Department of Mathematics, Computer Science and Physics, University of Udine

The Safety Fragment of Temporal Logics on Infinite Sequences

Lesson 15

Luca Geatti
luca.geatti@uniud.it
Angelo Montanari
angelo.montanari@uniud.it

May 23rd, 2024

Outline

1. Background
   1. Regular and ω-regular languages
   2. The First- and Second-order Theory of One Successor
   3. Automata over finite and infinite words
   4. Linear Temporal Logic
2. The safety fragment of LTL and its theoretical features
   1. Definition of Safety and Cosafety
   2. Characterizations and Normal Forms
   3. Kupferman and Vardi's Classification

Outline

SUCCINCTNESS AND PASTIFICATION

Known results and open questions

Informal definition.

Given two linear-time temporal logics $\mathbb{L}$ and $\mathbb{L}'$, we say that $\mathbb{L}$ can be exponentially more succinct than $\mathbb{L}'$ iff there exists a property such that

- *it can be succinctly expressed in $\mathbb{L}$,*
- *but all formulas of $\mathbb{L}'$ for it are at least exponentially larger.*

Formal definition.

Definition

Given two linear-time temporal logics $\mathbb{L}$ and $\mathbb{L}'$, we say that $\mathbb{L}$ *can be exponentially more succinct than $\mathbb{L}'$ over infinite trace* (resp., *over finite traces*) iff there exists an alphabet Σ and a family of languages $\{\mathcal{L}_n\}_{n>0} \subseteq (2^{\Sigma})^{\omega}$ (resp., $\{\mathcal{L}_n\}_{n>0} \subseteq (2^{\Sigma})^{*}$) such that, for any $n > 0$,

- there exists a formula $\phi \in \mathbb{L}$ over Σ such that its language over infinite traces (resp., over finite traces) is $\mathcal{L}_n$ and $|\phi| \in \mathcal{O}(n)$, and
- for all formulas $\phi' \in \mathbb{L}'$ over Σ, if the language of ϕ' over infinite traces (resp., finite traces) is $\mathcal{L}_n$, then $|\phi'| \in 2^{\Omega(n)}$.

Succinctness

Succinctness is important for various reasons.

In particular,

1. it helps choosing the right formalism when solving problems like reactive synthesis, model checking, and so on;
2. it is an important theoretical tool, that connects the study of computational complexity to that of expressive power.

A well-known result about LTL+P and LTL.

Theorem

LTL+P *can be exponentially more succinct than* LTL.

Reference:

Nicolas Markey (2003). "Temporal logic with past is exponentially more succinct". In: *Bull. EATCS* 79, pp. 122–128

Theorem

$\mathsf{F}(\mathsf{pLTL})$ *can be exponentially more succinct than* $\mathsf{coSafetyLTL}$.

It follows from the result by Markey.

Here we give a simplified version.

Proof.

Steps (proof by contradiction):

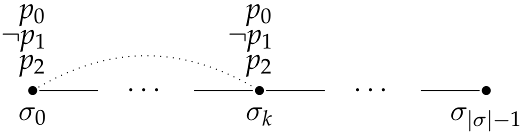
1. For all $n > 0$, find a language A_n such that $\mathcal{L}(\phi_n) = A_n$ and $|\phi_n| \in \mathcal{O}(n)$, for some $\phi_n \in$ F(pLTL).
2. Suppose by contradiction that, for all $n > 0$, there exists a formula ϕ'_n of coSafetyLTL such that $\mathcal{L}(\phi'_n) = \mathcal{L}(\phi_n)$ and $|\phi'_n|$ is polynomial in n.
3. Use ϕ'_n to build a formula ψ_n of LTL+P such that $|\psi_n|$ is polynomial in n. Let $B_n = \mathcal{L}(\psi_n)$.
4. Prove that all NBA for B_n are of size $2^{2^{\Omega(n)}}$.
5. Exploit the fact that there exists a singly exponential translation from LTL+P to equivalent NBA to prove that:
   - all LTL+P formulas of B_n are of size $2^{\Omega(n)}$.
6. Conclude that all formulas of coSafetyLTL that express A_n are of size $2^{\Omega(n)}$.

Succinctness of (co)safety fragments of LTL

1. For all $n > 0$, find a language A_n such that $\mathcal{L}(\phi_n) = A_n$ and $|\phi_n| \in \mathcal{O}(n)$, for some $\phi_n \in \mathsf{F}(\mathsf{pLTL})$.

Let $\Sigma = \{p_0, p_1, \ldots, p_n\}$.

$$A_n \coloneqq \{\sigma \in (2^{\Sigma})^+ \mid \exists k > 0 \,.\, (\bigwedge_{i=0}^{n} (p_i \in \sigma_k \leftrightarrow p_i \in \sigma_0))\}$$



1. For all $n > 0$, find a language A_n such that $\mathcal{L}(\phi_n) = A_n$ and $|\phi_n| \in \mathcal{O}(n)$, for some $\phi_n \in \mathsf{F}(\mathsf{pLTL})$.

