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

The Safety Fragment of Temporal Logics on Infinite Sequences

Lesson 6

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

April 15th, 2024

We say that a temporal logic $\mathbb{L}$ is *cosafety* iff, for any $\phi \in \mathbb{L}$, $\mathcal{L}(\phi)$ is *cosafety*.

coSafetyLTL

Definition

$\phi := p \mid \neg p \mid \phi \vee \phi \mid \phi \wedge \phi \mid \mathsf{X}\phi \mid \mathsf{F}\phi \mid \phi \,\mathsf{U}\, \phi$

Example:

$p \,\mathsf{U}\, q$

F(pLTL)

Definition

$\phi := \mathsf{F}(\alpha)$, where $\alpha \in$ pLTL, that is α is a pure-past LTL formula.

Example:

$\mathsf{F}(q \wedge \widetilde{\mathsf{Y}}\mathsf{H}p)$

F(pLTL) is the canonical form of coSafetyLTL.

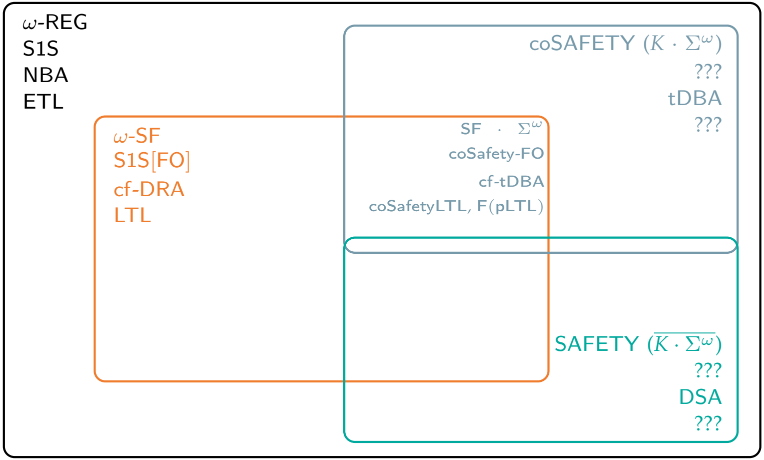
The cosafety fragment of LTL

Theorem

- coSafetyLTL *and* F(pLTL) *are expressively equivalent.*
- coSafetyLTL *and* F(pLTL) *are expressively complete w.r.t.* $[\![\mathsf{LTL}]\!] \cap \mathsf{coSAFETY}$.

Reference:

Edward Y. Chang, Zohar Manna, and Amir Pnueli (1992). "Characterization of Temporal Property Classes". In: *Proceedings of the 19th International Colloquium on Automata, Languages and Programming*. Ed. by Werner Kuich. Vol. 623. Lecture Notes in Computer Science. Springer, pp. 474–486. DOI: 10.1007/3-540-55719-9_97



ω-REG
S1S
NBA
ETL
coSAFETY $(K \cdot \Sigma^\omega)$
???
tDBA
???
ω-SF
S1S[FO]
cf-DRA
LTL
SF $\cdot$ Σ^ω
coSafety-FO
cf-tDBA
coSafetyLTL, F(pLTL)
SAFETY $(\overline{K \cdot \Sigma^\omega})$
???
DSA
???

Proposition

$[\![\mathsf{coSafetyLTL}]\!]^{<\omega} \subsetneq [\![\mathsf{LTL}]\!]^{<\omega}$

Proof.

- It is simple to prove that, for all $\phi \in \mathsf{coSafetyLTL}$, $\mathcal{L}^{<\omega}(\phi) = \mathcal{L}^{<\omega}(\phi) \cdot \Sigma^*$. In particular, either $|\mathcal{L}^{<\omega}(\phi)| = 0$ or $|\mathcal{L}^{<\omega}(\phi)| = \omega$ for all $\phi \in \mathsf{coSafetyLTL}$.
- In $\mathsf{LTL_f}$ we can use the *weak tomorrow* operator to hook the last position of a finite word.

$$\psi := p \wedge \widetilde{\mathsf{X}}\bot$$

The formula ψ is such that $|\mathcal{L}^{<\omega}(\psi)| = 1$. Therefore, it can't be expressed in coSafetyLTL over finite words. $\square$

