mathit{distance}(P_1, P_2) = \mathit{mid},\ T), \\
&\quad \textbf{holdsAt}(\mathit{orientation}(P_1, P_2) = \mathit{facing},\ T).
\end{aligned}
$$

$$
\begin{aligned}
&\textbf{terminatedAt}(\mathit{interaction}(P_1, P_2) = \mathit{greeting},\ T) \leftarrow \\
&\quad \textbf{happensAt}(\mathit{walking}(P_1),\ T), \\
&\quad \text{not } \textbf{holdsAt}(\mathit{orientation}(P_1, P_2) = \mathit{facing},\ T).
\end{aligned}
$$

$$
\begin{aligned}
&\textbf{terminatedAt}(\mathit{interaction}(P_1, P_2) = \mathit{greeting},\ T) \leftarrow \\
&\quad \textbf{happensAt}(\mathit{walking}(P_2),\ T), \\
&\quad \text{not } \textbf{holdsAt}(\mathit{orientation}(P_1, P_2) = \mathit{facing},\ T).
\end{aligned}
$$

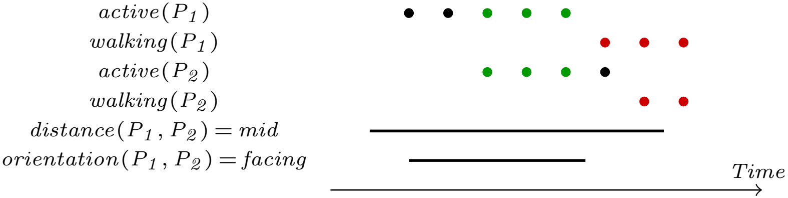


Activity Recognition

$$
\begin{aligned}
&\textbf{initiatedAt}(\mathit{interaction}(P_1, P_2) = \mathit{greeting},\ T) \leftarrow \\
&\quad \textbf{happensAt}(\mathit{active}(P_1),\ T),\ \textbf{happensAt}(\mathit{active}(P_2),\ T), \\
&\quad \textbf{holdsAt}(\mathit{distance}(P_1, P_2) = \mathit{mid},\ T), \\
&\quad \textbf{holdsAt}(\mathit{orientation}(P_1, P_2) = \mathit{facing},\ T).
\end{aligned}
$$

$$
\begin{aligned}
&\textbf{terminatedAt}(\mathit{interaction}(P_1, P_2) = \mathit{greeting},\ T) \leftarrow \\
&\quad \textbf{happensAt}(\mathit{walking}(P_1),\ T), \\
&\quad \text{not } \textbf{holdsAt}(\mathit{orientation}(P_1, P_2) = \mathit{facing},\ T).
\end{aligned}
$$

$$
\begin{aligned}
&\textbf{terminatedAt}(\mathit{interaction}(P_1, P_2) = \mathit{greeting},\ T) \leftarrow \\
&\quad \textbf{happensAt}(\mathit{walking}(P_2),\ T), \\
&\quad \text{not } \textbf{holdsAt}(\mathit{orientation}(P_1, P_2) = \mathit{facing},\ T).
\end{aligned}
$$

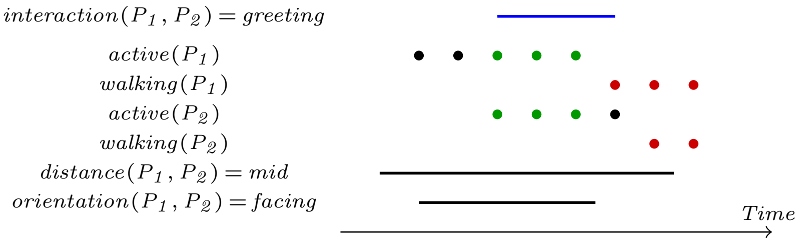


Activity Recognition

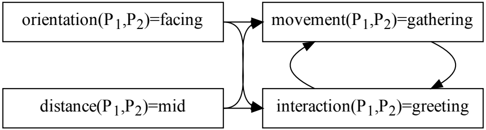
$$
\begin{aligned}
&\textbf{initiatedAt}(\mathit{interaction}(P_1, P_2) = \mathit{greeting},\ T) \leftarrow \\
&\quad \textbf{happensAt}(\mathit{active}(P_1),\ T),\ \textbf{happensAt}(\mathit{active}(P_2),\ T), \\
&\quad \textbf{holdsAt}(\mathit{distance}(P_1, P_2) = \mathit{mid},\ T), \\
&\quad \textbf{holdsAt}(\mathit{orientation}(P_1, P_2) = \mathit{facing},\ T).
\end{aligned}
$$

