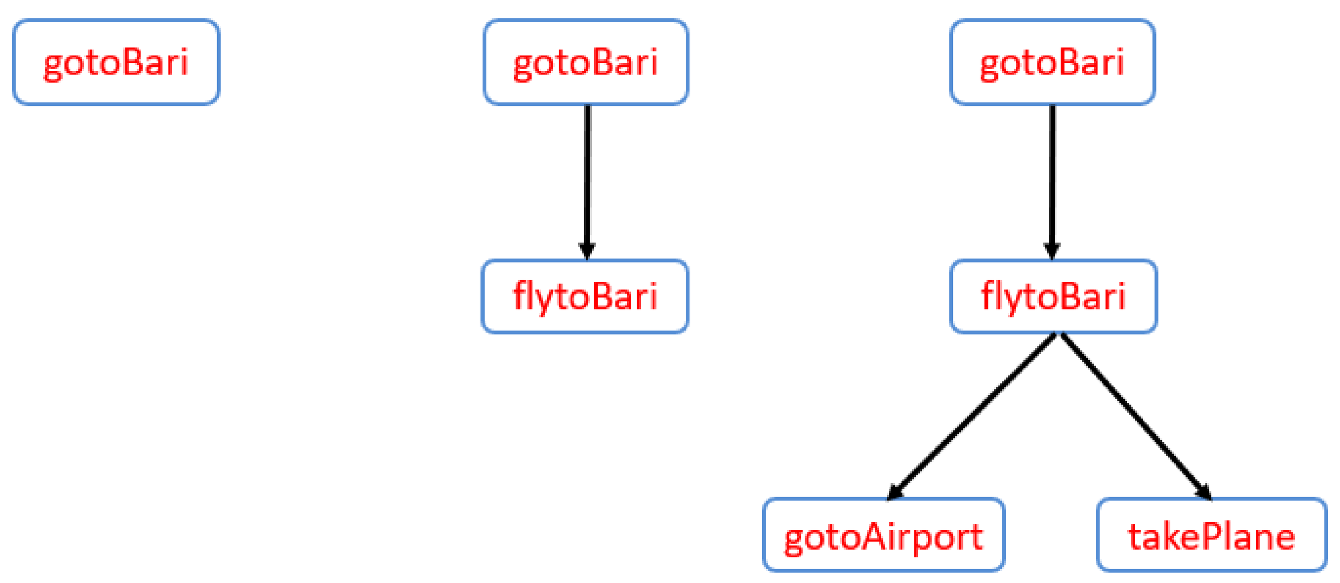
Deciding Refinement Relation in Belief-Intention Databases

Zhanhao Xiao[1,2], Andreas Herzig[1,3], Laurent Perrussel[1], Dongmo Zhang[2]
[1]IRIT, University of Toulouse, Toulouse, France
[2]SCEM, Western Sydney University, Penrith, Australia
[3]IRIT, CNRS, Toulouse, France
zhanhao.xiao@ut-capitole.fr

Motivation

Refinement: from a more abstract plan to a more elaborate plan.



(a) (b) (c)

(a) The initial intention of the agent is to go to Bari;
(b) this intention is refined by a lower-level intention to go to Bari by plane;
(c) flying to Bari is further refined into two intentions: going to the airport and taking the plane.

Main Definitions

Background

High level actions: the agent **cannot** directly do
Basic actions: the agent **can** directly do
Every action and event has precondition $\texttt{pre}(x)$ and postcondition $\texttt{post}(x)$

- only basic actions and events have **positive** effect and **negative** effect:
$$\texttt{post}(x) \leftrightarrow \bigwedge_{p \in \texttt{eff}^+(x)} p \wedge \bigwedge_{q \in \texttt{eff}^-(x)} \neg q$$
- **Coherence condition**: for any $a \in \text{Act}_0$, $E \subseteq \text{Evt}_0$, if $\texttt{pre}(\{a\} \cup E)$ is consistent, then $\texttt{post}(\{a\} \cup E)$ is consistent

Example

Alice has a high-level action '**buy**' of buying a movie ticket and a basic action 'buyWeb' of buying a ticket online. There is an exogenous event 'deliver' of the website delivering the electronic ticket.