Proposition

$\llbracket \mathsf{coSafetyLTL} \rrbracket^{<\omega} \subsetneq \llbracket \mathsf{LTL} \rrbracket^{<\omega}$

Proposition

$\llbracket \mathsf{coSafetyLTL} \rrbracket^{<\omega} \cdot (2^\Sigma)^\omega = \llbracket \mathsf{LTL} \rrbracket^{<\omega} \cdot (2^\Sigma)^\omega$

Reference:

Alessandro Cimatti et al. (2022). "A first-order logic characterisation of safety and co-safety languages". In: *Foundations of Software Science and Computation Structures - 25th International Conference, FOSSACS 2022, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2022, Munich, Germany, April 2-7, 2022, Proceedings*. Ed. by Patricia Bouyer and Lutz Schröder. Vol. 13242. Lecture Notes in Computer Science. Springer, pp. 244–263. DOI: 10.1007/978-3-030-99253-8_13. URL: https://doi.org/10.1007/978-3-030-99253-8%5C_13

$$
\begin{array}{l}
[\![\mathsf{LTL}]\!] \cap \mathsf{coSAFETY} \\
\qquad = \\
[\![\mathsf{coSafetyLTL}]\!] \\
\qquad = \\
[\![\mathsf{F}(\mathsf{pLTL})]\!] \\
\qquad = \\
[\![\mathsf{coSafetyLTL}]\!]^{<\omega} \cdot (2^{\Sigma})^{\omega} \\
\qquad = \\
[\![\mathsf{LTL}]\!]^{<\omega} \cdot (2^{\Sigma})^{\omega}
\end{array}
$$

The safety fragment of LTL

Definition

The *safety fragment of* LTL is the set of languages in this set:

$$[\![\mathsf{LTL}]\!] \cap \mathsf{SAFETY}$$

We will see four characterizations in terms of:

- regular expressions
- first-order logic
- automata
- temporal logic

Definition

The *safety fragment of* LTL is the set of languages in this set:

$$[\![\mathsf{LTL}]\!] \cap \mathsf{SAFETY}$$

ω-regular expressions

$\overline{\mathsf{SF} \cdot \Sigma^\omega} = \{\overline{K \cdot \Sigma^\omega} \mid K \in \mathsf{SF}\}$

- the "SF " part corresponds to LTL
- the "$\overline{\cdot \Sigma^\omega}$" part corresponds to being a safety fragment

---

Ina Schiering and Wolfgang Thomas (1996). "Counter-free automata, first-order logic, and star-free expressions extended by prefix oracles". In: *Developments in Language Theory, II (Magdeburg, 1995), Worl Sci. Publishing, River Edge, NJ*, pp. 166–175

Definition

The *safety fragment of* LTL is the set of languages in this set:

$$[\![\mathsf{LTL}]\!] \cap \mathsf{SAFETY}$$

First-order logic

We define Safety-FO as the fragment of S1S[FO] in which quantifiers are bounded as follows:

- $\exists y \,.\, (x < y < z \land \dots)$
- $\forall y \,.\, (x < y \rightarrow \dots)$

Alessandro Cimatti et al. (2022). "A first-order logic characterisation of safety and co-safety languages". In: *Foundations of Software Science and Computation Structures - 25th International Conference, FOSSACS 2022, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2022, Munich, Germany, April 2-7, 2022, Proceedings*. Ed. by Patricia Bouyer and Lutz Schröder. Vol. 13242. Lecture Notes in Computer Science. Springer, pp. 244–263. DOI: 10.1007/978-3-030-99253-8_13. URL: https://doi.org/10.1007/978-3-030-99253-8%5C_13

Definition

The *safety fragment of* LTL is the set of languages in this set:

$$[\![\mathsf{LTL}]\!] \cap \mathsf{SAFETY}$$

First-order logic

Example

$\phi(x) := \forall y \, . \, ((x < y \land G(y)) \to \exists z \, . \, (x < z < y \land R(z)))$