$$
\begin{aligned}
&\textbf{terminatedAt}(\mathit{interaction}(P_1, P_2) = \mathit{greeting},\ T) \leftarrow \\
&\quad \textbf{happensAt}(\mathit{walking}(P_1),\ T), \\
&\quad \text{not } \textbf{holdsAt}(\mathit{orientation}(P_1, P_2) = \mathit{facing},\ T).
\end{aligned}
$$

$$
\begin{aligned}
&\textbf{terminatedAt}(\mathit{interaction}(P_1, P_2) = \mathit{greeting},\ T) \leftarrow \\
&\quad \textbf{happensAt}(\mathit{walking}(P_2),\ T), \\
&\quad \text{not } \textbf{holdsAt}(\mathit{orientation}(P_1, P_2) = \mathit{facing},\ T).
\end{aligned}
$$

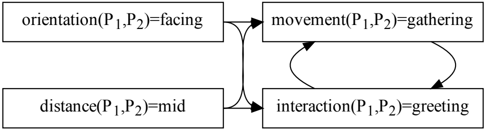


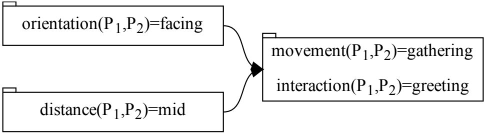
Semantics



Mantenoglou et al., Stream Reasoning with Cycles. In International Conference on Principles of Knowledge Representation and Reasoning (KR), 544–553, 2022.

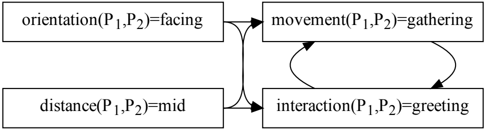
Semantics

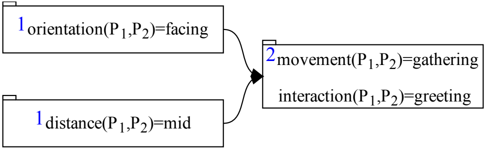




Mantenoglou et al., Stream Reasoning with Cycles. In International Conference on Principles of Knowledge Representation and Reasoning (KR), 544–553, 2022.

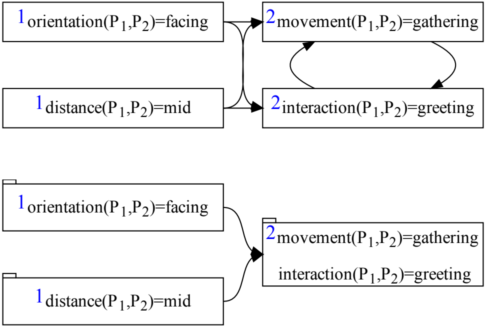
Semantics





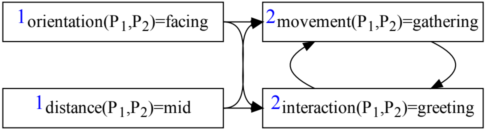
Mantenoglou et al., Stream Reasoning with Cycles. In International Conference on Principles of Knowledge Representation and Reasoning (KR), 544–553, 2022.

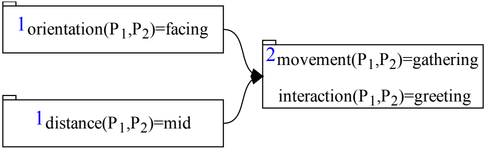
Semantics



Mantenoglou et al., Stream Reasoning with Cycles. In International Conference on Principles of Knowledge Representation and Reasoning (KR), 544–553, 2022.

Semantics



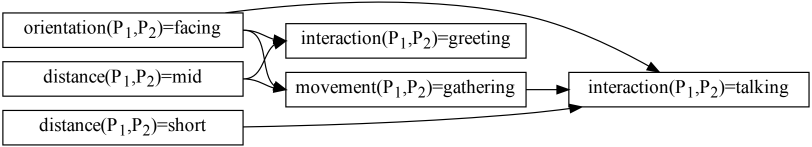


Semantics

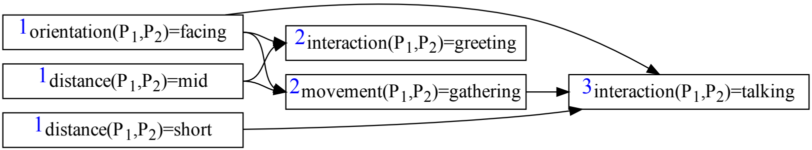
An event description of RTEC is a locally stratified logic program.

