Towards Dynamic, Metric and Temporal Answer Set Programming over Linear Finite Traces

Pedro Cabalar Martín Diéguez Torsten Schaub

University of Corunna, Spain

Université d'Angers, France

University of Potsdam, Germany

Potassco

Introduction

- Objective

  Extend Answer Set Programming (ASP) with means for representing and reasoning about dynamic knowledge

Introduction

- Objective

  Extend Answer Set Programming (ASP) with means for representing and reasoning about dynamic knowledge

- Approach

  Extend the base logic of ASP, namely the logic of Here-and-There (HT), with language elements from

  - Temporal Logic (LTL)
  - Dynamic Logic (LDL)
  - Metric Logic (MTL)

  over a common semantic structure, namely, finite linear HT traces

Introduction

- Objective

  Extend Answer Set Programming (ASP) with means for representing and reasoning about dynamic knowledge

- Approach

  Extend the base logic of ASP, namely the logic of Here-and-There (HT), with language elements from

  - Temporal Logic (LTL)
  - Dynamic Logic (LDL)
  - Metric Logic (MTL)

  over a common semantic structure, namely, finite linear HT traces

- Origin Temporal logic of Here-and-There (Cabalar and Pérez, 2007) over infinite traces

Potassco

The logic of Here-and-There

- Origin

  Three valued logic due to (Heyting, 1930; Gödel, 1932)
  - HT is based on Kripke semantics for intuitionistic logic
  - An HT model is a pair (H, T) such that $H \subseteq T$
  - Implication is a genuine connective

The logic of Here-and-There

- Origin

  Three valued logic due to (Heyting, 1930; Gödel, 1932)

  - HT is based on Kripke semantics for intuitionistic logic
  - An HT model is a pair (H, T) such that $H \subseteq T$
  - Implication is a genuine connective, and
    negation is defined in terms of implication: $\neg\varphi = \varphi \rightarrow \bot$

The logic of Here-and-There

- Origin

  Three valued logic due to (Heyting, 1930; Gödel, 1932)

  - HT is based on Kripke semantics for intuitionistic logic
  - An HT model is a pair (H, T) such that $H \subseteq T$
  - Implication is a genuine connective

Potassco

The logic of Here-and-There

- Origin

  Three valued logic due to (Heyting, 1930; Gödel, 1932)

  - HT is based on Kripke semantics for intuitionistic logic
  - An HT model is a pair (H, T) such that $H \subseteq T$
  - Implication is a genuine connective

- Discovery (Pearce, 1996)

  Minimal HT models correspond to answer sets

The logic of Here-and-There

- Origin

  Three valued logic due to (Heyting, 1930; Gödel, 1932)

  - HT is based on Kripke semantics for intuitionistic logic
  - An HT model is a pair (H, T) such that $H \subseteq T$
  - Implication is a genuine connective

- Discovery (Pearce, 1996)

  Minimal HT models correspond to answer sets, more precisely, an answer set T of ϕ is

  - a total HT model (T, T) of ϕ and
  - there is no $H \subset T$ such that (H, T) is an HT model of ϕ

The logic of Here-and-There

- Origin

  Three valued logic due to (Heyting, 1930; Gödel, 1932)

  - HT is based on Kripke semantics for intuitionistic logic
  - An HT model is a pair (H, T) such that $H \subseteq T$
  - Implication is a genuine connective

- Discovery (Pearce, 1996)

  Minimal HT models correspond to answer sets, more precisely, an answer set T of ϕ is

  - a total HT model (T, T) of ϕ and
  - there is no $H \subset T$ such that (H, T) is an HT model of ϕ

  Such models are are called equilibrium models

The logic of Here-and-There

- Origin (**monotonic**)

  Three valued logic due to (Heyting, 1930; Gödel, 1932)

  - HT is based on Kripke semantics for intuitionistic logic
  - An HT model is a pair (H, T) such that $H \subseteq T$
  - Implication is a genuine connective

- Discovery (Pearce, 1996) (**non-monotonic**)

  Minimal HT models correspond to answer sets, more precisely, an answer set T of ϕ is