The safety fragment of LTL

Definition

The *safety fragment of* LTL is the set of languages in this set:

$$[\![\mathsf{LTL}]\!] \cap \mathsf{SAFETY}$$

First-order logic

- the "first-order" part corresponds to LTL
- the "bounded quantifiers" part corresponds to being a safety fragment

Definition

The *safety fragment of* LTL is the set of languages in this set:

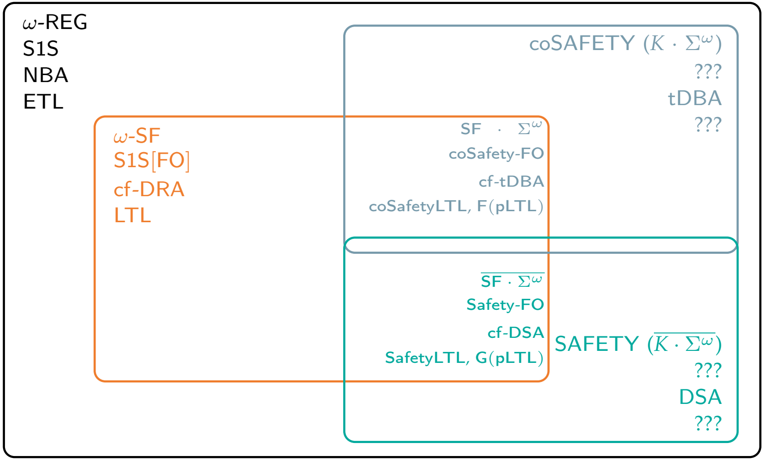
$$[\![\mathsf{LTL}]\!] \cap \mathsf{SAFETY}$$

Automata

cf-DSA = counter-free DSA

- the "counter-free" part corresponds to LTL
- the "DSA " part corresponds to being a safety fragment

Ina Schiering and Wolfgang Thomas (1996). "Counter-free automata, first-order logic, and star-free expressions extended by prefix oracles". In: *Developments in Language Theory, II (Magdeburg, 1995), Worl Sci. Publishing, River Edge, NJ*, pp. 166–175


ω-REG
S1S
NBA
ETL
coSAFETY $(K \cdot \Sigma^\omega)$
???
tDBA
???
ω-SF
S1S[FO]
cf-DRA
LTL
SF $\cdot$ Σ^ω
coSafety-FO
cf-tDBA
coSafetyLTL, F(pLTL)
$\overline{\text{SF} \cdot \Sigma^\omega}$
Safety-FO
cf-DSA
SafetyLTL, G(pLTL)
SAFETY $(\overline{K \cdot \Sigma^\omega})$
???
DSA
???

We say that a temporal logic $\mathbb{L}$ is *safety* iff, for any $\phi \in \mathbb{L}$, $\mathcal{L}(\phi)$ is *safety*.

SafetyLTL

Definition

$\phi := p \mid \neg p \mid \phi \vee \phi \mid \phi \wedge \phi \mid \mathsf{X}\phi \mid \mathsf{G}\phi \mid \phi \,\mathsf{R}\, \phi$

Example:

$\mathsf{G}(r \to \mathsf{XX}g)$

G(pLTL)

Definition

$\phi := \mathsf{G}(\alpha)$, where $\alpha \in$ pLTL, that is α is a pure-past LTL formula.

Example:

$\mathsf{G}(\widetilde{\mathsf{Y}}\widetilde{\mathsf{Y}}r \to g)$

G(pLTL) is the canonical form of SafetyLTL.

Proposition

- $\phi \in \mathsf{SafetyLTL}$ *iff* $\mathtt{nnf}(\neg\phi) \in \mathsf{coSafetyLTL}$
- $\phi \in \mathsf{G}(\mathsf{pLTL})$ *iff* $\mathtt{nnf}(\neg\phi) \in \mathsf{F}(\mathsf{pLTL})$

