Planning for Temporally Extended Goals in Pure-Past Linear Temporal Logic

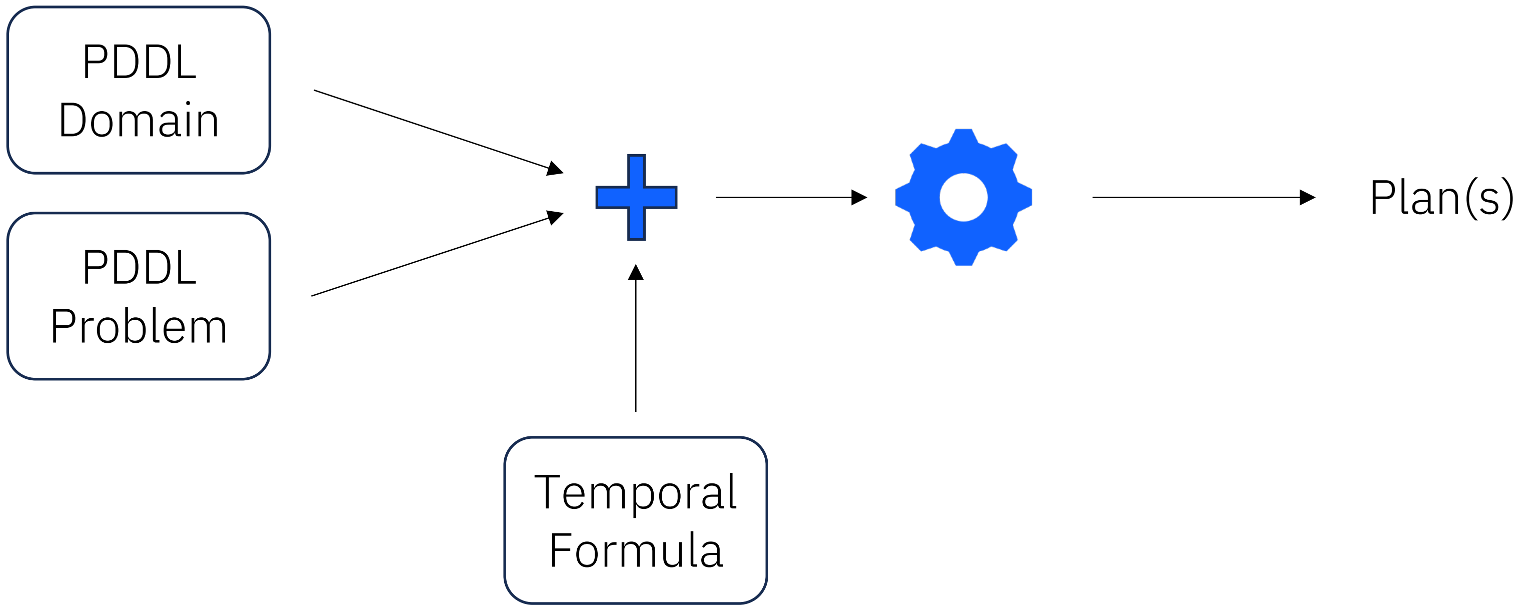
Luigi Bonassi, Giuseppe De Giacomo, Marco Favorito,
Francesco Fuggitti, Alfonso Gerevini, Enrico Scala

francesco.fuggitti@gmail.com

ICAPS 2023

Planning for Temporally Extended Goals

Objective: Restrict the way the planner achieves the goal



Deterministic Planning for Temporally Extended Goals

Deterministic Planning and Temporally Extended Goals

Planning for Temporally Extended Goals. Bacchus and Kabanza. AAAI, 1996.

Using Temporal Logics to Express Search Control Knowledge for Planning. Bacchus and Kabanza. AIJ, 2000.

Structured Solution Methods for non-Markovian Decision Processes. Bacchus, Boutilier, and Grove. AAAI, 1997.

via Model-Checking

Planning as Model Checking. Giunchiglia, Giunchiglia and Traverso. ECP, 1999.

Planning via Model Checking: A Decision Procedure for AR. Cimatti, Giunchiglia, Giunchiglia, and Traverso. ECP, 1997.