Mantenoglou et al., Stream Reasoning with Cycles. In International Conference on Principles of Knowledge Representation and Reasoning (KR), 544–553, 2022.

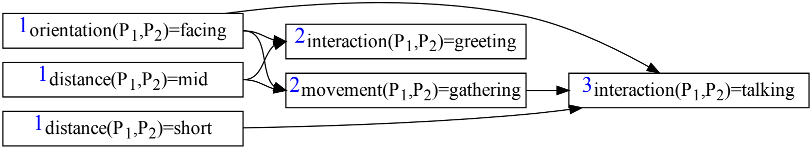
Problem Statement (1)



Problem Statement (1)

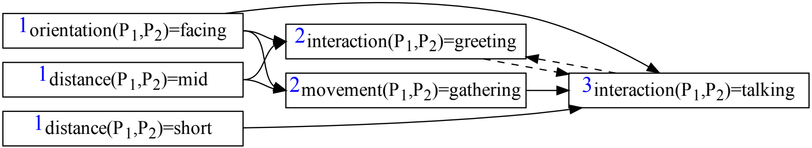


Problem Statement (1)



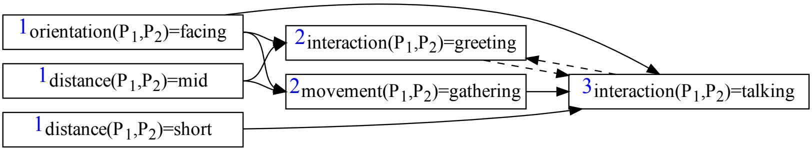
- A fluent may have at most one value at any time; an initiation of $F = V_1$ implies a termination of $F = V_2$, where $V_1 \neq V_2$.

Problem Statement (1)



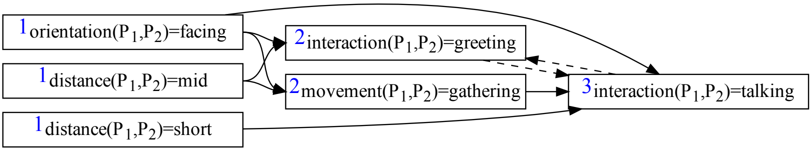
- A fluent may have at most one value at any time; an initiation of $F = V_1$ implies a termination of $F = V_2$, where $V_1 \neq V_2$.

Problem Statement (1)



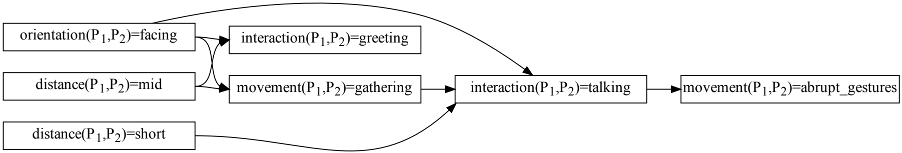
- A fluent may have at most one value at any time; an initiation of $F = V_1$ implies a termination of $F = V_2$, where $V_1 \neq V_2$.
- Implicit dependencies between fluent-value pairs with the same fluent may prohibit bottom-up processing.

Problem Statement (1)

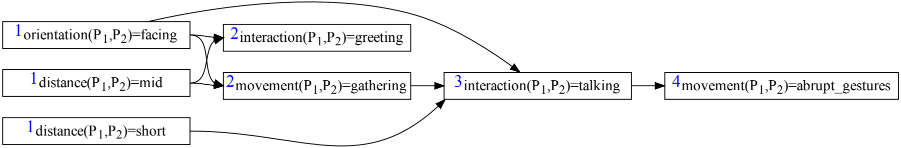


- A fluent may have at most one value at any time; an initiation of $F = V_1$ implies a termination of $F = V_2$, where $V_1 \neq V_2$.
- Implicit dependencies between fluent-value pairs with the same fluent may prohibit bottom-up processing.
- Solution: assign to fluent-value pairs of $interaction(P_1, P_2)$ a higher level than $movement(P_1, P_2) = gathering$.

