On the Expressive Power of Planning Formalisms in Conjunction with LTL

Songtuan Lin & Pascal Bercher

College of Engineering & Computer Science, The Australian National University

{firstName.lastName@anu.edu.au}

Introduction

Objective: The objective of this paper is to study the expressiveness of various hierarchical and non-hierarchical planning formalisms in conjunction with Linear Temporal Logic (LTL).
Method: The approach we consider for this purpose is viewing the solution set of a planning problem as a formal language and compare it with other formal ones.

LTL and Finite LTL

LTL: The syntax of an LTL formula φ is defined as follows:

$$\varphi = \top \mid p \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \bigcirc\varphi \mid \varphi_1 \,\mathtt{U}\, \varphi_2$$

The semantics of LTL is defined in terms of a state sequence of infinite length: $\pi = \langle s_1\ s_2 \cdots \rangle$. We denote $\pi[\![i]\!] = \langle s_i \cdots \rangle$.

- $\pi[\![i]\!] \vDash \top$
- $\pi[\![i]\!] \vDash p$ *iff* $p \in s_i$
- $\pi[\![i]\!] \vDash \neg\varphi$ *iff* $\pi[\![i]\!] \nvDash \varphi$
- $\pi[\![i]\!] \vDash \bigcirc\varphi$ *iff* $\pi[\![i+1]\!] \vDash \varphi$
- $\pi[\![i]\!] \vDash \varphi_1 \wedge \varphi_2$ *iff* $\pi[\![i]\!] \vDash \varphi_1 \wedge \pi[\![i]\!] \vDash \varphi_2$
- $\pi[\![i]\!] \vDash \varphi_1 \,\mathtt{U}\, \varphi_2$ *iff* there exists a $j \geq i$ such that $\pi[\![j]\!] \vDash \varphi_2$ and $\pi[\![k]\!] \vDash \varphi_1$ for all $i \leq k < j$.

Finite LTL: The syntax of f-LTL is identical to that of LTL, but the semantics is defined in terms of a finite state sequence $\pi = \langle s_1 \cdots s_n \rangle$:

- $\pi[\![i]\!] \vDash \top$
- $\pi[\![i]\!] \vDash p$ *iff* $p \in s_i$
- $\pi[\![i]\!] \vDash \neg\varphi$ *iff* $\pi[\![i]\!] \nvDash \varphi$
- $\pi_i \vDash \bigcirc\varphi$ *iff* $i < n$ and $\pi_{i+1} \vDash \varphi$
- $\pi[\![i]\!] \vDash \varphi_1 \wedge \varphi_2$ *iff* $\pi[\![i]\!] \vDash \varphi_1 \wedge \pi[\![i]\!] \vDash \varphi_2$
- $\pi_i \vDash \varphi_1 \mathtt{U} \varphi_2$ *iff* there exists a j with $i \leq j \leq n$ such that $\pi_j \vDash \varphi_2$, and for each $i \leq k < j$, $\pi_k \vDash \varphi_1$

One crucial power of f-LTL is to express *the end of a state sequence*, written $\odot$, in terms of the operator $\bigcirc$:

$$\odot = \bigcirc(\neg\top)$$

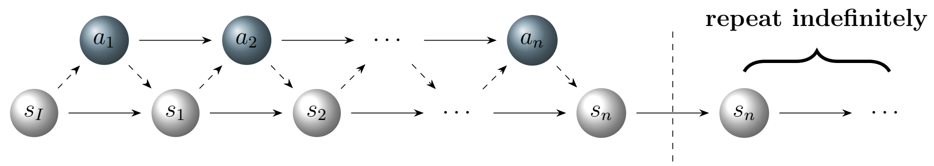
More concretely, we have that $\pi[\![i]\!] \vDash \odot$ *iff* $i = n$.