Let $\Sigma = \{p_0, p_1, \ldots, p_n\}$.

$$A_n \coloneqq \{\sigma \in (2^\Sigma)^+ \mid \exists k > 0 \,.\, (\bigwedge_{i=0}^{n}(p_i \in \sigma_k \leftrightarrow p_i \in \sigma_0))\}$$

Lemma

For any $n > 0$, there exists a formula $\phi \in \mathsf{F}(\mathsf{pLTL})$ such that $\mathcal{L}(\phi) = A_n$ and $|\phi| \in \mathcal{O}(n)$.

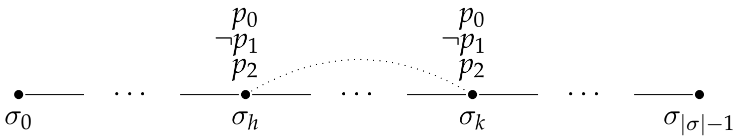
Proof.

$$\mathsf{F}\Big(\bigwedge_{i=0}^{n}(p_i \leftrightarrow \mathsf{Y}\mathsf{O}(\widetilde{\mathsf{Y}}\bot \wedge p_i))\Big)$$

Succinctness of (co)safety fragments of LTL

2. Suppose by contradiction that, for all $n > 0$, there exists a formula ϕ'_n of coSafetyLTL such that $\mathcal{L}(\phi'_n) = \mathcal{L}(\phi_n)$ and $|\phi'_n|$ is polynomial in n.
3. Use ϕ'_n to build a formula ψ_n of LTL+P such that $|\psi_n|$ is polynomial in n. Let $B_n = \mathcal{L}(\psi_n)$.

- $\psi_n := \mathsf{F}(\phi'_n)$
- $B_n := \{\sigma \in (2^{\Sigma})^+ \mid \exists h \geq 0 \,.\, \exists k > h \,.\, (\bigwedge_{i=0}^{n}(p_i \in \sigma_k \leftrightarrow p_i \in \sigma_h))\}$



Succinctness of (co)safety fragments of LTL

2. Suppose by contradiction that, for all $n > 0$, there exists a formula ϕ'_n of coSafetyLTL such that $\mathcal{L}(\phi'_n) = \mathcal{L}(\phi_n)$ and $|\phi'_n|$ is polynomial in n.
3. Use ϕ'_n to build a formula ψ_n of LTL+P such that $|\psi_n|$ is polynomial in n. Let $B_n = \mathcal{L}(\psi_n)$.

Lemma

If there exists a formula of coSafetyLTL for A_n of size less than exponential in n, then there exists a formula of LTL+P for B_n of size less than exponential in n.

4. Prove that all NBA for B_n are of size $2^{2^{\Omega(n)}}$.

Lemma

For any $n > 0$ and any NBA $\mathcal{A}$ *over the alphabet* 2^{Σ}, *if* $\mathcal{L}(\mathcal{A}) = B_n$ *then* $|\mathcal{A}| \in 2^{2^{\Omega(n)}}$.

Reference:

Kousha Etessami, Moshe Y. Vardi, and Thomas Wilke (2002). "First-Order Logic with Two Variables and Unary Temporal Logic". In: *Inf. Comput.* 179.2, pp. 279–295. DOI: 10.1006/inco.2001.2953. URL: https://doi.org/10.1006/inco.2001.2953

4. Exploit the fact that there exists a singly exponential translation from LTL+P to equivalent NBA to prove that:
   - all LTL+P formulas of B_n are of size $2^{\Omega(n)}$.

Proposition

For any LTL *formula* ϕ*, with* $|\phi| = n$*, over the set of atomic propositions* $\mathcal{AP}$*, there exists an* NBA $\mathcal{A}_\phi$ *over the alphabet* $2^{\mathcal{AP}}$ *such that:*

- $\mathcal{L}(\phi) = \mathcal{L}(\mathcal{A}_\phi)$
- $|\,\mathcal{A}_\phi\,| \in 2^{\mathcal{O}(n)}$

Lemma

For any formula $\phi \in$ LTL+P*, if* $\mathcal{L}(\phi) = B_n$*, then* $|\phi| \in 2^{\Omega(n)}$.

4. Conclude that all formulas of coSafetyLTL that express A_n are of size $2^{\Omega(n)}$.

Theorem

For any $n > 0$ *and any formula* $\phi \in \mathsf{coSafetyLTL}$*, if* $\mathcal{L}(\phi) = A_n$*, then* $|\phi| \in 2^{\Omega(n)}$.

Corollary

F(pLTL) *can be exponentially more succinct than* coSafetyLTL.

Succinctness of (co)safety fragments of LTL

By a simple duality argument:

Corollary

G(pLTL) *can be exponentially more succinct than* SafetyLTL.

