Reasoning about Action and Planning in LTL action Theories

Diego Calvanese, Giuseppe De Giacomo, Moshe Y. Vardi

Alessio Antonini, Giuseppe Rizzo, Manuel Rodriguez

January 30th, 2014

Introduction

This paper is about actions and planning with incomplete information in a dynamic system represented with Linear Temporal Logic (LTL) and Quantified Linear Temporal Logic (QLTL).

Introduction

The paper addresses three main issues:

i reasoning about action effects (i.e., projection, historical queries)

ii legally execution of actions in a given situation

iii finding conformant plans for temporally extended goals

For each of these problems authors present techniques and characterize the computational complexity.

Introduction

Settings

- The system is described with a set of atomic facts (fluents)
- Fluents have a truth value which change as result of actions
- The behavior of the system as a set of sequences of situations
- Actions cause transitions from one situation to another

Linear Temporal Logic

Introduction

Linear Temporal Logic (LTL) is a linear-time temporal logic used for *specifying and verifying properties* of dynamic systems, such as:

- safety,
- liveness,
- fairness,
- etc.

Linear Temporal Logic

Language

Formulas of LTL are built from a set $\mathcal{P}$ of propositional symbols and are closed under the boolean operators:

- the unary temporal operators $\bigcirc$, $\Box$ and $\Diamond$
- the binary temporal operator $\mathcal{U}$

Linear Temporal Logic

Operators

$\circ\varphi \doteq \varphi$ holds at the next instant

$\diamond\varphi \doteq \varphi$ will eventually hold at some future instant

$\Box\varphi \doteq$ from the current instant on φ will always hold

$\varphi\mathcal{U}\psi \doteq$ at some future instant ψ will hold and until that point φ holds

$\vee \doteq$ or

$\rightarrow \doteq$ implies

$\diamond\varphi \doteq$ true $\mathcal{U}\varphi$

$\Box\varphi \doteq \neg\Diamond\neg\varphi$

Linear Temporal Logic

Semantic

- The semantics of LTL is given in terms of interpretations over a linear structure $\mathbb{N}$. Given an instant $i \in \mathbb{N}$, the successive instant is $i + 1$.
- An interpretation is a function $\pi : \mathbb{N} \rightarrow 2^{\mathcal{P}}$, which assigns to each element of $\mathcal{P}$ a truth value at each instant $i \in \mathbb{N}$.

Linear Temporal Logic

Semantic

For an interpretation π is inductively defined when an LTL formula φ is true at an instant $i \in \mathbb{N}$, in symbols $\pi, i \models \varphi$, as follows:

$$\pi, i \models p, p \in \mathcal{P} \text{ iff } p \in \pi(i)$$

$$\pi, i \models \neg\varphi \text{ iff } \neg\pi, i \models \varphi$$

$$\pi, i \models \varphi \wedge \varphi' \text{ iff } \pi, i \models \varphi \text{ and } \pi, i \models \varphi'$$

$$\pi, i \models \circ\varphi \text{ iff } \pi, i + 1 \models \varphi$$

$$\pi, i \models \varphi\mathcal{U}\varphi' \text{ iff for some } j >= i \text{ we have}$$

$$\pi, j \models \varphi' \text{ and for all } k, i <= k < j, \text{ we have that}$$

$$\pi, k \models \varphi$$

A formula φ is *true* in π in notation $\pi \models \varphi$, if $\pi, 0 \models \varphi$. A formula φ is *satisfable* if it is true in some interpretation, and is *valid*, if it is true in every interpretation.

Linear Temporal Logic
Büchi automata

LTL is connected to Buchi automata.

- Given $\mathcal{P}$, the set of interpretations $2^{\mathcal{P}}$ of the propositional variables in $\mathcal{P}$ can be considered the alphabet of a Büchi automaton.
- An infinite word over the alphabet $2^{\mathcal{P}}$ accepted by an automaton can be viewed as an interpretation of an LTL formula over $\mathcal{P}$.

Linear Temporal Logic

Büchi automata

LTL formula

- Represents a set of infinite traces which satisfy such formula

Büchi Automaton

- Accepts a set of infinite traces

We can build an automaton which accepts all and only the infinite traces represented by an LTL formula.