  - a total HT model (T, T) of ϕ and
  - there is no $H \subset T$ such that (H, T) is an HT model of ϕ

  Such models are are called equilibrium models

Dynamic and Temporal Logic of Here-and-There

Dynamic and Temporal Logic of Here-and-There

- Structure An HT trace is a sequence $(H_i, T_i)_{i=0}^{\lambda}$ of HT models

Dynamic and Temporal Logic of Here-and-There

- Structure An HT trace is a sequence $(H_i, T_i)_{i=0}^{\lambda}$ of HT models
- Satisfaction $(H_i, T_i)_{i=0}^{\lambda} = (\mathbf{H}, \mathbf{T})$

Dynamic and Temporal Logic of Here-and-There

- Structure An HT trace is a sequence $(H_i, T_i)_{i=0}^{\lambda}$ of HT models
- Satisfaction $(H_i, T_i)_{i=0}^{\lambda} = (\mathbf{H}, \mathbf{T})$
  - Something Boolean $(\mathbf{H}, \mathbf{T}), k \models \varphi \rightarrow \psi$ if $(\mathbf{H}', \mathbf{T}), k \not\models \varphi$ or $(\mathbf{H}', \mathbf{T}), k \models \psi$, for all $\mathbf{H}' \in \{\mathbf{H}, \mathbf{T}\}$

Potassco

Dynamic and Temporal Logic of Here-and-There

- Structure An HT trace is a sequence $(H_i, T_i)_{i=0}^{\lambda}$ of HT models
- Satisfaction $(H_i, T_i)_{i=0}^{\lambda} = (\mathbf{H}, \mathbf{T})$
  - Something Boolean $(\mathbf{H}, \mathbf{T}), k \models \varphi \rightarrow \psi$ if
    $(\mathbf{H}', \mathbf{T}), k \not\models \varphi$ or $(\mathbf{H}', \mathbf{T}), k \models \psi$, for all $\mathbf{H}' \in \{\mathbf{H}, \mathbf{T}\}$
  - Something Temporal
    $(\mathbf{H}, \mathbf{T}), k \models \Box\varphi$ if $(\mathbf{H}, \mathbf{T}), i \models \varphi$ for any $i = k..\lambda$
    $(\mathbf{H}, \mathbf{T}), k \models \Diamond\varphi$ if $(\mathbf{H}, \mathbf{T}), i \models \varphi$ for some $i = k..\lambda$

Potassco

Dynamic and Temporal Logic of Here-and-There

- Structure An HT trace is a sequence $(H_i, T_i)_{i=0}^{\lambda}$ of HT models
- Satisfaction $(H_i, T_i)_{i=0}^{\lambda} = (\mathbf{H}, \mathbf{T})$
  - Something Boolean $(\mathbf{H}, \mathbf{T}), k \models \varphi \to \psi$ if
    $(\mathbf{H}', \mathbf{T}), k \not\models \varphi$ or $(\mathbf{H}', \mathbf{T}), k \models \psi$, for all $\mathbf{H}' \in \{\mathbf{H}, \mathbf{T}\}$
  - Something Temporal
    $(\mathbf{H}, \mathbf{T}), k \models \Box\varphi$ if $(\mathbf{H}, \mathbf{T}), i \models \varphi$ for any $i = k..\lambda$
    $(\mathbf{H}, \mathbf{T}), k \models \Diamond\varphi$ if $(\mathbf{H}, \mathbf{T}), i \models \varphi$ for some $i = k..\lambda$
  - Something Dynamic $(\mathbf{H}, \mathbf{T}), k \models [\rho]\varphi$ if
    $(\mathbf{H}', \mathbf{T}), i \models \varphi$ for all $i = 0..\lambda$ with $(k, i) \in \parallel \rho \parallel^{(\mathbf{H}', \mathbf{T})}$,
    for all $\mathbf{H}' \in \{\mathbf{H}, \mathbf{T}\}$

Potassco

Pedestrian traffic light

Metric equilibrium logic

$$\square(\mathit{red} \land \mathit{green} \to \bot) \tag{1}$$

$$\square(\neg \mathit{green} \to \mathit{red}) \tag{2}$$

$$\square\big(\mathit{push} \to \Diamond_{[1..15)}(\square_{\leq 30}\ \mathit{green})\big) \tag{3}$$