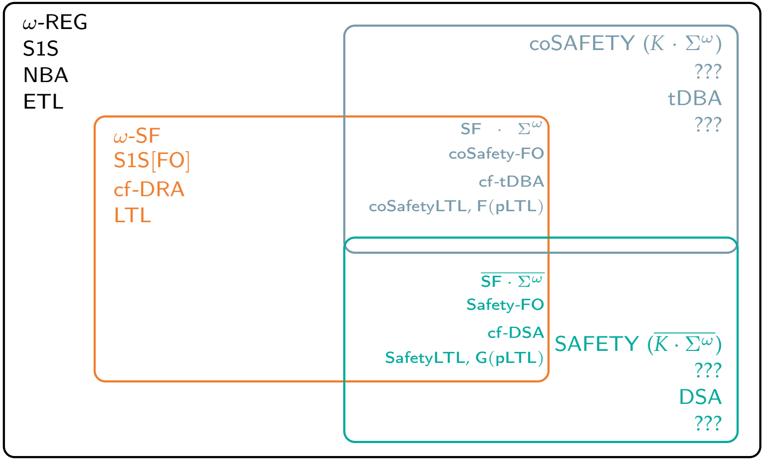
The cosafety fragment of LTL

Theorem

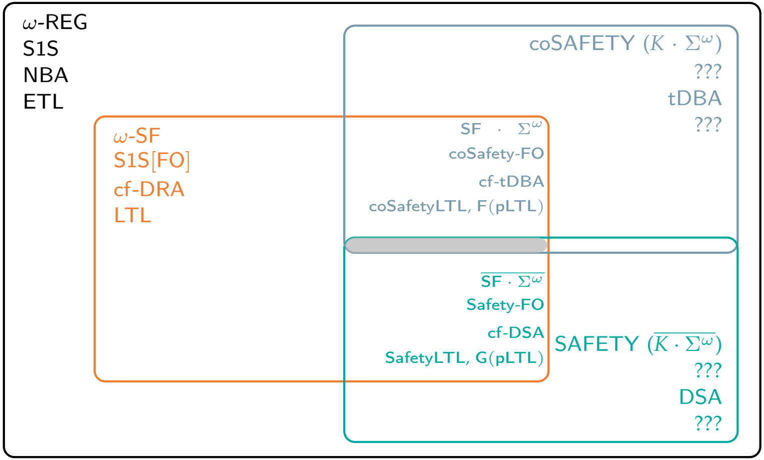
- SafetyLTL *and* $\mathsf{G(pLTL)}$ *are expressively equivalent.*
- SafetyLTL *and* $\mathsf{G(pLTL)}$ *are expressively complete w.r.t.* $[\![\mathsf{LTL}]\!] \cap \mathsf{SAFETY}$.

Reference:

Edward Y. Chang, Zohar Manna, and Amir Pnueli (1992). "Characterization of Temporal Property Classes". In: *Proceedings of the 19th International Colloquium on Automata, Languages and Programming*. Ed. by Werner Kuich. Vol. 623. Lecture Notes in Computer Science. Springer, pp. 474–486. DOI: 10.1007/3-540-55719-9_97


ω-REG
S1S
NBA
ETL
coSAFETY $(K \cdot \Sigma^\omega)$
???
tDBA
???
ω-SF
S1S[FO]
cf-DRA
LTL
SF $\cdot$ Σ^ω
coSafety-FO
cf-tDBA
coSafetyLTL, F(pLTL)
$\overline{\text{SF} \cdot \Sigma^\omega}$
Safety-FO
cf-DSA
SafetyLTL, G(pLTL)
SAFETY $(\overline{K \cdot \Sigma^\omega})$
???
DSA
???

Set-theoretic view of (co)safety ω-languages



- We denote with $\mathbb{B}$ the set of Boolean formulas.
- We denote with $\mathsf{LTL}[\mathsf{X}]$ the set of LTL formulas in which the only temporal operator that is used is the *tomorrow* (X).

Proposition

- $\mathbb{B} \subseteq \mathsf{LTL} \cap \mathsf{coSAFETY} \cap \mathsf{SAFETY}$
- $\mathsf{LTL}[\mathsf{X}] \subseteq \mathsf{LTL} \cap \mathsf{coSAFETY} \cap \mathsf{SAFETY}$

Other safety and cosafety fragments

Cosafety

- We denote with $\mathsf{LTL}[\mathsf{X}, \mathsf{F}]$ the set of coSafetyLTL formulas in which the only temporal operators that are used are the *tomorrow* (X) and the *eventually* (F).
- Clearly, $\mathsf{LTL}[\mathsf{X}, \mathsf{F}]$ is a cosafety logic, but it is strictly less expressive than coSafetyLTL.