$\texttt{pre}(\textbf{buy})$	$= \top$	$\texttt{post}(\textbf{buy})$	$= \text{Ticket}$
$\texttt{pre}(\text{wait})$	$= \top$	$\texttt{post}(\text{wait})$	$= \top$
$\texttt{pre}(\text{buyWeb})$	$= \top$	$\texttt{post}(\text{buyWeb})$	$= \text{PaidWeb}$
$\texttt{pre}(\text{deliver})$	$= \text{PaidWeb} \wedge \neg\text{Delivered}$	$\texttt{post}(\text{deliver})$	$= \text{Ticket} \wedge \text{Delivered}$

Belief-Intention Database

A belief-intention database is a finite set
$$\Delta \subseteq (\mathbb{N} \times \mathscr{L}_{\mathbb{P}}) \cup (\mathbb{N} \times \text{Evt}_0) \cup (\mathbb{N} \times \overline{\text{Evt}_0}) \cup (\mathbb{N} \times \text{Act} \times \mathbb{N}).$$

Example

Alice's initial database $\Delta_c = \{(0, \textbf{buy}, 2)\}$: Alice intends to buy a movie ticket within temporal interval $[0, 2]$.

Models

A model is tuple $\pi = \langle V, H, D \rangle$ where $V : \mathbb{N} \longrightarrow 2^{\mathbb{P}}$, $H : \mathbb{N} \longrightarrow 2^{\text{Evt}_0}$, and $D : \mathbb{N} \longrightarrow \text{Act}_0$, such that

- $V(t{+}1) = \Big(V(t) \cup \texttt{eff}^+\big(H(t) \cup \{D(t)\}\big)\Big) \setminus \texttt{eff}^-\big(H(t) \cup \{D(t)\}\big)$
- $H(t) = \{e \in \text{Evt}_0 \mid V(t) \models \texttt{pre}(e)\}$ event e happens *iff* $\texttt{pre}(e)$ is true
- $D(t) \in \{a \in \text{Act}_0 \mid V(t) \models \texttt{pre}(a)\}$ basic action a is done *implies* that $\texttt{pre}(a)$ is true

Satisfiability

A model $\pi = \langle V, H, D \rangle$ is a model of Δ, noted $\pi \Vdash \Delta$, if

- for every $(t, \varphi) \in \Delta$: $V(t) \models \varphi$;
- for every $(t, e) \in \Delta$: $e \in H(t)$;
- for every $(t, \bar{e}) \in \Delta$: $e \notin H(t)$;
- for every $(t, \alpha, d) \in \Delta$: there exists t', d' such that $t \leq t' < d' \leq d$, $V(t') \models \texttt{pre}(\alpha)$ and $V(d') \models \texttt{post}(\alpha)$; if α is basic action, $D(t') = \alpha$.

If Δ has a model then we say Δ is satisfiable.
If every model of Δ is also a model of Δ' then we say $\Delta \models \Delta'$.
If Δ' is a singleton $\{i\}$, we write $\Delta \models i$.

Example

$\Delta_c = \{(0, \textbf{buy}, 2)\}$ has a model:

- buyWeb at 0 and then wait (believing event deliver will occur at 1);
$D(0) = \text{buyWeb}, D(1) = \text{wait}, E(0) = \emptyset, E(1) = \text{deliver}$

Refinement of Intentions

Intention i can be refined by intention set J in Δ, noted $\Delta \models i \lhd J$, iff

1. there is no $j \in J$ such that $\Delta \models j$;
2. $\Delta \cup J$ has a model;
3. $(\Delta \cup J) \setminus \{i\} \models i$;
4. $(\Delta \cup J') \setminus \{i\} \not\models i$ for every $J' \subset J$;

Intuitively, to refine an intention i means to add a minimal set of new intentions J to the database which, together with other intentions but i, suffice to entail i.

Example

$\Delta_c \models (0, \textbf{buy}, 2) \lhd \{(0, \text{buyWeb}, 1)\}$

Complexity Results

Theorem

The satisfiability problem of a belief-intention database is PSPACE-complete.

Theorem

The consequence problem of deciding whether $\Delta' \models \Delta$ is PSPACE-complete.

Deciding refinement consists of satisfiability and consequence, so it is also PSPACE-complete.