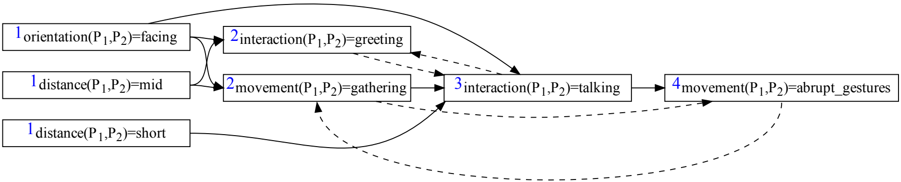
Problem Statement (2)



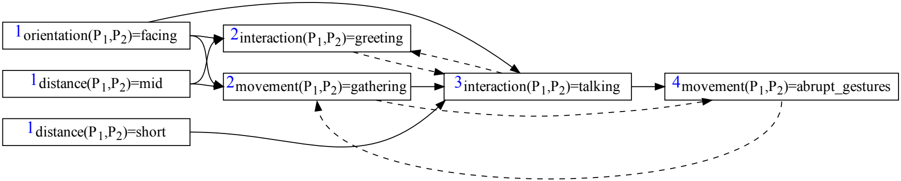
Problem Statement (2)



Problem Statement (2)

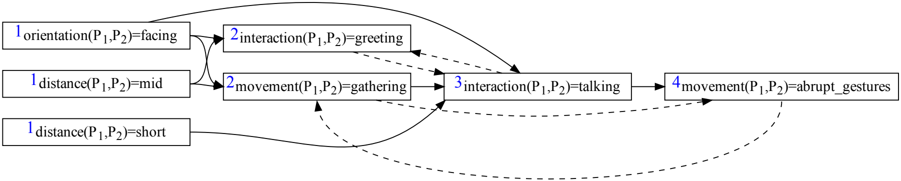


Problem Statement (2)



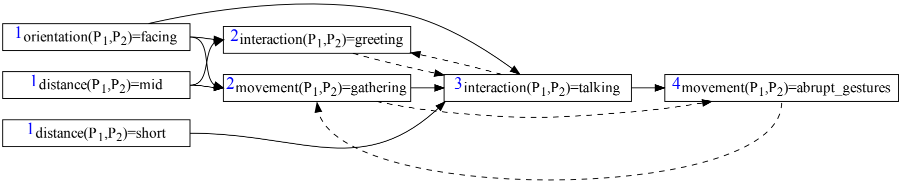
- We can have arbitrary implicit relations.

Problem Statement (2)



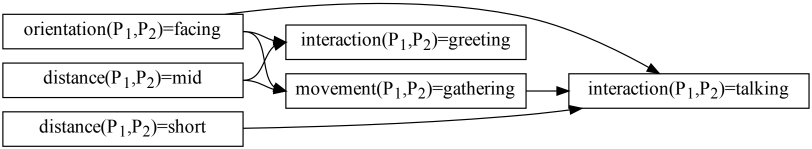
- We can have arbitrary implicit relations.
- Solution: assign to fluent-value pairs of $interaction(P_1, P_2)$ and $movement(P_1, P_2)$ the same level.

Problem Statement (2)

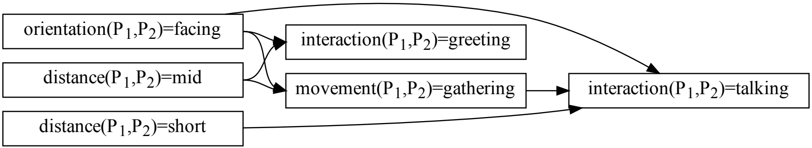


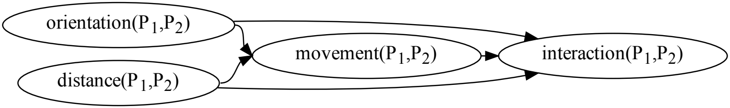
- We can have arbitrary implicit relations.
- Solution: assign to fluent-value pairs of $interaction(P_1, P_2)$ and $movement(P_1, P_2)$ the same level.
- RTEC_{fl}: An extension of RTEC that supports every possible dependency graph.