Automata-theoretic Approach to Planning for Temporally Extended Goals. De Giacomo and Vardi. ECP, 1999.

Finite-trace variant of LTL

Planning with Temporally Extended Goals Using Heuristic Search. Baier and McIlraith. ICAPS, 2006.

Planning with First-order Temporally Extended Goals Using Heuristic Search. Baier and McIlraith. AAAI, 2006.

Polynomial-time Reformulations of LTL Temporally Extended Goals into Final-state Goals. Torres and Baier. ICAPS, 2015.

Declarative/Procedural constraints

Beyond Classical Planning: Procedural Control Knowledge and Preferences in State-of-the-Art Planners. Baier, Fritz, Bienvenu, and McIlraith. AAAI, 2008.

Deterministic Planning in the 5th IPC: PDDL3 and Experimental Evaluation of Planners. Gerevini, Haslum, Long, Saetti, and Dimopoulos. AIJ, 2009.

Deterministic Planning for Temporally Extended Goals

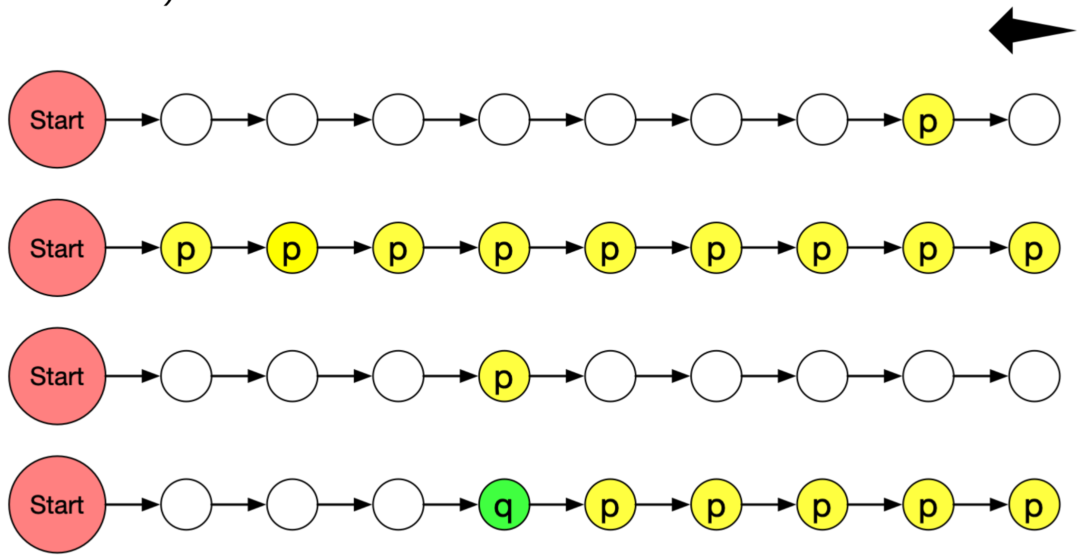
	Reachability Goal	Temporally Extended Goal (e.g., LTLf)
Complexity	PSPACE-complete	PSPACE-complete

- Looking at the details, due to the LTLf nature, state-of-the-art encodings for LTLf are
    - Exponential, but preserves the plan length [Baier & McIlraith, 2006]
    - Polynomial, but increases polynomially the plan length [Torres & Baier, 2015]

In our work, using PPLTL, we introduce a linear encoding of planning for PPLTL goals into classical planning that preserves plan length

Pure-Past Linear Temporal Logic (PPLTL)

- Among the compelling formal languages to express temporal specifications evaluated on finite traces
- Talks about properties of the world and uses past temporal operators only: (Y)esterday, (H)istorically, (O)nce, (S)ince
- Contrary to LTLf, PPLTL looks at the trace backward, and evaluates formulas on the last instant of the trace (i.e., the current instant)
  - Y(p)
  - H(p)
  - O(p)
  - p S q