Non-hierarchical Planning Formalism

A $\mathcal{STRIPS}$ planning problem $\mathcal{P}$ is a tuple $\mathcal{P} = (\mathcal{F}, \mathcal{A}, \delta, s_I, g)$:

- $\mathcal{F}$: A set of propositions
- $\mathcal{A}$: A set of actions
- g: $g \subseteq \mathcal{F}$
- s_I: $s_I \in 2^{\mathcal{F}}$
- δ: $\mathcal{A} \to 2^{\mathcal{F}} \times 2^{\mathcal{F}} \times 2^{\mathcal{F}}$ – $\delta(a) = (prec(a), \mathit{eff}^+(a), \mathit{eff}^-(a))$

A solution to $\mathcal{P}$ is an action sequence $\overline{a} = \langle a_1 \cdots a_n \rangle$ which results in a state sequence $\pi = \langle s_0 \cdots s_n \rangle$ such that $s_0 = s_I$, $g \subseteq s_n$, and for each $1 \leq i \leq n$, $prec(a_i) \subseteq s_i$ and $s_i = (s_{i-1} \setminus \mathit{eff}^-(a_i)) \cup \mathit{eff}^+(a_i)$.



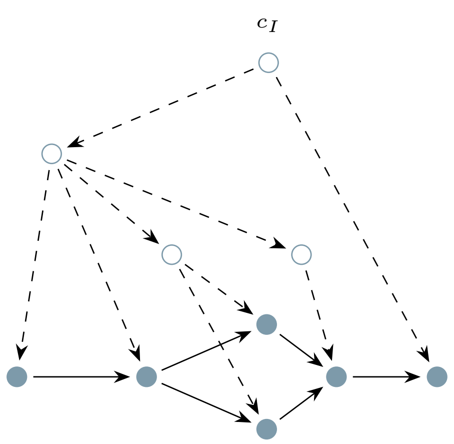
A $\mathcal{STRIPS}$-$\mathcal{L}$ or a $\mathcal{STRIPS}$-$\mathcal{FL}$ planning problem $\mathcal{P}$ is a tuple $\mathcal{P} = (\mathcal{F}, \mathcal{A}, \delta, s_I, g)$ where g is respectively an LTL or an f-LTL formula.

A solution to a $\mathcal{STRIPS}$-$\mathcal{L}$ or a $\mathcal{STRIPS}$-$\mathcal{FL}$ problem is an action sequence $\overline{a}$ which results in a state sequence π with $\pi[\![0]\!] \vDash g$.

Remark: For a $\mathcal{STRIPS}$-$\mathcal{L}$ problem, since the semantics of LTL is defined over an infinite state sequence, we have to **artificially** extend π to infinite by repeating its last state indefinitely (see the figure).

Hierarchical Planning Formalism

An $\mathcal{HTN}$ planning problem is $\mathcal{P} = ((\mathcal{F}, \mathcal{A}, \mathcal{C}, \mathcal{M}, \delta), c_I, g)$ where $\mathcal{C}$ is a set of compound tasks, and $\mathcal{M}$ is a set of methods.



A compound task is decomposed into a partial order set of actions and compound tasks called task network by a method.
A solution is a task network consisting solely of actions which is obtained from the initial compound task and has an executable linearisation resulting in a state sequence π satisfying g.

We can incorporate LTL and f-LTL into HTN planning formalism by replacing g with a respective LTL or f-LTL formula.

Languages of Planning Problems

The language of a **non-hierarchical** planning problem $\mathcal{P}$:

$$\mathcal{L}(\mathcal{P}) = \{\omega \mid \omega \text{ is a solution to } \mathcal{P}\}$$