Pedestrian traffic light

Metric equilibrium logic

$$\square(\mathit{red} \wedge \mathit{green} \to \bot) \tag{1}$$

$$\square(\neg \mathit{green} \to \mathit{red}) \tag{2}$$

$$\square\big(\mathit{push} \to \Diamond_{[1..15)}(\square_{\leq 30}\, \mathit{green})\big) \tag{3}$$

- $\{(1), (2), (3)\} \models_{MEL} \square(\mathit{red} \wedge \neg \mathit{green} \wedge \neg \mathit{push})$

Pedestrian traffic light

Metric equilibrium logic

$$\square(\mathit{red} \wedge \mathit{green} \rightarrow \bot) \tag{1}$$

$$\square(\neg \mathit{green} \rightarrow \mathit{red}) \tag{2}$$

$$\square\big(\mathit{push} \rightarrow \Diamond_{[1..15)}(\square_{\leq 30}\ \mathit{green})\big) \tag{3}$$

- $\{(1),(2),(3)\} \models_{MEL} \square(\mathit{red} \wedge \neg \mathit{green} \wedge \neg \mathit{push})$
- $\{(1),(2),(3), \circ_5\ \mathit{push}\}$ has 14 metric equilibrium models of length 3
  - $T_0 = \{\mathit{red}\}$ $\tau(0) = 0$
  - $T_1 = \{\mathit{push}, \mathit{red}\}$ $\tau(1) = 5$
  - $T_2 = \{\mathit{green}\}$ $\tau(2) \in \{6, \ldots, 19\}$

Potassco

telingo

- **telingo**
  - extends the full modeling language of **clingo** with (past and future) temporal operators
  - relies on finite traces
  - implements an incremental translation

telingo

- **telingo** — `https://github.com/potassco/telingo`
  - extends the full modeling language of **clingo** with (past and future) temporal operators
  - relies on finite traces
  - implements an incremental translation

Potassco

telingo

- **telingo** — `https://github.com/potassco/telingo`
  - extends the full modeling language of **clingo** with (past and future) temporal operators
  - relies on finite traces
  - implements an incremental translation
- Primes allow for expressing (iterated) next and previous operators
  - $\bullet p(a)$ and $\circ q(b)$ can be expressed by `'p(a)` and `q'(b)`

telingo

- **telingo** — `https://github.com/potassco/telingo`
  - extends the full modeling language of **clingo** with (past and future) temporal operators
  - relies on finite traces
  - implements an incremental translation
- Primes allow for expressing (iterated) next and previous operators
  - $\bullet p(a)$ and $\circ q(b)$ can be expressed by `'p(a)` and `q'(b)`
- Example *"A robot cannot lift a box unless its capacity exceeds the box's weight plus that of all held objects"*

```
:- lift(R,B), robot(R), box(B,W),
   #sum {   C : capacity(R,C);
          -V,O : 'holding(R,O,V) } < W.
```

Summary

- **ASP + Temporal, Dynamic, and Metric Logics**

  via extending HT over the common semantic structure of finite HT traces

Summary

- **ASP + Temporal, Dynamic, and Metric Logics**

  via extending HT over the common semantic structure of finite HT traces

- Interested? JANCL'13, ICLP'18, KR'18, LPNMR'19, TPLP'23

Summary

- **ASP + Temporal, Dynamic, and Metric Logics**

  via extending HT over the common semantic structure
  of finite HT traces

- Interested? JANCL'13, ICLP'18, KR'18, LPNMR'19, TPLP'23
- Playful? `https://github.com/potassco/telingo`

Potassco

Summary

- **ASP + Temporal, Dynamic, and Metric Logics**

  via extending HT over the common semantic structure of finite HT traces

- Interested? JANCL'13, ICLP'18, KR'18, LPNMR'19, TPLP'23
- Playful?[1] `https://github.com/potassco/telingo`

[1] Classical logic is obtained in ASP by adding choices; eg., '`{a}.`' stands for '$a \vee \neg a$'.