Linear Temporal Logic

Büchi automata

Given a finite non empty alphabet Σ, an *infinite word* is an element of Σ^ω. A Büchi automaton is a tuple $\mathcal{A} = (\Sigma, \mathcal{S}, \mathcal{S}_0, \rho, \mathcal{F})$ where:

- Σ is the alphabet of the automaton
- $\mathcal{S}$ is the set of finite states
- $\mathcal{S}_0$ is the initial state
- $\rho : \mathcal{S} \times \Sigma \to 2^{\mathcal{S}}$ is the transition function
- $\mathcal{F} \subseteq \mathcal{S}$

The *input words* of $\mathcal{A}$ are infinite words $a_0, a_1, \ldots \in \Sigma^\omega$.

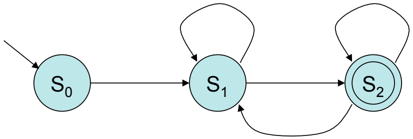
Linear Temporal Logic

Büchi automata

Theorem [1]: for every LTL formula φ one can effectively construct a Büchi automaton A_φ whose number of states is at most exponential in the lenght of φ and such that $L(A_\varphi)$ is the set of models of φ.

Linear Temporal Logic

Büchi automata



$\sigma_1 = S_0 S_1 S_2 S_2 S_2 S_2 \ldots$ **ACCEPTED**

$\sigma_2 = S_0 S_1 S_2 S_1 S_2 S_1 \ldots$ **ACCEPTED**

$\sigma_3 = S_0 S_1 S_2 S_1 S_1 S_1 \ldots$ **REJECTED**

Linear temporal logic

Quantified linear temporal logic

Quantified Linear Temporal Logic (QLTL) is an extention of LTL. QLTL formulas are LTL forumulas plus the existential quantifier of propositions

- $\exists p.\varphi(p)$ where p is a proposition variable and $\varphi(p)$ a QLTL formula in which p occours free
- $\forall p.\varphi(p) = \neg\exists p.\varphi(p)$

The semantic is defined as follows:

- $\pi, i \models \exists p.\varphi$ iff there is some π' that agrees with π exept for the interpretation of proposition p, and such that $\pi', i \models \varphi$.

Linear Temporal Logic

Quantified linear temporal logic

Theorem [2]: satisfability for Σ_{k+1}^{QLTL} formulas [...], with $k >= 1$ is *k-EXSPACE-complete*.

Reasoning about Actions in LTL

General considerations

The behavior of a dynamic system is characterized by a set of evolutions (i.e. sequence of situations)

- Under incomplete information, only a set of possible evolutions can be isolated
- Dynamic system are specified by:
  - a set of fluents $\mathcal{F}$, facts concerning the current situation
  - a set of actions $\mathcal{A}$, the effects of actions on such a set of facts

Reasoning about Actions in LTL

Structural requirements

A dynamic system can be described as a conjunction of a finite set of LTL formulas.

- add the following LTL formulas to model actions:
  - $\square(\bigvee_{a \in \mathcal{A}} a)$ i.e. an action can be performed at a time
  - $\square(\bigwedge_{a \in \mathcal{A}} (a \rightarrow \bigwedge_{b \in \mathcal{A}, b \neq a} \neg b))$ i.e. an action must be performed

In this way an unique proposition $a \in \mathcal{A}$ holds in each situation and it specifies the action performed to get to that situation

Reasoning about Actions in LTL

Specifying effects

In order to specify the effects of an action we need to:

- describe an initial situation by means of a formula φ_{init} that involves only fluents
- describe effects of actions by means of formulas of the form

$$\Box(\varphi \rightarrow (\circ a \rightarrow \psi))$$

  where φ and ψ are LTL formulas which involves only fluents
- some contraints can be specified as follows

$$\Box(\phi)$$

Reasoning about Actions in LTL

Specifying effects

- Using this formalization, there is uncertainty about the initial situation and the effects of the actions
- Complete specification using Reiter's successor state axioms

$$\circ F \equiv \bigvee_{a \in \mathcal{A}} ((\varphi_a^+ \wedge \circ a) \vee (F \wedge \bigwedge(\neg\varphi_b^- \vee \circ b))$$

where F are fluents, a's are action that make F become true under situations described by φ_a^+ and b's are actions that makes F become false under φ_b^-