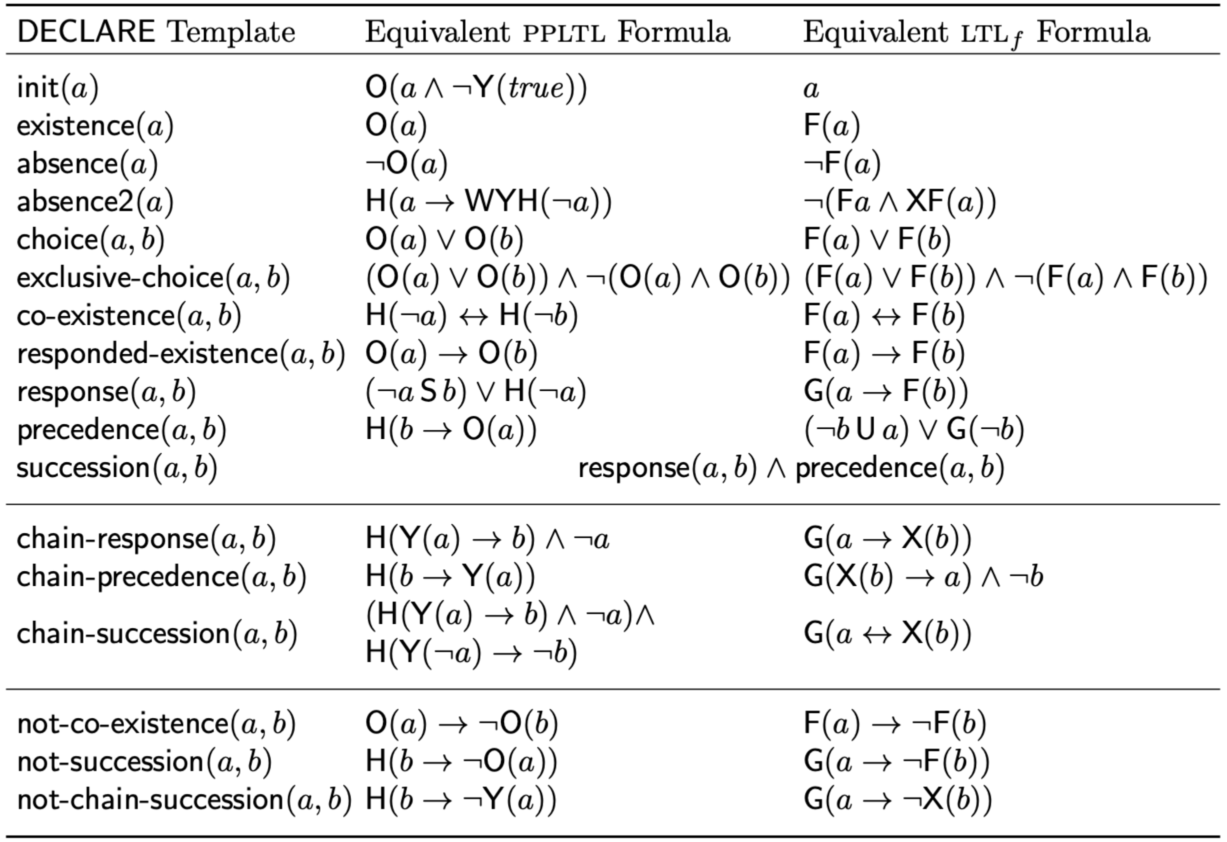
Computational Properties of PPLTL

- Equally expressive as LTLf, but translating one into the other is prohibitive (3EXPTIME) [De Giacomo et al., 2020]
- Translation of PPLTL formulas to Deterministic Finite-state Automaton (DFA) is worst-case *single exponential* in the size of the formula (vs. double exponential for LTLf to DFA) [Chandra et al., 1981; De Giacomo et al., 2020]