All these results have been collected in:

Reference:

Alessandro Artale et al. (2023b). "LTL over finite words can be exponentially more succinct than pure-past LTL, and vice versa". In: *Proceedings of the 30th International Symposium on Temporal Representation and Reasoning, TIME 2023, September 25-26, 2023, NCSR Demokritos, Athens, Greece*. Ed. by Florian Bruse Alexander Artikis and Luke Hunsberger. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik

Open problem:

Can coSafetyLTL be exponentially more succinct than F(pLTL)?

Conjecture:

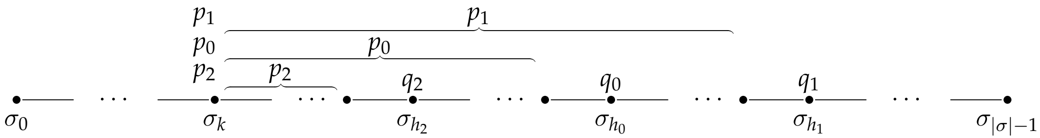
coSafetyLTL can be $n!$ more succinct than F(pLTL).

Succinctness of (co)safety fragments of LTL

Conjecture:

coSafetyLTL can be $n!$ more succinct than F(pLTL).

- $C_n := \{\sigma \in (2^\Sigma)^\omega \mid \exists k \geq 0 \, . \, \bigwedge_{i=1}^{n} (\exists h > k \, . \, (q_i \in \sigma_h \wedge \forall k \leq l < h \, . \, p_i \in \sigma_l))\}$
- $\mathsf{F}(\bigwedge_{i=1}^{n} p_i \, \mathsf{U} \, q_i)$



- In F(pLTL), one needs to specify all permutations of the set $\{q_1, \ldots, q_n\}$.

Theorem

$\mathsf{LTL}[\mathsf{X}, \mathsf{F}]$ *can be exponentially more succinct than* $\mathsf{F}(\mathsf{pLTL}[\mathsf{Y}, \widetilde{\mathsf{Y}}, \mathsf{O}])$, *and vice versa.*

Reference:

Luca Geatti, Alessio Mansutti, and Angelo Montanari (2024). "Succinctness of Cosafety Fragments of LTL via Combinatorial Proof Systems". In: *Foundations of Software Science and Computation Structures - 27th International Conference, FoSSaCS 2024, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2024, Luxembourg City, Luxembourg, April 6-11, 2024, Proceedings, Part II*. ed. by Naoki Kobayashi and James Worrell. Vol. 14575. Lecture Notes in Computer Science. Springer, pp. 95–115. DOI: 10.1007/978-3-031-57231-9_5. URL: https://doi.org/10.1007/978-3-031-57231-9%5C_5

Recall that $\llbracket \mathsf{LTL} \rrbracket \cap \mathsf{SAFETY} = \llbracket \mathsf{LTL} \rrbracket^{<\omega} \cdot (2^{\Sigma})^{\omega}$

Consider now $\mathsf{LTL}_{\mathsf{f}}$, that is, $\llbracket \mathsf{LTL} \rrbracket^{<\omega}$. The following incomparability result holds.

Theorem

- $\mathsf{LTL}_{\mathsf{f}}$ *can be exponentially more succinct than* pLTL.
- pLTL *can be exponentially more succinct than* $\mathsf{LTL}_{\mathsf{f}}$.

Reference:

Alessandro Artale et al. (2023b). "LTL over finite words can be exponentially more succinct than pure-past LTL, and vice versa". In: *Proceedings of the 30th International Symposium on Temporal Representation and Reasoning, TIME 2023, September 25-26, 2023, NCSR Demokritos, Athens, Greece*. Ed. by Florian Bruse Alexander Artikis and Luke Hunsberger. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik

Succinctness and Pastification

- Let us consider again the case of coSafetyLTL and F(pLTL).
- Succinctness properties can be considered as lower bounds for the transformation of coSafetyLTL into F(pLTL).
- The transformation of a pure future fragment into a pure past one is called

PASTIFICATION

- Originally introduced in the context of synthesis of timed temporal logics:

Reference:

Oded Maler, Dejan Nickovic, and Amir Pnueli (2007). "On synthesizing controllers from bounded-response properties". In: *Proceedings of the International Conference on Computer Aided Verification*. Springer, pp. 95–107. DOI: 10.1023/A:1008734703554

- We now look at some pastification algorithms (upper bounds)

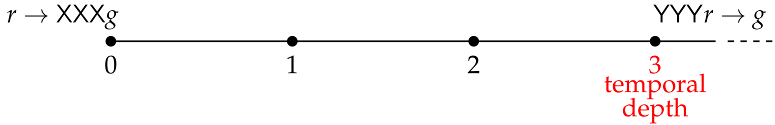
Pastification Algorithms

Let us briefly consider pastification algorithms for the following fragments:

- $\mathsf{LTL}[\mathsf{X}]$
  - *polynomial-size pastification*
- $\mathsf{LTL}[\mathsf{X}, \mathsf{F}]$
  - *exponential-size pastification*
- $\mathsf{coSafetyLTL}$
  - *triply exponential-size pastification*
- LTL_f
  - *triply exponential-size pastification*