Reasoning about Actions in LTL

Reasoning about Actions Effects in LTL

- Problem: given a finite sequence of action, determines whether a certain property holds
  - Projection problem: does the property ϕ hold after the execution of the sequence of actions $a_0 \cdots a_k$?
  - Historical queries:does the property ϕ always hold (resp. hold at some point) over the duration of the sequence of actions $a_0 \cdots a_k$?
- The problem is P-SPACE COMPLETE (due to validity of LTL formula)

Reasoning about Actions in LTL

Reasoning about Actions Effects in LTL

- The following formula needs to be introduced

$$\begin{aligned} & Occurs(a_0, \cdots, a_k) \\ \doteq\ & (a_0 \wedge \circ(a_1 \wedge \circ(\cdots \circ (a_k \wedge rs)\cdots))) \wedge \Box(rs \rightarrow \circ \Box \neg rs) \end{aligned}$$

  it represents that the execution of $a_0, \cdots, a_k$ results in a situation rs that will be true once

Reasoning about Actions in LTL

Reasoning about Actions Effects in LTL

- Given a system description Γ:
  - Projection problem can be solved checking that:

$$\Gamma \rightarrow (Occurs(a_0 \cdots a_k, rs) \rightarrow \Box(rs \rightarrow \phi))$$

  - Historical queries can be solved checking that:

$$\Gamma \rightarrow (Occurs(a_0 \cdots a_k, rs) \rightarrow \Box(\Diamond(rs) \rightarrow \phi))$$

$$\Gamma \rightarrow (Occurs(a_0 \cdots a_k, rs) \rightarrow \Box(\varphi \wedge \Diamond(rs)))$$

Reasoning about Actions in LTL

Legal sequencies action

- Various formalism for reasoning about actions use a prescriptive approach has been proposed
    - specifying the circumstances for the execution
- Instead, LTL uses a non-prescriptive approach
    - it is always possible to execute an action, unless it contradicts the system specification
    - due to the partial knowledge ensuring the executability of an action is needed

Reasoning about Actions in LTL

Legal sequencies action: dealing with uncomplete knowledge

- principle of directionality : information about a given time is deductively independent from information about a later time
- Instead, LTL introduces the notion of legality of a sequence of actions w.r.t. Γ

Reasoning about Actions in LTL

Legal sequencies action: consistencies and legality

- Consistency: An infinite sequence of actions $a_0, a_1, \cdots$ is consistent with Γ if there exists a model of Γ whose interpretation of the actions coincides with $a_0, a_1, \cdots$.
- Legality: an action a_{k+1} is legal after the sequence of actions $a0, \cdots, a_k$ if for all sequences $\sigma_0, \cdots, \sigma_k$ of truth assignments to the fluents in $\mathcal{F}$, if $a0, \cdots, a_k$ and $\sigma_0, \cdots, \sigma_k$ are consistent with Γ, then also $a0, \cdots, a_k, a_{k+1}$ and $\sigma_0, \cdots, \sigma_k$ are consistent with Γ

Reasoning about Actions in LTL

Legality in QLTL

- Legality can be captured in QLTL
- Let now a proposition acts as marker and $\vec{x}$ and $\vec{y}$ are tuples of variables, one for each fluent, and $\vec{x} \equiv \vec{y}$ stands for the conjunction of equivalences among corresponding components
- The following QLTL formulas can be defined:

$$Point(now) \doteq \diamond(now) \wedge \Box(now \rightarrow \circ \Box \neg now)$$

$$EqUntil(\vec{x}, \vec{y}, now) \doteq \Box((\circ now) \rightarrow \vec{x} \equiv \vec{y})$$

$$EqNext(\vec{x}, \vec{y}, now) \doteq \Box(now \rightarrow \circ(\vec{x} \equiv \vec{y}))$$

Reasoning about Actions in LTL

Legality in QLTL

- $\mathit{Point}(\mathit{now})$ expresses that now holds at a single time point
- $\mathit{EqUntil}(\vec{x}, \vec{y}, \mathit{now})$ expresses that $\vec{x}$ and $\vec{y}$ coincide at every time point until now holds.
- $\mathit{EqNext}(\vec{x}, \vec{y}, \mathit{now})$ expresses that $\vec{x}$ and $\vec{y}$ coincide at the time point following the one where now holds.