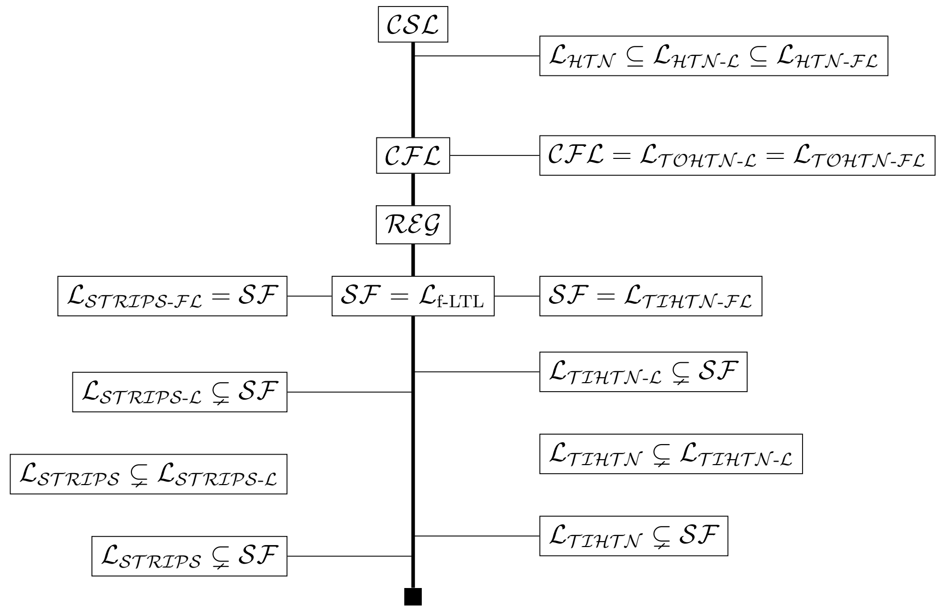
The language of a **hierarchical** planning problem $\mathcal{P}$:

$$\mathcal{L}(\mathcal{P}) = \left\{\pi \;\middle|\; \begin{array}{l} \pi \text{ is an executable linearization of } tn, \\ tn \text{ is a solution to } \mathcal{P} \end{array}\right\}$$

The class of languages of a (hierarchical or non-hierarchical) planning formalism X, e.g., $X = \mathcal{STRIPS}$-$\mathcal{FL}$:

$$\mathcal{L}_X = \{\mathcal{L}(\mathcal{P}) \mid \mathcal{P} \text{ is a planning problem in the formalism } X\}$$

Results and Interpretation



- Incorporating LTL and f-LTL into the $\mathcal{STRIPS}$ formalism **increases** its expressiveness. In particular:

  $$\mathcal{L}_{\mathcal{STRIPS}} \subsetneq \mathcal{L}_{\mathcal{STRIPS}\text{-}\mathcal{L}} \subsetneq \mathcal{L}_{\mathcal{STRIPS}\text{-}\mathcal{FL}} = \mathcal{SF} \subsetneq \mathcal{REG}$$

  where $\mathcal{SF}$ and $\mathcal{REG}$ refer to the star-free languages and regular languages, respectively.

- Incorporating LTL and f-LTL into $\mathcal{TIHTN}$ also **increases** its expressiveness. In particular:

  $$\mathcal{L}_{\mathcal{TIHTN}} \subsetneq \mathcal{L}_{\mathcal{TIHTN}\text{-}\mathcal{L}} \subsetneq \mathcal{L}_{\mathcal{TIHTN}\text{-}\mathcal{FL}} = \mathcal{SF}$$

  where $\mathcal{TIHTN}$ refers to $\mathcal{HTN}$ planning with task insertions.

- Incorporating LTL and f-LTL into $\mathcal{TOHTN}$ (total order $\mathcal{HTN}$ planning) does **not** increase its expressiveness. They are all equivalent to context-free languages ($\mathcal{CFL}$):

  $$\mathcal{L}_{\mathcal{TOHTN}} = \mathcal{L}_{\mathcal{TOHTN}\text{-}\mathcal{L}} = \mathcal{L}_{\mathcal{TOHTN}\text{-}\mathcal{FL}} = \mathcal{CFL}$$

- All formalisms are below context-sensitive languages ($\mathcal{CSL}$):

  $$\mathcal{L}_{\mathcal{HTN}} \subseteq \mathcal{L}_{\mathcal{HTN}\text{-}\mathcal{L}} \subseteq \mathcal{L}_{\mathcal{HTN}\text{-}\mathcal{FL}} \subseteq \mathcal{CSL}$$