Transforming LTL[X] into F(pLTL)

- Let $\phi \in \mathsf{LTL}[\mathsf{X}]$.
- There exists a time point $d \in \mathbb{N}$, that is, the temporal depth of ϕ, such that the subsequent states cannot be constrained by ϕ.
  - temporal depth of ϕ = maximum number of nested X operators
- Thus, we can write a formula (the pastification of ϕ) that uses only past operators and is equivalent to ϕ when interpreted at d.
- Example: $\phi := r \rightarrow \mathsf{XXX}g$



It holds that: $r \rightarrow \mathsf{XXX}g \equiv \mathsf{F}(\mathsf{at}_3 \wedge (\mathsf{YYY}r \rightarrow g))$.

- where $\mathsf{at}_3 := \widetilde{\mathsf{Y}}\widetilde{\mathsf{Y}}\widetilde{\mathsf{Y}}\bot \wedge \mathsf{YY}\top$.

Transforming LTL[X] into F(pLTL)

Theorem

There is a polynomial-size pastification of LTL[X] *into* F(pLTL).

Reference:

Oded Maler, Dejan Nickovic, and Amir Pnueli (2007). "On synthesizing controllers from bounded-response properties". In: *Proceedings of the International Conference on Computer Aided Verification*. Springer, pp. 95–107. DOI: 10.1023/A:1008734703554

Theorem

There is a 1 exponential-size pastification of LTL[X, F] *into* F(pLTL).

- Data structure: *dependency trees*

Reference:

Alessandro Artale et al. (2023a). "A Singly Exponential Transformation of LTL[X,F] into Pure Past LTL". In: *Proceedings of the 20th International Conference on Principles of Knowledge Representation and Reasoning, KR 2023, Rhodes, Greece. September 2-8, 2023*

Classical Pastification Approach:

Formula ϕ of $\mathsf{LTL_f}$

of size n

$\downarrow$

NFA

of size $2^{\mathcal{O}(n)}$

Subset construction $\downarrow$

DFA

of size $2^{2^{\mathcal{O}(n)}}$

Krohn-Rhodes decomposition theorem $\downarrow$

Formula ψ of pLTL

of size $2^{2^{2^{\mathcal{O}(n)}}}$

Transforming LTL[X, F] into F(pLTL)

Classical Pastification Approach:

Formula ϕ of $\mathsf{LTL_f}$

of size n

$\downarrow$

NFA

of size $2^{\mathcal{O}(n)}$

$\downarrow$ Subset construction

DFA

of size $2^{2^{\mathcal{O}(n)}}$

$\downarrow$ Krohn-Rhodes decomposition theorem

Formula ψ of pLTL

of size $2^{2^{2^{\mathcal{O}(n)}}}$

Our Approach:

Formula ϕ of LTL[X, F]

of size n

$\downarrow$

Normal Form

of size $c^{\mathcal{O}(n^2)}$

$\downarrow$

Dependency Tree

of size $c^{\mathcal{O}(n^2)}$

$\downarrow$

Formula ψ of F(pLTL)

of size $(c^{\mathcal{O}(n^2)})^2$

Transforming LTL[X, F] into F(pLTL)

Classical Pastification Approach:

Formula ϕ of $\mathsf{LTL_f}$

of size n

$\downarrow$

NFA

of size $2^{\mathcal{O}(n)}$

Subset construction $\downarrow$

DFA

of size $2^{2^{\mathcal{O}(n)}}$

Krohn-Rhodes decomposition theorem $\downarrow$

Formula ψ of pLTL

of size $2^{2^{2^{\mathcal{O}(n)}}}$

Our Approach:

- Purely syntactical.
- Implementation in Pastello (<500 lines of code).

Consider the following formula of $\mathsf{LTL}[\mathsf{X},\mathsf{F}]$:

$$\mathsf{F}\big((p_1 \vee \mathsf{X}\mathsf{F}q_1) \wedge (p_2 \vee \mathsf{X}\mathsf{F}q_2)\big)$$

In general, formulas of $\mathsf{LTL}[\mathsf{X},\mathsf{F}]$ contains two degrees of uncertainty:

- both on which eventualities have to happen
  - *"which of the q_i are going to be fulfilled?"*
- and on when an eventuality has to be realized
  - *"in which order the q_i are going to be fulfilled?"*

We designed the normal form to *remove* the 1st type of uncertainty.

Transformation into Normal Form

Definition

Let ψ be a pLTL formula. The logic $\mathsf{LTL}[\mathsf{F}, \wedge]$ is the set of formulas ϕ generated by the following grammar:

$$\phi := \psi \mid \phi \wedge \phi \mid \mathsf{F}\phi$$

Uncertainty *only* about *when* an eventuality is going to be fulfilled.

