LTL and Past LTL on Finite Traces for Planning and Declarative Process Mining

Francesco Fuggitti

Master of Science in
Engineering in Computer Science
Sapienza, University of Rome

Advisor: Prof. Giuseppe De Giacomo

A.Y. 2017/2018

Outline

Introduction

- Classic Reinforcement Learning:
  - An *agent* interacts with an *environment* by taking *actions* so to maximize *rewards*;
  - No knowledge about the transition model, but assume Markov property (history does not matter): Markov Decision Process (MDP)
  - Solution: (Markovian) policy $\rho : S \to A$

- RL for Non-Markovian Decision Process (NMRDP):
  - Rewards depend from history, not just the last transition;
  - Specify proper behaviours by using temporal logic formulas;
  - Solution: (Non-Markovian) policy $\rho : S^* \to A$
  - Reduce the problem to MDP (with extended state space)

- In (Brafman et al. 2018) specify reward using:
  - Linear-time Temporal Logic on Finite Traces LTL$_f$
  - Linear-time Dynamic Logic on Finite Traces LDL$_f$

Objectives

- Provide an efficient technique to transform LTL$_f$/PLTL formulas into DFAs

- Provide an approach to FOND Planning for LTL$_f$/PLTL goals:
  - reduce the problem to standard FOND planning
  - working with FOND domains instead of automata

- Provide a generalization of the Janus approach to declarative process mining:
  - generalization of the constraint formula

PLTL and LTL$_f$ (De Giacomo and Vardi, 2013)

- Linear Temporal Logic on finite traces: LTL$_f$
  - exactly the same syntax of LTL
  - interpreted over *finite* traces
    - next: $\bigcirc$*happy*
    - until: *reply* $\mathcal{U}$ *acknowledge*
    - eventually: $\Diamond$*rich*
    - always: $\Box$*safe*
- Past Linear Temporal Logic: PLTL
  - same syntax of LTL$_f$, but looks into the past
    - yesterday: $\ominus$*happy*
    - since: *reply* $\mathcal{S}$ *acknowledge*
    - once: $\Diamond\!\!\!-$*rich*
    - hystorically: $\boxminus$*safe*
- Reasoning in LTL$_f$/PLTL:
  - transform formulas φ into DFAs $\mathcal{A}_\varphi$
  - for every trace π an LTL$_f$/PLTL formula φ:

$$\pi \models \varphi \iff \pi \in \mathcal{L}(\mathcal{A}_\varphi)$$

Translation of LTL$_f$ and PLTL formulas to DFA

- Translation Procedure
  1. starting from an LTL$_f$/PLTL formula φ we translate it to FOL on finite sequences (De Giacomo and Vardi 2013; Zhu et al. 2018)
  2. apply the highly optimized tool MONA able to transform Monadic Second Order Logic (and hence FOL as well) on finite strings to DFA automata

- Example: $\varphi = \Diamond G$
  1. FOL translation: $fol(\varphi, x) = \exists y.x \leq y \leq last \wedge G(y)$, where $[x/0]$
  2. MONA program: `m2l-str; var2 G; ex1 y:  0<=y & y<=max($) & y in G`

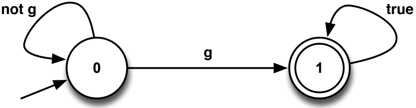
The implementation of the translation procedure: LTL$_f$2DFA

Python package supporting:

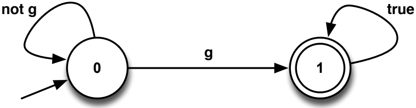
- LTL$_f$/PLTL formulas parsing
- translation to FOL, DFA
- option for DECLARE assumption (De Giacomo et al. 2014)

Output examples:

$\varphi_1 = \Box(a \Rightarrow b)$



$\varphi_2 = \Diamond g$



FOND Planning for Extended Temporal Goals

- A *non-deterministic* domain with initial state is a tuple $\mathcal{D} = \langle 2^{\mathcal{F}}, A, s_0, \partial, \alpha \rangle$, specified in PDDL
- Who controls what?
  - Fluents controlled by the environment
  - Actions controlled by the agent
- Goals, planning and plans
  - Goal: an LTL_f/PLTL formula φ
  - Planning: a *game* between two players
  - Plan: *strategy* to *win* the game