Reasoning about Actions in LTL
Legality in QLTL

- Given a system specification $\Gamma(\vec{a}, \vec{f})$ the legality can be defined as:

$$\begin{aligned} \mathit{LegalNext}(\Gamma, \vec{a}, \mathit{now}) \doteq{}& \mathit{Point}(\mathit{now}) \\ & \wedge \forall \vec{a}_1 \forall \vec{f}_1 \Gamma(\vec{a}_1, \vec{f}_1) \wedge \mathit{EqUntil}(\vec{a}_1, \vec{a}, \mathit{now}) \\ & \rightarrow \exists \vec{a}_2 \exists \vec{f}_2 \Gamma(\vec{a}_2, \vec{f}_2) \wedge \mathit{EqUntil}(\vec{a}_2, \vec{a}_1, \mathit{now}) \\ & \wedge \mathit{EqUntil}(\vec{f}_2, \vec{f}_1, \mathit{now}) \wedge \mathit{EqNext}(\vec{a}_2, \vec{a}, \mathit{now}) \end{aligned}$$

i.e. it exists a further interpretation in accord with the actions and the fluents of another one till now, in which the action performed next is the one selected by $\vec{a}$.

Reasoning about Actions in LTL
Legality in QLTL

- Legality for an infinite sequence of actions can be defined as:

$$Legal(\Gamma, \vec{a}) \doteq \exists \vec{f} \Gamma(\vec{a}, \vec{f}) \wedge \forall now\ LegalNext(\Gamma, \vec{a}, now)$$

i.e. the sequence of actions resulting from the interpretation of $\vec{a}$ is consistent with Γ and every prefix of such a sequence continues next with a legal action

Reasoning about Actions in LTL
Complexity

- An infinite sequence of actions $a_0, a_1 \cdots$ is legal wrt an LTL system specification Γ iff there exists an interpretation π interpreting the actions $\vec{a}$ according to $a_0, a_1 \cdots$ and such that $\pi \models \mathit{Legal}(\Gamma, \vec{a})$
- Checking the existence of an infinite sequence of actions that is legal wrt an LTL system specification Γ can be done in 2-EXPSPACE.
  - it is a 2-EXPSPACE-hard problem

Planning

General considerations

- Dynamic properties of LTL, can be verified writting the system specification and used as temporally extended goals to synthetize plans.
- For example:
  - Reachability of the desire state of affairs, is a situation where a given goal φ_{goal} holds reachable? $\diamond\varphi_{\text{goal}}$.
  - Archiving and maintenance of goals, is it possible to archive a goal φ_{goal} while maintaining another goal φ'_{goal}? $\varphi'_{\text{goal}}\mathcal{U}\varphi_{\text{goal}}$
- Goals can be more sophisticated and similarly, safety, invariance, liveness and fairness properties can be expressed in LTL.

Planning

Conformant Planning

Conformant planning, is the way in the setting for specify by arbitrary formulas of LTL both the Dynamic system and the goal.
The problem consist in constructing a plan.

$$plan(\Gamma, \gamma, \vec{a}) \doteq \forall \vec{f}. \Gamma(\vec{a}, \vec{f}) \rightarrow \gamma(\vec{a}, \vec{f})$$

A plan can be infinite but still finitely representable.

Planning

Conformant Planning

To check the existence of a lega plan, we simple need to check the satisfability of

$$\mathit{plan}(\Gamma, \gamma, \vec{a}) \wedge \mathit{valid}(\Gamma, \vec{a})$$

Thi can be done building a Büchi automaton and checking the nonemptiness.
Theorem[3]: Verifing the validity of a plan in LTL is *2-EXPACE-complete*.

Conclusions

- Those results can be exteded to other LTL extensions such as μ LTL that is a translation of LTL to Buchi automata that is able to represent any property like safety, invariance, liveness and fairness
- Adopting the techniques discussed in this paper, it can be shown:
  - that the reasoning on action effects remains PSPACE-complete
  - synthesing nonnecessarily legal plans remains EXPSPACE-complete
  - while testing legality becomes EXPSPACE-complete

Conclusions

Considerations

- Complexity is probably going to increase, by moving to the branching-time setting.
- One should stress that for system specifications of a special form, checking legality of sequences of actions may become much easier.

Conclusions

Finally

Finally algorithms for cheking nonemptiness of Buchi automata, which are at the base of reasoning procedures for LTL and QLTL, have proved to be well suites for scalling up to very large Systems.