Transformation into Normal Form

Definition

Let ψ be a pLTL formula. The logic $\mathsf{LTL}[\mathsf{F}, \wedge]$ is the set of formulas ϕ generated by the following grammar:

$$\phi := \psi \mid \phi \wedge \phi \mid \mathsf{F}\phi$$

Uncertainty *only* about *when* an eventuality is going to be fulfilled.

Definition

The *normal form of* $\mathsf{LTL}[\mathsf{X}, \mathsf{F}]$, denoted with $\mathsf{nfLTL}[\mathsf{X}, \mathsf{F}]$, is the set of formulas of type

$$\mathsf{X}^k \bigvee_{i=1}^{c} \phi_i$$

for some $k, c \in \mathbb{N}$, such that $\phi_i \in \mathsf{LTL}[\mathsf{F}, \wedge]$ for any $1 \leq i \leq c$.

The uncertainty on *which* eventuality is going to be fulfilled is *only* at top-level.

R_1: $\mathsf{X}^i\phi_1 \otimes \mathsf{X}^j\phi_2 \rightsquigarrow \begin{cases} \mathsf{X}^i(\phi_1 \otimes \mathsf{Y}^{i-j}\phi_2) & \text{if } i > j \\ \mathsf{X}^j(\mathsf{Y}^{j-i}\phi_1 \otimes \phi_2) & \text{otherwise} \end{cases}$

R_2: $\mathsf{F}\mathsf{X}^i\phi_1 \rightsquigarrow \mathsf{X}^i\mathsf{F}\phi_1$

R_3: $\mathsf{Y}^i(\phi_1 \otimes \phi_2) \rightsquigarrow \mathsf{Y}^i\phi_1 \otimes \mathsf{Y}^i\phi_2$

R_4: $\mathsf{Y}^i\mathsf{F}\phi_1 \rightsquigarrow \mathsf{F}\mathsf{Y}^i\phi_1$

R_5: $\mathsf{F}(\bigvee_{i=1}^{c}\phi_i) \rightsquigarrow \bigvee_{i=1}^{c}\mathsf{F}\phi_i$

R_6: $\bigwedge_{i=1}^{c}\bigvee_{j=1}^{d_i}\phi_{i,j} \rightsquigarrow \bigvee_{S\in A}\bigwedge_{\psi\in S}\psi$

Example

$\mathsf{F}\big((p_1 \vee \mathsf{X}\mathsf{F}q_1) \wedge (p_2 \vee \mathsf{X}\mathsf{F}q_2)\big)$

$\equiv$ rule R_1 two times

$\mathsf{F}\big(\mathsf{X}((\mathsf{Y}p_1 \vee \mathsf{F}q_1) \wedge (\mathsf{Y}p_2 \vee \mathsf{F}q_2))\big)$

$\equiv$ rule R_2

$\mathsf{X}\mathsf{F}\big((\mathsf{Y}p_1 \vee \mathsf{F}q_1) \wedge (\mathsf{Y}p_2 \vee \mathsf{F}q_2)\big)$

$\equiv$ rule R_6

$\mathsf{X}\mathsf{F}\big((\mathsf{Y}p_1 \wedge \mathsf{Y}p_2) \vee (\mathsf{Y}p_1 \wedge \mathsf{F}q_2) \vee (\mathsf{F}q_1 \wedge \mathsf{Y}p_2) \vee (\mathsf{F}q_1 \wedge \mathsf{F}q_2)\big)$

$\equiv$ rule R_5

$\mathsf{X}\big(\mathsf{F}(\mathsf{Y}p_1 \wedge \mathsf{Y}p_2) \vee \mathsf{F}(\mathsf{Y}p_1 \wedge \mathsf{F}q_2) \vee$
$\mathsf{F}(\mathsf{F}q_1 \wedge \mathsf{Y}p_2) \vee \mathsf{F}(\mathsf{F}q_1 \wedge \mathsf{F}q_2)\big)$

DEPENDENCY TREES

From Normal Form to Dependency Trees

- From the transformation into normal form, we have a formula of this type:

$$\mathsf{X}^{k} \bigvee_{i=1}^{c} \phi_i$$

  with $\phi_i \in \mathsf{LTL}[\mathsf{F}, \wedge]$ and $k \in \mathbb{N}$.
- We consider separately each $\mathsf{LTL}[\mathsf{F}, \wedge]$ formula ϕ_i (and the k) and we build its dependency tree.

From Normal Form to Dependency Trees

Each formula ϕ of $\mathsf{LTL}[\mathsf{F}, \wedge]$ is of the form

$$\alpha \wedge \mathsf{F}(\beta_1) \wedge \cdots \wedge \mathsf{F}(\beta_n)$$

for some $n \in \mathbb{N}$, where $\alpha \in \mathsf{pLTL}$ and $\beta_i \in \mathsf{LTL}[\mathsf{F}, \wedge]$, for each $1 \leq i \leq n$.