Proposition

$[\![\mathsf{LTL}[\mathsf{X}, \mathsf{F}]]\!] \subsetneq [\![\mathsf{coSafetyLTL}]\!]$

E.g. $p \,\mathsf{U}\, q$ is not definable in $\mathsf{LTL}[\mathsf{X}, \mathsf{F}]$.

Safety

- We denote with $\mathsf{LTL}[\mathsf{X}, \mathsf{G}]$ the set of SafetyLTL formulas in which the only temporal operators that are used are the *tomorrow* (X) and the *globally* (G).
- Clearly, $\mathsf{LTL}[\mathsf{X}, \mathsf{G}]$ is a safety logic, but it is strictly less expressive than SafetyLTL.

Proposition

$[\![\mathsf{LTL}[\mathsf{X}, \mathsf{G}]]\!] \subsetneq [\![\mathsf{SafetyLTL}]\!]$

E.g. $p \,\mathsf{R}\, q$ is not definable in $\mathsf{LTL}[\mathsf{X}, \mathsf{G}]$.

The Temporal Hierarchy

Reactivity

$\bigcup_{n=1}^{\infty}\left(\bigwedge_{i=1}^{n}(\mathsf{GF}\alpha_i \vee \mathsf{FG}\beta_i) \right)$

Response

$\mathsf{GF}\alpha$

Persistence

$\mathsf{FG}\alpha$

Obligation

$\bigcup_{n=1}^{\infty}\left(\bigwedge_{i=1}^{n}(\mathsf{G}\alpha_i \vee \mathsf{F}\beta_i) \right)$

Safety

$\mathsf{G}\alpha$

Cosafety

$\mathsf{F}\alpha$

Legend:

- $\alpha, \alpha_i, \beta, \beta_i$ are pure-past LTL formulas (pLTL)
- $\rightarrow$ denotes set inclusion

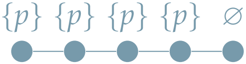
Theorem

Reactivity $= [\![\mathsf{LTL}]\!]$

Zohar Manna and Amir Pnueli (1990). "A hierarchy of temporal properties (invited paper, 1989)". In: *Proceedings of the 9th annual ACM symposium on Principles of distributed computing*, pp. 377–410. DOI: 10.1145/93385.93442

Kupferman and Vardi's Classification of the safety properties of LTL

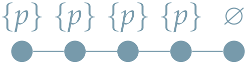
Consider the formula $\mathsf{G}(p)$. The following trace is a *bad prefix*:



Recall that $\sigma \in \Sigma^*$ is a bad prefix for a language $\mathcal{L}$ iff $\sigma \cdot \sigma' \notin \mathcal{L}$, for all $\sigma' \in \Sigma^\omega$.

Classification of Safety Properties
by Kupferman and Vardi

Consider the formula $\mathsf{G}(p)$. The following trace is a *bad prefix*:



Recall that $\sigma \in \Sigma^*$ is a bad prefix for a language $\mathcal{L}$ iff $\sigma \cdot \sigma' \notin \mathcal{L}$, for all $\sigma' \in \Sigma^\omega$. Consider now the formula $\mathsf{G}(p \vee (\mathsf{X}q \wedge \mathsf{X}\neg q))$.

- it is equivalent to $\mathsf{G}(p)$
- therefore, it is a safety formula
- its set of *bad prefixes* is the same as the one of $\mathsf{G}(p)$

Consider the formula $\mathsf{G}(p)$. The following trace is a *bad prefix*:

$$\{p\} \quad \{p\} \quad \{p\} \quad \{p\} \quad \varnothing$$

Recall that $\sigma \in \Sigma^*$ is a bad prefix for a language $\mathcal{L}$ iff $\sigma \cdot \sigma' \notin \mathcal{L}$, for all $\sigma' \in \Sigma^\omega$. Consider now the formula $\mathsf{G}(p \vee (\mathsf{X}q \wedge \mathsf{X}\neg q))$.