RTEC$_{fl}$: Fluent Dependency Graph

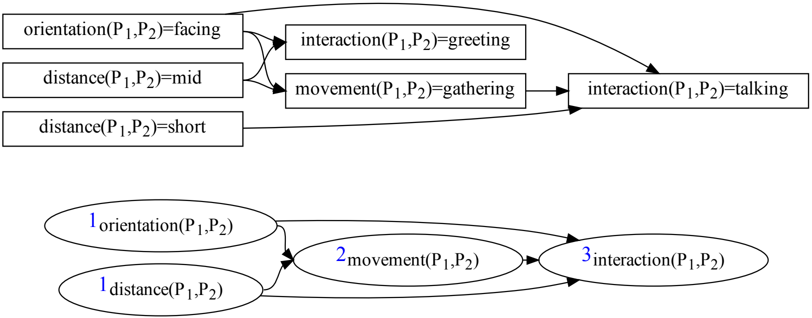


RTEC$_{fl}$: Fluent Dependency Graph

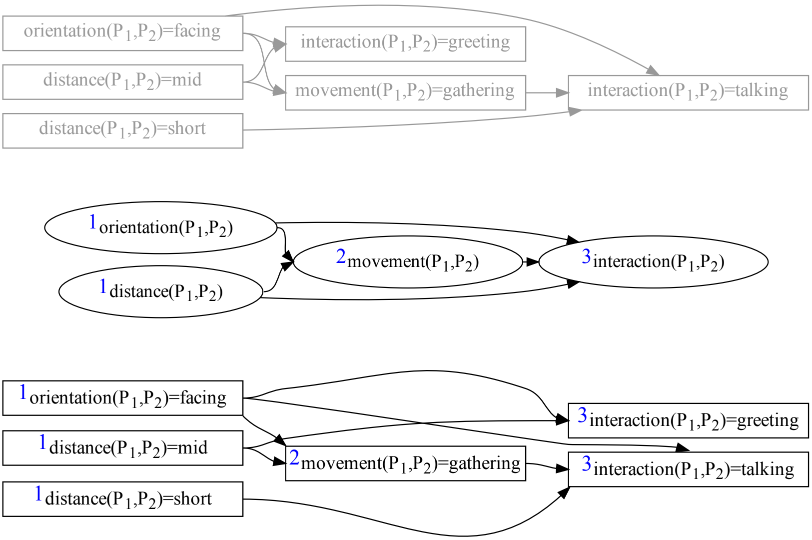




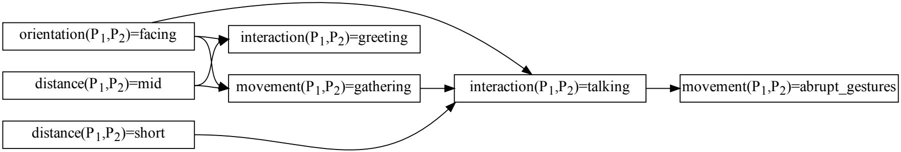
RTEC$_{fl}$: Fluent Dependency Graph



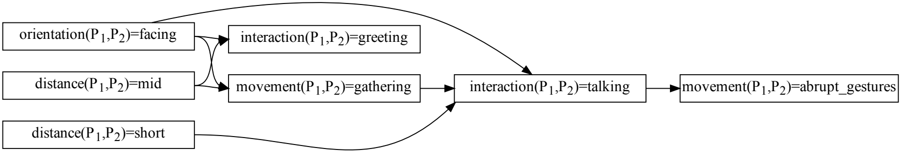
RTEC$_{fl}$: Fluent Dependency Graph

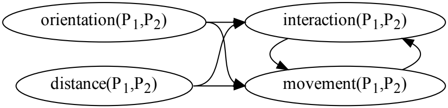


RTEC$_{fl}$: Fluent Dependency Graph

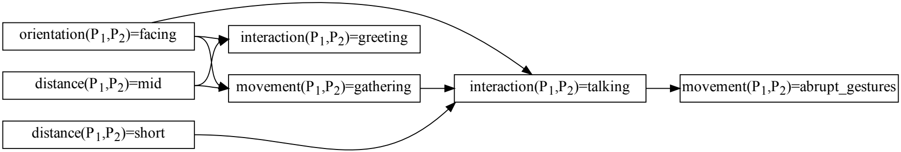