Embedding in Linear Temporal Logic (PLTL)

Translating the background into a conjunction of PLTL formulas $\text{Tr}(\mathcal{D})$:

$$\bigwedge_{a \in \text{Act}_0} (do_a \to \texttt{pre}_a) \wedge \bigwedge_{e \in \text{Evt}_0} (h_e \leftrightarrow \texttt{pre}_e)$$
$$\wedge \bigvee_{a \in \text{Act}_0} do_a \wedge \bigwedge_{a,b \in \text{Act}_0, a \neq b} \neg(do_a \wedge do_b)$$
$$\wedge \bigwedge_{p \in \mathbb{P}} (\mathbf{X}p \leftrightarrow \bigvee_{\substack{e \in \text{Evt}_0 \\ p \in \texttt{eff}^+(e)}} h_e \vee \bigvee_{\substack{a \in \text{Act}_0 \\ p \in \texttt{eff}^+(a)}} do_a \vee (p \wedge \bigwedge_{\substack{e \in \text{Evt}_0 \\ p \in \texttt{eff}^-(e)}} \neg h_e \wedge \bigwedge_{\substack{a \in \text{Act}_0 \\ p \in \texttt{eff}^-(a)}} \neg do_a))$$
$$\wedge \bigwedge_{p \in \mathbb{P}} (\mathbf{X}\neg p \leftrightarrow \bigvee_{\substack{e \in \text{Evt}_0 \\ p \in \texttt{eff}^-(e)}} h_e \vee \bigvee_{\substack{a \in \text{Act}_0 \\ p \in \texttt{eff}^-(a)}} do_a \vee (\neg p \wedge \bigwedge_{\substack{e \in \text{Evt}_0 \\ p \in \texttt{eff}^+(e)}} \neg h_e \wedge \bigwedge_{\substack{a \in \text{Act}_0 \\ p \in \texttt{eff}^+(a)}} \neg do_a))$$

Example

- $(do_{\text{buyWeb}} \to \top) \wedge (h_{\text{deliver}} \leftrightarrow \mathit{PaidWed} \wedge \neg \mathit{Delivered})$
- Once Alice buy the ticket online, *PaidWeb* holds at the next state;
Once the ticket paid online, it is paid forever
$\mathbf{X}\mathit{PaidWeb} \leftrightarrow do_{\text{buyWeb}} \vee \mathit{PaidWeb}$
- If the event 'deliver' doesn't happen, *Delivered* keeps false
$\mathbf{X}\neg\mathit{Delivered} \leftrightarrow (\neg\mathit{Delivered} \wedge \neg h_{\text{deliver}})$

Translating the database into a conjunction of PLTL formulas $\text{Tr}(\Delta)$:

$$\bigwedge_{(t,\varphi) \in \Delta} \mathbf{X}^t \varphi \wedge \bigwedge_{(t,e) \in \Delta} \mathbf{X}^t h_e \wedge \bigwedge_{(t,\bar{e}) \in \Delta} \mathbf{X}^t \neg h_e$$
$$\wedge \bigwedge_{\substack{\alpha \notin \text{Act}_0 \\ (t,\alpha,d) \in \Delta}} \bigvee_{t \leq t' < d' \leq d} (\mathbf{X}^{t'} pre_\alpha \wedge \mathbf{X}^{d'} post_\alpha)$$
$$\wedge \bigwedge_{\substack{a \in \text{Act}_0 \\ (t,a,d) \in \Delta}} \bigvee_{t \leq t' < d} \mathbf{X}^{t'} do_a$$

Example

To achieve the intention $(0, \textbf{buy}, 2)$:

- its precondition $\top$ holds at state 0 and postcondition *Ticket* holds at 1
$\top \wedge \mathbf{X}\mathit{Ticket}$
- or its precondition $\top$ holds at state 0 and postcondition *Ticket* holds at 2
$\top \wedge \mathbf{XX}\mathit{Ticket}$
- or its precondition $\top$ holds at state 1 and postcondition *Ticket* holds at 2
$\mathbf{X}\top \wedge \mathbf{XX}\mathit{Ticket}$

Theorem

A database Δ is satisfiable iff $\mathbf{G}\,Tr(\mathcal{D}) \wedge Tr(\Delta)$ is satisfiable.

Theorem

$\Delta \models \Delta'$ iff $\mathbf{G}\,Tr(\mathcal{D}) \to (Tr(\Delta) \to Tr(\Delta'))$ is valid.