- it is equivalent to $\mathsf{G}(p)$
- therefore, it is a safety formula
- its set of *bad prefixes* is the same as the one of $\mathsf{G}(p)$

Nevertheless, the previous prefix does *not* tell the whole story about the violation of $\mathsf{G}(p \vee (\mathsf{X}q \wedge \mathsf{X}\neg q))$. In fact:

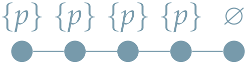
- Negation of the above formula:

  $$\mathsf{F}(\neg p \wedge (\mathsf{X}\neg q \vee \mathsf{X}q))$$

- Any violation depends on the fact that at certain point:
  - p is false and
  - in the *next* state q or $\neg q$ holds. (*this is always true*)
- In the previous prefix, the point in which $\neg p$ holds does *not* have a successor:
  - the prefix is *not informative*

Classification of Safety Properties
by Kupferman and Vardi

Consider the formula $\mathsf{G}(p)$. The following trace is a *bad prefix*:



Recall that $\sigma \in \Sigma^*$ is a bad prefix for a language $\mathcal{L}$ iff $\sigma \cdot \sigma' \notin \mathcal{L}$, for all $\sigma' \in \Sigma^\omega$. Consider now the formula $\mathsf{G}(p \vee (\mathsf{X}q \wedge \mathsf{X}\neg q))$.

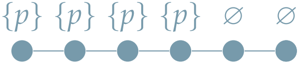
- it is equivalent to $\mathsf{G}(p)$
- therefore, it is a safety formula
- its set of *bad prefixes* is the same as the one of $\mathsf{G}(p)$

Nevertheless, the previous prefix does *not* tell the whole story about the violation of $\mathsf{G}(p \vee (\mathsf{X}q \wedge \mathsf{X}\neg q))$. In fact:

- Negation of the above formula:

  $$\mathsf{F}(\neg p \wedge (\mathsf{X}\neg q \vee \mathsf{X}q))$$

- This prefix is *informative* for the formula:



Classification of Safety Properties
by Kupferman and Vardi

- Consider the specification:

$$\mathsf{G}(p \lor (\mathsf{X}q \land \phi \land \mathsf{X}\neg q))$$

  where ϕ is a very complex Boolean formula.

- If the user is given the prefix

$$\{p\}\ \{p\}\ \{p\}\ \{p\}\ \varnothing$$

  then it is very hard for him/her to notice that the specification contains a redundant part ($\mathsf{X}q \land \mathsf{X}\neg q$).

- If instead the user is given this prefix

$$\{p\}\ \{p\}\ \{p\}\ \{p\}\ \varnothing\ \varnothing$$

  then he/she
  - notice that the *first* state in which $\neg p$ holds has a successor
  - inspect the parts of the specification that talk about the successor state ($\mathsf{X}q \land \mathsf{X}\neg q$)
  - notice that they are *redundant*
  - and finally remove them.

- This intuition of a prefix that *"tells the whole story"* is the base for a classification of safety properties in three distinct safety levels.
- This intuition is formalized by defining the notion of *informative prefix*
  - it is based on the semantics of LTL over finite traces

Reference:

Orna Kupferman and Moshe Y Vardi (2001). "Model checking of safety properties". In: *Formal Methods in System Design* 19.3, pp. 291–314. DOI: 10.1023/A:1011254632723

Usage:

- Detect the cause of inconsistent specifications:
  - e.g.: in formulas like $\mathsf{G}(p \vee (\mathsf{X}q \wedge \phi \wedge \mathsf{X}\neg q))$, the cause of inconsistency may not be easy to notice by the user, especially in more complicated examples
- Efficient automata construction
  - The automaton that recognizes all and only the informative prefixes of a formula is *exponentially smaller* than the automaton recognizing all and only the bad prefixes.
  - $\Rightarrow$ Efficient algorithms for model checking

Reference:

Orna Kupferman and Moshe Y Vardi (2001). "Model checking of safety properties". In: *Formal Methods in System Design* 19.3, pp. 291–314. DOI: 10.1023/A:1011254632723

Classification of Safety Properties
by Kupferman and Vardi

Recall that $\mathtt{nnf}(\psi)$ is the *negation normal form* of ψ, that is, a formula equivalent to ψ but with negations only applied to atomic propositions.