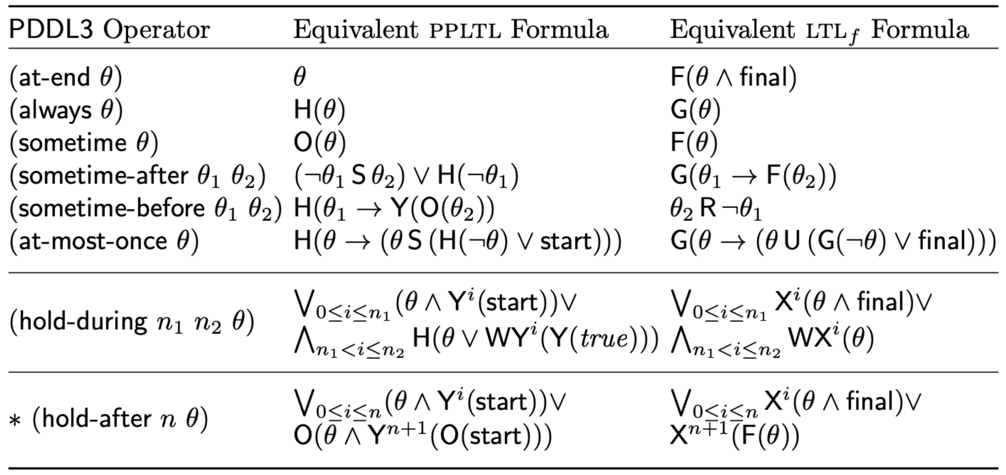
PPLTL in Planning

- Little attention in AI planning, but commonly employed in other areas of AI
  - non-Markovian rewards in MDPs [Bacchus et al., 1996]
  - non-Markovian models [Gabaldon 2011]
  - explanations in dynamical systems [Sohrabi et al., 2011]
  - norms in multi-agent systems [Fisher & Wooldridge, 2005; Knobbout et al., 2016; Alechina et al., 2018]
  - synthesis specifications [Cimatti et al., 2020]

- Many interesting properties expressed in LTLf are polynomially related (in their size) to their semantic equivalent PPLTL (and vice versa)



DECLARE Template	Equivalent PPLTL Formula	Equivalent LTL$_f$ Formula
init(a)	$\mathsf{O}(a \land \neg \mathsf{Y}(true))$	a
existence(a)	$\mathsf{O}(a)$	$\mathsf{F}(a)$
absence(a)	$\neg\mathsf{O}(a)$	$\neg\mathsf{F}(a)$
absence2(a)	$\mathsf{H}(a \rightarrow \mathsf{WYH}(\neg a))$	$\neg(\mathsf{F}a \land \mathsf{XF}(a))$
choice(a, b)	$\mathsf{O}(a) \lor \mathsf{O}(b)$	$\mathsf{F}(a) \lor \mathsf{F}(b)$
exclusive-choice(a, b)	$(\mathsf{O}(a) \lor \mathsf{O}(b)) \land \neg(\mathsf{O}(a) \land \mathsf{O}(b))$	$(\mathsf{F}(a) \lor \mathsf{F}(b)) \land \neg(\mathsf{F}(a) \land \mathsf{F}(b))$
co-existence(a, b)	$\mathsf{H}(\neg a) \leftrightarrow \mathsf{H}(\neg b)$	$\mathsf{F}(a) \leftrightarrow \mathsf{F}(b)$
responded-existence(a, b)	$\mathsf{O}(a) \rightarrow \mathsf{O}(b)$	$\mathsf{F}(a) \rightarrow \mathsf{F}(b)$
response(a, b)	$(\neg a\, \mathsf{S}\, b) \lor \mathsf{H}(\neg a)$	$\mathsf{G}(a \rightarrow \mathsf{F}(b))$
precedence(a, b)	$\mathsf{H}(b \rightarrow \mathsf{O}(a))$	$(\neg b\, \mathsf{U}\, a) \lor \mathsf{G}(\neg b)$
succession(a, b)	response$(a, b) \land$ precedence(a, b)	
chain-response(a, b)	$\mathsf{H}(\mathsf{Y}(a) \rightarrow b) \land \neg a$	$\mathsf{G}(a \rightarrow \mathsf{X}(b))$
chain-precedence(a, b)	$\mathsf{H}(b \rightarrow \mathsf{Y}(a))$	$\mathsf{G}(\mathsf{X}(b) \rightarrow a) \land \neg b$
chain-succession(a, b)	$(\mathsf{H}(\mathsf{Y}(a) \rightarrow b) \land \neg a) \land$ $\mathsf{H}(\mathsf{Y}(\neg a) \rightarrow \neg b)$	$\mathsf{G}(a \leftrightarrow \mathsf{X}(b))$
not-co-existence(a, b)	$\mathsf{O}(a) \rightarrow \neg\mathsf{O}(b)$	$\mathsf{F}(a) \rightarrow \neg\mathsf{F}(b)$
not-succession(a, b)	$\mathsf{H}(b \rightarrow \neg\mathsf{O}(a))$	$\mathsf{G}(a \rightarrow \neg\mathsf{F}(b))$
not-chain-succession(a, b)	$\mathsf{H}(b \rightarrow \neg\mathsf{Y}(a))$	$\mathsf{G}(a \rightarrow \neg\mathsf{X}(b))$