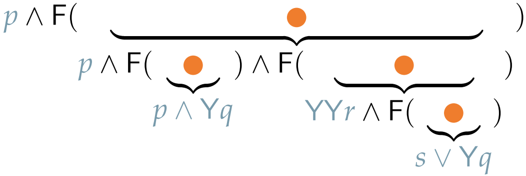
Example:

$p \wedge \mathsf{F}(p \wedge \mathsf{F}(p \wedge \mathsf{Y}q) \wedge \mathsf{F}(\mathsf{YY}r \wedge \mathsf{F}(s \vee \mathsf{Y}q)))$

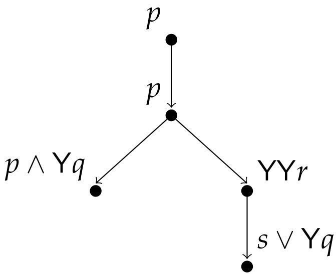
$$p \wedge \mathsf{F}(\underbrace{\bullet}_{p \wedge \mathsf{F}(\underbrace{\bullet}_{p \wedge \mathsf{Y}q}) \wedge \mathsf{F}(\underbrace{\bullet}_{\mathsf{YY}r \wedge \mathsf{F}(\underbrace{\bullet}_{s \vee \mathsf{Y}q})})})$$

Dependency Trees

- A *Dependency Tree* is a tree-shaped structure that reflects the nesting of the F operators in ϕ.
    - nodes = pLTL subformulas
    - edges = F operators
- Whenever a *conjunction* of multiple eventualities has to be realised in the future of a given node, the tree branches *without imposing any ordering among them*.



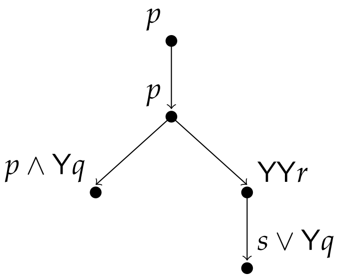
$p \wedge \mathsf{F}($ $)$
$p \wedge \mathsf{F}($ $) \wedge \mathsf{F}($ $)$
$p \wedge \mathsf{Y}q$
$\mathsf{YY}r \wedge \mathsf{F}($ $)$
$s \vee \mathsf{Y}q$



p
p
$p \wedge \mathsf{Y}q$
$\mathsf{YY}r$
$s \vee \mathsf{Y}q$

FROM DEPENDENCY TREES TO PURE PAST LTL

How can we *"pastify"* this dependency tree?

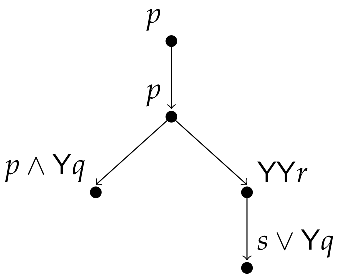


From Dependency Trees to Pure Past LTL

How can we *"pastify"* this dependency tree?

Wrong solution:

- Specify all the orders between the nodes on different branches.
- E.g.
  1. $p \wedge \mathsf{Y}q \quad < \quad \mathsf{YY}r \quad < \quad s \vee \mathsf{Y}q$
  2. $\mathsf{YY}r \quad < \quad p \wedge \mathsf{Y}q \quad < \quad s \vee \mathsf{Y}q$
  3. $\mathsf{YY}r \quad < \quad s \vee \mathsf{Y}q \quad < \quad p \wedge \mathsf{Y}q$
- Complexity: $n!$ (n factorial)

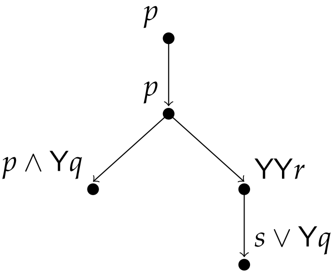


From Dependency Trees to Pure Past LTL

How can we *"pastify"* this dependency tree?

Efficient solution:

- Look the tree *bottom up*.

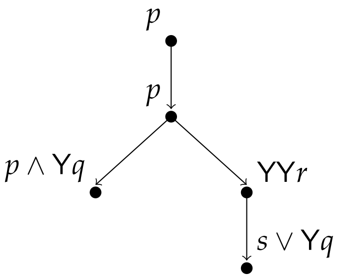


From Dependency Trees to Pure Past LTL

How can we *"pastify"* this dependency tree?

Efficient solution:

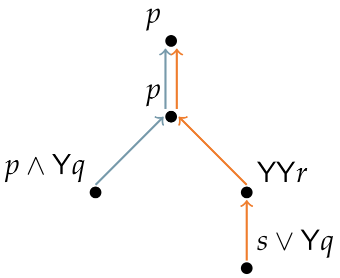
- Look the tree *bottom up*.
- Consider separately each path of the tree that goes from the root to a leaf.



How can we *"pastify"* this dependency tree?

Efficient solution:

- Look the tree *bottom up*.
- Consider separately each path of the tree that goes from the root to a leaf.
- *"Rewrite"* each branch upside-down (*i.e.*, going from the leaf to the root), by means of a pLTL formula.