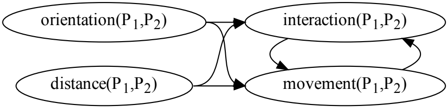
RTEC$_{fl}$: Fluent Dependency Graph

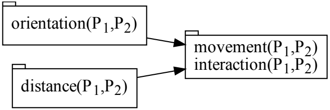




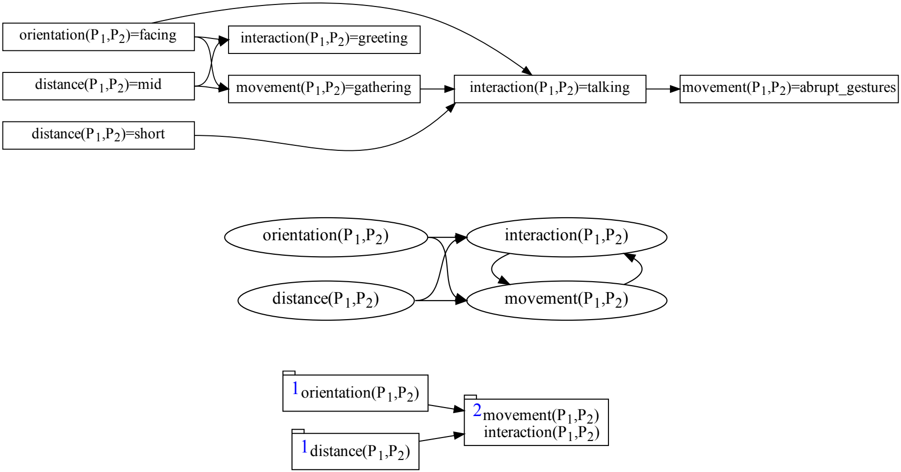
RTEC$_{fl}$: Contracted Dependency Graph



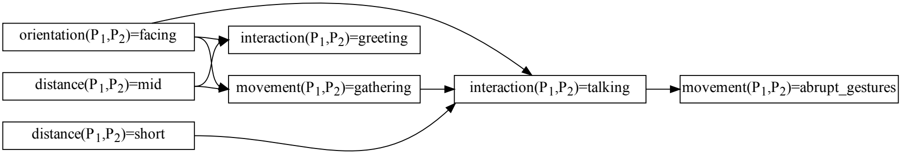


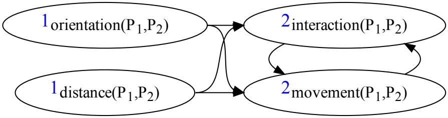


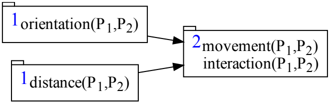
RTEC$_{fl}$: Contracted Dependency Graph



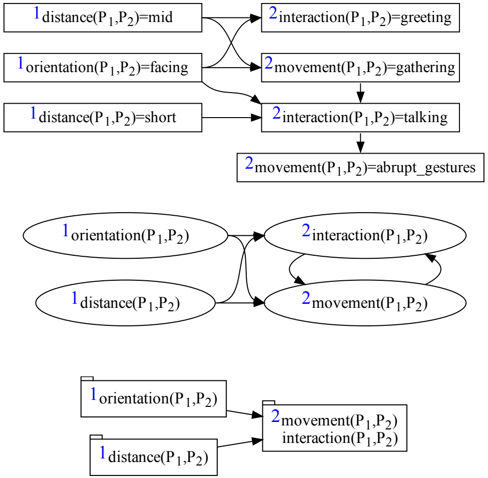
RTEC$_{fl}$: Contracted Dependency Graph