We define a new semantics for LTL interpreted over finite traces, that we denote with $\models_{\mathrm{KV}}$.

- $\sigma, i \models_{\mathrm{KV}} p$ iff $p \in \sigma_i$
- $\sigma, i \models_{\mathrm{KV}} \phi_1 \vee \phi_2$ iff $\sigma, i \models_{\mathrm{KV}} \phi_1$ or $\sigma, i \models_{\mathrm{KV}} \phi_2$
- $\sigma, i \models_{\mathrm{KV}} \phi_1 \wedge \phi_2$ iff $\sigma, i \models_{\mathrm{KV}} \phi_1$ and $\sigma, i \models_{\mathrm{KV}} \phi_2$
- $\sigma, i \models_{\mathrm{KV}} \mathsf{X}\phi$ iff $i+1 < |\sigma|$ and $\sigma, i+1 \models_{\mathrm{KV}} \phi$

Recall that $\mathtt{nnf}(\psi)$ is the *negation normal form* of ψ, that is, a formula equivalent to ψ but with negations only applied to atomic propositions.

We define a new semantics for LTL interpreted over finite traces, that we denote with $\models_{\mathrm{KV}}$.

- $\sigma, i \models_{\mathrm{KV}} p$ iff $p \in \sigma_i$
- $\sigma, i \models_{\mathrm{KV}} \phi_1 \vee \phi_2$ iff $\sigma, i \models_{\mathrm{KV}} \phi_1$ or $\sigma, i \models_{\mathrm{KV}} \phi_2$
- $\sigma, i \models_{\mathrm{KV}} \phi_1 \wedge \phi_2$ iff $\sigma, i \models_{\mathrm{KV}} \phi_1$ and $\sigma, i \models_{\mathrm{KV}} \phi_2$
- $\sigma, i \models_{\mathrm{KV}} \mathsf{X}\phi$ iff $i + 1 < |\sigma|$ and $\sigma, i + 1 \models_{\mathrm{KV}} \phi$
- $\sigma, i \models_{\mathrm{KV}} \mathsf{F}\phi$ iff $\exists i \leq j < |\sigma|$ and $\sigma, j \models_{\mathrm{KV}} \phi$
- $\sigma, i \models_{\mathrm{KV}} \mathsf{G}\phi$ is always false

Recall that $\mathtt{nnf}(\psi)$ is the *negation normal form* of ψ, that is, a formula equivalent to ψ but with negations only applied to atomic propositions.

We define a new semantics for LTL interpreted over finite traces, that we denote with $\models_{\mathrm{KV}}$.

- $\sigma, i \models_{\mathrm{KV}} p$ iff $p \in \sigma_i$
- $\sigma, i \models_{\mathrm{KV}} \phi_1 \vee \phi_2$ iff $\sigma, i \models_{\mathrm{KV}} \phi_1$ or $\sigma, i \models_{\mathrm{KV}} \phi_2$
- $\sigma, i \models_{\mathrm{KV}} \phi_1 \wedge \phi_2$ iff $\sigma, i \models_{\mathrm{KV}} \phi_1$ and $\sigma, i \models_{\mathrm{KV}} \phi_2$
- $\sigma, i \models_{\mathrm{KV}} \mathsf{X}\phi$ iff $i + 1 < |\sigma|$ and $\sigma, i + 1 \models_{\mathrm{KV}} \phi$
- $\sigma, i \models_{\mathrm{KV}} \phi_1 \mathsf{U} \phi_2$ iff $\exists i \leq j < |\sigma| \,.\, \sigma, j \models_{\mathrm{KV}} \phi_2$ and $\forall i \leq k < j \,.\, \sigma, k \models_{\mathrm{KV}} \phi_1$
- $\sigma, i \models_{\mathrm{KV}} \phi_1 \mathsf{R} \phi_2$ iff $\exists i \leq j < |\sigma| \,.\, \sigma, j \models_{\mathrm{KV}} \phi_1$ and $\forall i \leq k < j \,.\, \sigma, k \models_{\mathrm{KV}} \phi_2$

Intuition:

If $\sigma \models_{\mathrm{KV}} \mathtt{nnf}(\neg\phi)$, then σ carries all the information to violate ϕ over infinite traces.