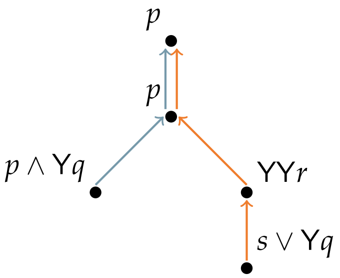
$\mathsf{O}(p \wedge \mathsf{Y}q \wedge \mathsf{O}(p \wedge \mathsf{O}(p)))$

From Dependency Trees to Pure Past LTL

How can we *"pastify"* this dependency tree?

Efficient solution:

- Look the tree *bottom up*.
- Consider separately each path of the tree that goes from the root to a leaf.
- *"Rewrite"* each branch upside-down (*i.e.*, going from the leaf to the root), by means of a pLTL formula.
- Consider the *conjunction* between the pure past formulas corresponding to each branch.



$$\mathsf{O}((p \wedge \mathsf{Y}q) \wedge \mathsf{O}(p \wedge \mathsf{O}(p)))$$

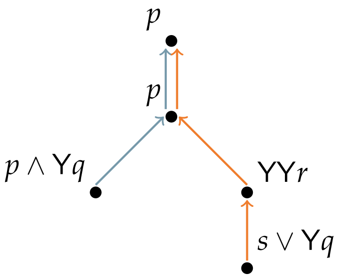
$$\wedge$$

$$\mathsf{O}((s \vee \mathsf{Y}q) \wedge \mathsf{O}(\mathsf{YY}r \wedge \mathsf{O}(p \wedge \mathsf{O}(p))))$$

From Dependency Trees to Pure Past LTL

Why does it work?

- Such formulas (one for each reversed path) will coincide in the description of the "common past".



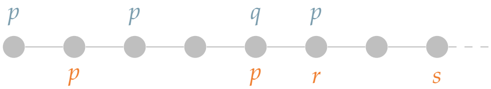
$$\mathsf{O}((p \wedge \mathsf{Y}q) \wedge \mathsf{O}(p \wedge \mathsf{O}(p)))$$

$$\wedge$$

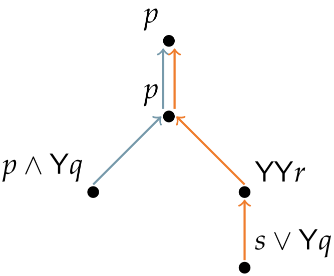
$$\mathsf{O}((s \vee \mathsf{Y}q) \wedge \mathsf{O}(\mathsf{YY}r \wedge \mathsf{O}(p \wedge \mathsf{O}(p))))$$

Why does it work?

- Such formulas (one for each reversed path) will coincide in the description of the "common past".
- This can create this situation:



They are *out of phase*.



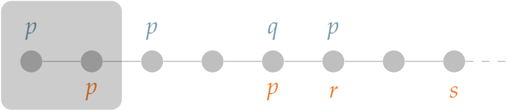
$$\mathsf{O}((p \wedge \mathsf{Y}q) \wedge \boxed{\mathsf{O}(p \wedge \mathsf{O}(p))})$$

$$\wedge$$

$$\mathsf{O}((s \vee \mathsf{Y}q) \wedge \mathsf{O}(\mathsf{YY}r \wedge \boxed{\mathsf{O}(p \wedge \mathsf{O}(p))}))$$

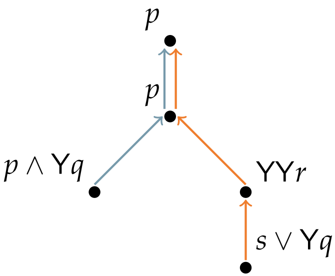
Why does it work?

- Such formulas (one for each reversed path) will coincide in the description of the "common past".
- This can create this situation:



They are *out of phase*.

- The *first* fulfillment of the "common past" is good for both branches.



$$\mathsf{O}((p \wedge \mathsf{Y}q) \wedge \boxed{\mathsf{O}(p \wedge \mathsf{O}(p))}\,)$$

$$\wedge$$

$$\mathsf{O}((s \vee \mathsf{Y}q) \wedge \mathsf{O}(\mathsf{Y}\mathsf{Y}r \wedge \boxed{\mathsf{O}(p \wedge \mathsf{O}(p))}\,))$$

Transforming coSafetyLTL into F(pLTL)

Theorem

There is a 3 exponential-size pastification of coSafetyLTL *into* F(pLTL).

Let ϕ be a coSafetyLTL formula.

1. Build the DFA $\mathcal{A}'_\phi$ for the set of *good prefixes* of ϕ:
   - doubly exponential blow-up
2. Use the Krohn-Rhodes Primary Decomposition Theorem to build a cascade product equivalent to $\mathcal{A}'_\phi$.
   - exponential blow-up
3. Translate the cascade product into a formula ψ of pLTL. Return $\mathsf{F}(\psi)$.
   - linear

Total: triply exponential pastification algorithm.