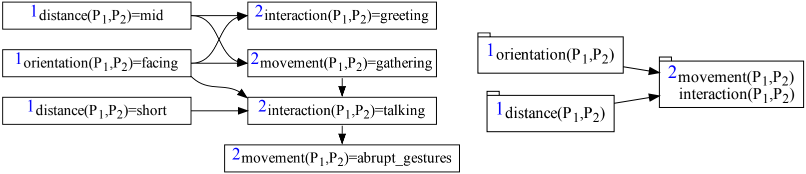




RTEC$_{fl}$: Contracted Dependency Graph



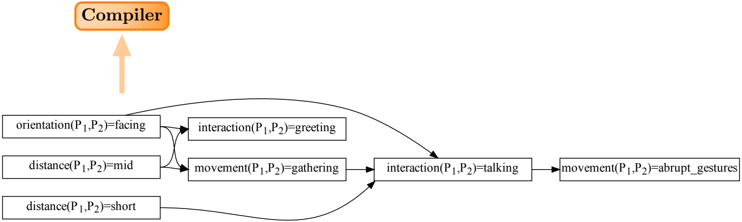
RTEC$_{fl}$: Semantics



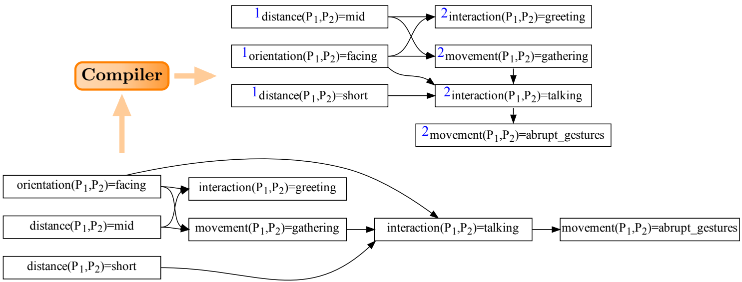
Semantics

An event description of RTEC$_{fl}$ is a locally stratified logic program.

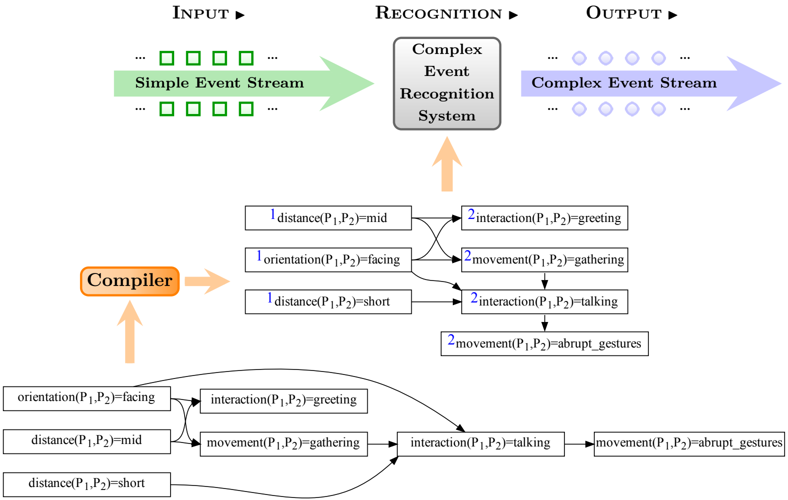
RTEC$_{fl}$: Compiler



RTEC$_{fl}$: Compiler



RTEC$_{fl}$: Compiler



https://github.com/aartikis/RTEC

Summary & Further Work

RTEC$_{fl}$:

- An open-source framework for composite event recognition.

Summary & Further Work

RTEC$_{fl}$:

- An open-source framework for composite event recognition.
- Wider range of specifications than RTEC.

Summary & Further Work

RTEC$_{fl}$:

- An open-source framework for composite event recognition.
- Wider range of specifications than RTEC.
- Same worst-case time complexity as RTEC.

Summary & Further Work

RTEC$_{fl}$:

- An open-source framework for composite event recognition.
- Wider range of specifications than RTEC.
- Same worst-case time complexity as RTEC.
- Semantics: Locally stratified specifications.

Summary & Further Work

$\mathrm{RTEC}_{\mathit{fl}}$:

- An open-source framework for composite event recognition.
- Wider range of specifications than RTEC.
- Same worst-case time complexity as RTEC.
- Semantics: Locally stratified specifications.
- Compiler: Support for any dependency graph.

Summary & Further Work

$\mathrm{RTEC}_{\mathit{fl}}$:

- An open-source framework for composite event recognition.
- Wider range of specifications than RTEC.
- Same worst-case time complexity as RTEC.
- Semantics: Locally stratified specifications.
- Compiler: Support for any dependency graph.
- Applications:
  - human activity recognition.
  - city transport management.
  - maritime situational awareness.
  - multi-agent systems monitoring.

Summary & Further Work

RTEC$_{fl}$:

- An open-source framework for composite event recognition.
- Wider range of specifications than RTEC.
- Same worst-case time complexity as RTEC.
- Semantics: Locally stratified specifications.
- Compiler: Support for any dependency graph.
- Applications:
  - human activity recognition.
  - city transport management.
  - maritime situational awareness.
  - multi-agent systems monitoring.

Further Work:

- Capture sequencing phenomena.
- Neuro-symbolic framework for composite event recognition.