PDDL3 Operator	Equivalent PPLTL Formula	Equivalent LTL$_f$ Formula
(at-end θ)	θ	$\mathsf{F}(\theta \land \mathsf{final})$
(always θ)	$\mathsf{H}(\theta)$	$\mathsf{G}(\theta)$
(sometime θ)	$\mathsf{O}(\theta)$	$\mathsf{F}(\theta)$
(sometime-after θ_1 θ_2)	$(\neg\theta_1\, \mathsf{S}\, \theta_2) \lor \mathsf{H}(\neg\theta_1)$	$\mathsf{G}(\theta_1 \rightarrow \mathsf{F}(\theta_2))$
(sometime-before θ_1 θ_2)	$\mathsf{H}(\theta_1 \rightarrow \mathsf{Y}(\mathsf{O}(\theta_2)))$	$\theta_2\, \mathsf{R}\, \neg\theta_1$
(at-most-once θ)	$\mathsf{H}(\theta \rightarrow (\theta\, \mathsf{S}\, (\mathsf{H}(\neg\theta) \lor \mathsf{start})))$	$\mathsf{G}(\theta \rightarrow (\theta\, \mathsf{U}\, (\mathsf{G}(\neg\theta) \lor \mathsf{final})))$
(hold-during n_1 n_2 θ)	$\bigvee_{0 \le i \le n_1} (\theta \land \mathsf{Y}^i(\mathsf{start})) \lor$ $\bigwedge_{n_1 < i \le n_2} \mathsf{H}(\theta \lor \mathsf{WY}^i(\mathsf{Y}(true)))$	$\bigvee_{0 \le i \le n_1} \mathsf{X}^i(\theta \land \mathsf{final}) \lor$ $\bigwedge_{n_1 < i \le n_2} \mathsf{WX}^i(\theta)$
$*$ (hold-after n θ)	$\bigvee_{0 \le i \le n} (\theta \land \mathsf{Y}^i(\mathsf{start})) \lor$ $\mathsf{O}(\theta \land \mathsf{Y}^{n+1}(\mathsf{O}(\mathsf{start})))$	$\bigvee_{0 \le i \le n} \mathsf{X}^i(\theta \land \mathsf{final}) \lor$ $\mathsf{X}^{n+1}(\mathsf{F}(\theta))$

Handling PPLTL Goals

Intuition: given the prefix of a trace, while LTLf must consider all possible extensions, PPLTL can simply be evaluated on the prefix (i.e., the history produced so far)

How?

Exploit the "fixpoint characterization" of temporal formulas [Gabbay et al., 1980; Manna 1982; Barringer et al., 1989; Emerson 1990]

To evaluate a PPLTL formula:

1. Only two instants are sufficient (no need for the entire history)
2. We only need to keep track of the truth value of *some* of its subformulas!!!

$$\mathsf{pnf}(p) = p;$$

$$\mathsf{pnf}(\mathsf{Y}\phi) = \boxed{\mathsf{Y}\phi;}$$

$$\mathsf{pnf}(\phi_1\,\mathsf{S}\,\phi_2) = \mathsf{pnf}(\phi_2) \vee (\mathsf{pnf}(\phi_1) \wedge \boxed{\mathsf{Y}(\phi_1\,\mathsf{S}\,\phi_2)});$$

$$\mathsf{pnf}(\phi_1 \wedge \phi_2) = \mathsf{pnf}(\phi_1) \wedge \mathsf{pnf}(\phi_2);$$

$$\mathsf{pnf}(\neg\phi) = \neg\mathsf{pnf}(\phi).$$

Evaluating PPLTL Goals Using Planning

- The pnf tells us that we can evaluate a PPLTL formula by only considering the fluents true at the *current planning state* and which temporal subformulas held true at the *previous instant* (that has been already computed!!!)