Transforming coSafetyLTL into F(pLTL)

Theorem

There is a 3 exponential-size pastification of coSafetyLTL *into* F(pLTL).

Reference:

Oded Maler and Amir Pnueli (1990). "Tight bounds on the complexity of cascaded decomposition of automata". In: *Proceedings of the 31st Annual Symposium on Foundations of Computer Science*. IEEE, pp. 672–682

There are two missing exponentials between the best-known upper and lower bounds:

- best known upper bound: triply exponential
- best known lower bound: singly exponential

Transforming $\mathsf{LTL_f}$ into pLTL

As for $\mathsf{LTL_f}$, the best known algorithm is the same as the one for coSafetyLTL.

Let ϕ be a $\mathsf{LTL_f}$ formula.

1. Build a NFA $\mathcal{A}_\phi$ for ϕ.
   - exponential blow-up
2. Determinize $\mathcal{A}_\phi$ into a DFA $\mathcal{A}'_\phi$.
   - exponential blow-up
3. Use the Krohn-Rhodes Primary Decomposition Theorem to build a cascade product equivalent to $\mathcal{A}'_\phi$.
   - exponential blow-up
4. Translate the cascade product into pLTL.
   - linear

Total: triply exponential pastification algorithm.

	Upper bound	Lower bound
LTL[X]	linear	linear
LTL[X, F]	1-exp	1-exp
coSafetyLTL	3-exp	1-exp
LTL_f	3-exp	1-exp

A *polynomial-size* pastification algorithm is a very uncommon feature for a logic.

CONCLUSIONS

Conclusions: results

- Characterizations of safety and cosafety fragments of LTL:
  - reduction from infinite to finite words reasoning
- Role of past temporal operators in the definition of canonical forms
- Kupferman & Vardi's classification of safety properties:
  - intentionally, accidentally, and pathologically safe.
- Algorithms to recognize safety automata and LTL safety formulas
- Algorithms to build the set of bad prefixes
  - *doubly exponential* DFA
- Algorithms for
  - satisfiability checking
  - model checking
    - the worst-case complexity does not change
    - efficient algorithms in practice
  - reactive synthesis
    - avoid Safra's determinization
    - by using past operators, the worst-case complexity can be decreased by one exponential
- Succinctness issues
  - $\mathsf{G}(\mathsf{pLTL})$ can be exponentially more succinct than $\mathsf{SafetyLTL}$
- Pastification algorithms

Conclusions: open problems

- Some interesting open problems:
  - Worst-case complexity of safety model checking
  - Succinctness lower bounds
    - coSafetyLTL
    - $\mathsf{LTL_f}$
  - Efficient pastification algorithms

REFERENCES

Alessandro Artale et al. (2023a). "A Singly Exponential Transformation of LTL[X,F] into Pure Past LTL". In: *Proceedings of the 20th International Conference on Principles of Knowledge Representation and Reasoning, KR 2023, Rhodes, Greece. September 2-8, 2023.*

— (2023b). "LTL over finite words can be exponentially more succinct than pure-past LTL, and vice versa". In: *Proceedings of the 30th International Symposium on Temporal Representation and Reasoning, TIME 2023, September 25-26, 2023, NCSR Demokritos, Athens, Greece.* Ed. by Florian Bruse Alexander Artikis and Luke Hunsberger. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

Kousha Etessami, Moshe Y. Vardi, and Thomas Wilke (2002). "First-Order Logic with Two Variables and Unary Temporal Logic". In: *Inf. Comput.* 179.2, pp. 279–295. DOI: 10.1006/inco.2001.2953. URL: https://doi.org/10.1006/inco.2001.2953.

Luca Geatti, Alessio Mansutti, and Angelo Montanari (2024). "Succinctness of Cosafety Fragments of LTL via Combinatorial Proof Systems". In: *Foundations of Software Science and Computation Structures - 27th International Conference, FoSSaCS 2024, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2024, Luxembourg City, Luxembourg, April 6-11, 2024, Proceedings, Part II*. Ed. by Naoki Kobayashi and James Worrell. Vol. 14575. Lecture Notes in Computer Science. Springer, pp. 95–115. DOI: 10.1007/978-3-031-57231-9_5. URL: https://doi.org/10.1007/978-3-031-57231-9%5C_5.

Oded Maler, Dejan Nickovic, and Amir Pnueli (2007). "On synthesizing controllers from bounded-response properties". In: *Proceedings of the International Conference on Computer Aided Verification*. Springer, pp. 95–107. DOI: 10.1023/A:1008734703554.

Oded Maler and Amir Pnueli (1990). "Tight bounds on the complexity of cascaded decomposition of automata". In: *Proceedings of the 31st Annual Symposium on Foundations of Computer Science*. IEEE, pp. 672–682.

Nicolas Markey (2003). "Temporal logic with past is exponentially more succinct". In: *Bull. EATCS* 79, pp. 122–128.