The FOND4LTL$_f$/PLTL approach: Given a PDDL domain $\mathcal{D}$, the initial state s_0 and a goal formula φ, we obtain $\mathcal{A}_\varphi = \langle \Sigma, Q, q_0, \delta, F \rangle$ through LTL$_f$2DFA and we encode $\mathcal{A}_\varphi$ into $\mathcal{D}$

- We need to perform an action on the domain and then update the automaton state. It is achieved through the *turnDomain* predicate
- Encoding of automaton $\mathcal{A}_\varphi$ to PDDL in domain $\mathcal{D}$:

Action `trans`

parameters: $(x_0, \ldots, x_k)$, where $x_i \in \mathcal{V}$
preconditions: $\neg turnDomain$
effects:
when $(q_i(x_0, \ldots, x_k) \wedge a'_j)$ *then* $(\delta'(q'_i, a'_j) = q''_i(x_0, \ldots, x_k) \wedge (\neg q, \forall q \in Q \text{ s.t. } q \neq q''_i) \wedge turnDomain)$, $\forall i, j : 0 \leq i \leq m, 0 \leq j \leq n$

- New initial state: $s_0 \wedge \texttt{turnDomain} \wedge q_0(o_0, \ldots, o_k)$
- New goal specification: $\texttt{turnDomain} \wedge (\bigvee_{q \in F} q(o_0, \ldots, o_k))$

Results: the Triangle Tireworld example

- Objective: Drive from one location to another. A tire may be going flat. If there is a spare tire in the location of the car, then the car can use it to fix the flat tire.
- Goal: $\varphi = \mathit{vehicleAt}(l13) \wedge \diamondminus(\mathit{vehicleAt}(l23))$
- Result:

The Janus approach for Declarative Process Mining

- Declarative Process Mining is the set of techniques aimed at extracting and validating temporal constraints out of event logs (i.e. multi-sets of *finite* traces). The Janus approach (Cecconi et al. 2018) allows to compute the *interestingness degree* of traces in event logs
- Determine if a trace is "relevant" or "interesting" wrt a given temporal specification

Example

- Given a trace $t = \langle d, f, a, f, c, a, f, b, a, f \rangle$
- Given a constraint $\Psi = a \mapsto (\ominus b \vee \Diamond c)$, where a is the activation condition and $(\ominus b \vee \Diamond c)$ is the formula to be validated
- Result: the *interestingness degree* is $\dfrac{2}{3} = 0.667$

Our generalization of the Janus approach extends the original Janus approach in the following terms:

- provide a new constraint representation:

$$\Psi \doteq \bigvee_{j=1}^{m} (\varphi^{\blacktriangleleft} \wedge \varphi^{\blacktriangledown} \wedge \varphi^{\blacktriangleright})_j \tag{1}$$

  - $\varphi^{\blacktriangleleft}$ a formula on the past that checks if the past makes the triggering relevant
  - $\varphi^{\blacktriangledown}$ a propositional formula on the current instant that triggers potential interest on the instant itself
  - $\varphi^{\blacktriangleright}$ a formula on the future that must be satisfied for considering the current instant interesting

Conclusions

Thesis results:

- Extended (Brafman et al. 2018) to the context of RL;
- Formalization of *RL for* $\text{LTL}_f/\text{LDL}_f$ *goals*, devising of a solution and analysis of the advantages;
- Provided an implementation and given experimental evidence of the goodness of our approach.

Future works:

- Minimal low-level representation to tackle $\text{LTL}_f/\text{LDL}_f$ goals;
- Optimize FLLOAT, enrich RLTG;
- Try the approach in several real world applications;
- Design of ad-hoc algorithms (e.g. "automata-aware" exploration);
- Extend the approach to the framework of Multi-Agent Systems.

Publication: https://arxiv.org/abs/1807.06333. Status: submitted