How we do that?

1. Add variables to the planning state to keep track of the truth value of those temporal subformulas we are interested in
2. As the planning goes on, we iteratively update such variables in the planning state to reflect the satisfaction of the PPLTL goal formula
3. We stop when the PPLTL goal formula is entailed by the current planning state

Encoding PPLTL Goals in Planning Domains

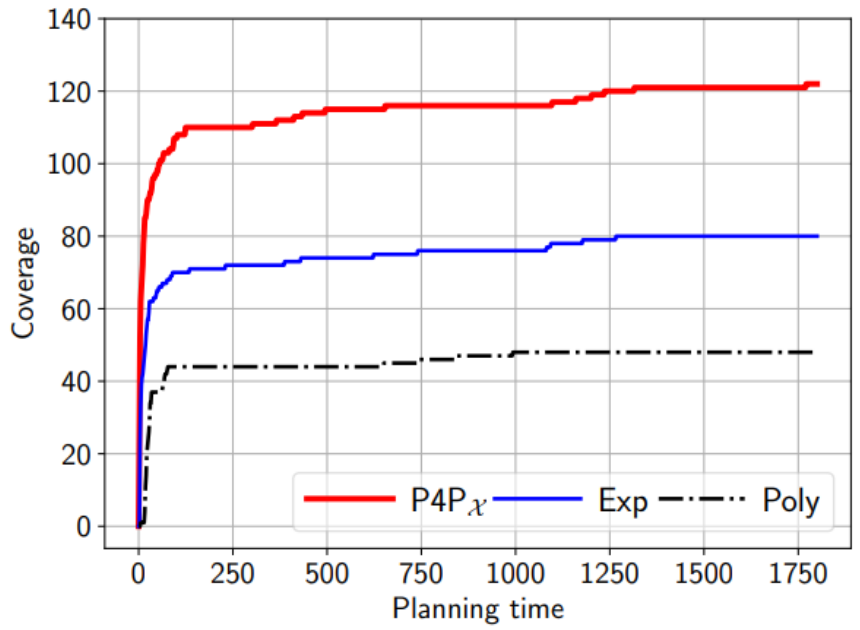
- Introduce only *few* new fluents, at most linear in the size of the PPLTL goal, i.e., minimal overhead
- No spurious additional actions
- Sidestep altogether the standard automata construction

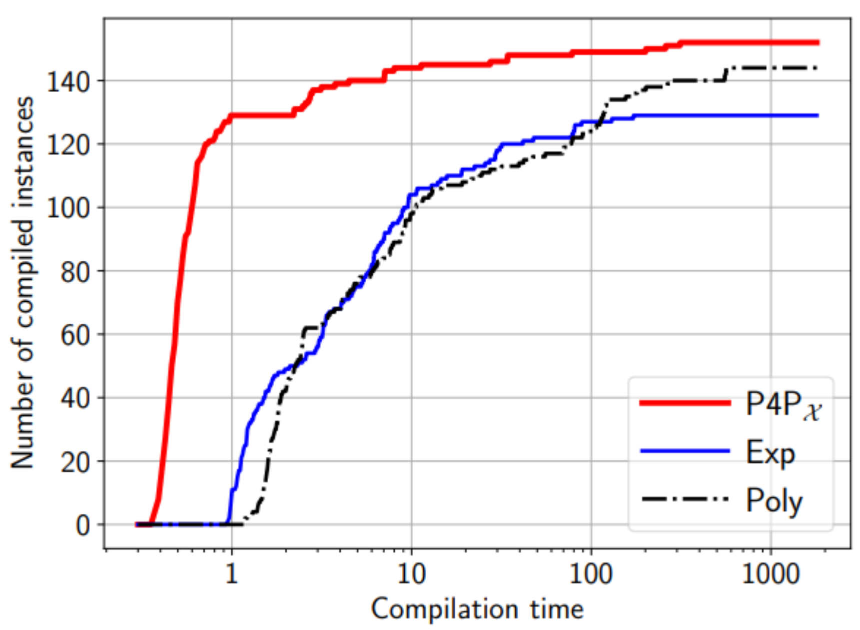
Components	Encoding
Fluents $\mathcal{F}'$	$\mathcal{F}' := \mathcal{F} \cup \Sigma_\varphi$
Derived Predicates $\mathcal{F}'_{der}$	$\mathcal{F}'_{der} := \mathcal{F}_{der} \cup \{\mathsf{val}_\phi \mid \phi \in \mathsf{sub}(\varphi)\}$
Axioms $\mathcal{X}'$	$\mathcal{X}' := \mathcal{X} \cup \{x_\phi \mid \phi \in \mathsf{sub}(\varphi)\}$ where x_ϕ is $\begin{cases} \mathsf{val}_p \leftarrow p & (\phi = p) \\ \mathsf{val}_{\mathsf{Y}\phi'} \leftarrow \text{“}\mathsf{Y}\phi'\text{”} & (\phi = \mathsf{Y}\phi') \\ \mathsf{val}_{\phi_1 \mathsf{S} \phi_2} \leftarrow (\mathsf{val}_{\phi_2} \vee (\mathsf{val}_{\phi_1} \wedge \text{“}\mathsf{Y}(\phi_1 \mathsf{S} \phi_2)\text{”})) & (\phi = \phi_1 \mathsf{S} \phi_2) \\ \mathsf{val}_{\phi_1 \wedge \phi_2} \leftarrow (\mathsf{val}_{\phi_1} \wedge \mathsf{val}_{\phi_2}) & (\phi = \phi_1 \wedge \phi_2) \\ \mathsf{val}_{\neg\phi'} \leftarrow \neg\mathsf{val}_{\phi'} & (\phi = \neg\phi') \end{cases}$
Action Labels A	$A := A$, i.e., unchanged
Preconditions pre	$pre(a) := pre(a)$ for every $a \in A$, i.e., unchanged
Effects eff'	$eff'(a) := \mathsf{eff}(a) \cup \mathsf{eff}_{\mathsf{val}}$, where $\mathsf{eff}_{\mathsf{val}} = \{\mathsf{val}_\phi \rhd \{\text{“}\mathsf{Y}\phi\text{”}\}, \neg\mathsf{val}_\phi \rhd \{\neg\text{“}\mathsf{Y}\phi\text{”}\} \mid \text{“}\mathsf{Y}\phi\text{”} \in \Sigma_\varphi\}$
Initial State s'_0	$s'_0 := \sigma_0 \cup s_0$
Goal G'	$G' := \mathsf{val}_\varphi$

Sound and complete approach to symbolically encode PPLTL goal formulas in planning domains that is polynomial in the size of the domain specification and linear in the size of the PPLTL goal

Experimental Results

- Introduce the Plan4Past[1] (P4P) system (https://github.com/whitemech/Plan4Past)
- Compare Plan4Past against state-of-the-art techniques for LTLf, “Exp” [Baier & McIlraith 2006] and “Poly” [Torres & Baier 2015], on a set of equivalent (semantic- and size-wise) and challenging LTLf/PPLTL formulas
- IPC domains: BLOCKS, ELEVATOR, OPENSTACKS, ROVERS





Summary and Future Work

- How PPLTL can specify planning temporally extended goals
- How to efficiently handle and evaluate PPLTL formulas
- Sound and complete approach to solve planning for PPLTL goals that is optimal wrt theoretical complexity with a clear advantage in practice

Future work

- Developing PPLTL-aware heuristics that exploit the structure of the formula
- Incorporate PPLTL patterns directly into PDDL, envisioning an extension of PDDL3