Remark

The definition of $\models_{\mathrm{KV}}$ is exactly the one used in *Bounded Model Checking* for defining the truth of an LTL formula over a finite trace.

Reference:

Armin Biere et al. (2003). "Bounded model checking". In: *Adv. Comput.* 58, pp. 117–148. DOI: 10.1016/S0065-2458(03)58003-2. URL: https://doi.org/10.1016/S0065-2458(03)58003-2

Definition (Informative Prefix)

Let ϕ be an LTL formula over $\mathcal{AP}$ and let $\sigma \in (2^{\mathcal{AP}})^+$ be a finite trace over $2^{\mathcal{AP}}$.

σ is an *informative prefix for* ϕ

iff

$\sigma \models_{\mathrm{KV}} \mathtt{nnf}(\neg\phi)$

Note: in the original paper by Kupferman and Vardi, informative prefixes are defined using a mapping L. This is equivalent to our definition.

Definition (Informative Prefix)

Let ϕ be an LTL formula over $\mathcal{AP}$ and let $\sigma \in (2^{\mathcal{AP}})^{+}$ be a finite trace over $2^{\mathcal{AP}}$.

σ is an *informative prefix for* ϕ
iff
$\sigma \models_{\mathrm{KV}} \mathtt{nnf}(\neg\phi)$

Example:

This prefix is *informative for* $\mathsf{G}(p)$.

$$\{p\} \quad \{p\} \quad \{p\} \quad \{p\} \quad \varnothing$$

$\mathtt{nnf}(\neg\mathsf{G}(p)) := \mathsf{F}(\neg p)$

Classification of Safety Properties
by Kupferman and Vardi

Definition (Informative Prefix)

Let ϕ be an LTL formula over $\mathcal{AP}$ and let $\sigma \in (2^{\mathcal{AP}})^+$ be a finite trace over $2^{\mathcal{AP}}$.

σ is an *informative prefix for* ϕ

iff

$$\sigma \models_{\mathrm{KV}} \mathtt{nnf}(\neg\phi)$$

Example:

This prefix is *<u>not</u> informative for* $\phi := \mathsf{G}(p \vee (\mathsf{X}q \wedge \mathsf{X}\neg q))$.

$$\{p\} \quad \{p\} \quad \{p\} \quad \{p\} \quad \varnothing$$

$\mathtt{nnf}(\neg\phi) := \mathsf{F}(\neg p \wedge (\mathsf{X}\neg q \vee \mathsf{X}q))$

Classification of Safety Properties
by Kupferman and Vardi

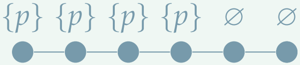
Definition (Informative Prefix)

Let ϕ be an LTL formula over $\mathcal{AP}$ and let $\sigma \in (2^{\mathcal{AP}})^+$ be a finite trace over $2^{\mathcal{AP}}$.

$$\sigma \text{ is an } \textit{informative prefix for } \phi$$
$$\text{iff}$$
$$\sigma \models_{\mathrm{KV}} \mathtt{nnf}(\neg\phi)$$

Example:

This prefix is *informative for* $\phi := \mathsf{G}(p \vee (\mathsf{X}q \wedge \mathsf{X}\neg q))$.



$\mathtt{nnf}(\neg\phi) := \mathsf{F}(\neg p \wedge (\mathsf{X}\neg q \vee \mathsf{X}q))$

Classification of Safety Properties
by Kupferman and Vardi

Definition (Informative Prefix)

Let ϕ be an LTL formula over $\mathcal{AP}$ and let $\sigma \in (2^{\mathcal{AP}})^+$ be a finite trace over $2^{\mathcal{AP}}$.

$$\sigma \text{ is an } \textit{informative prefix for } \phi$$
$$\text{iff}$$
$$\sigma \models_{\mathrm{KV}} \mathtt{nnf}(\neg\phi)$$

Example:

This prefix is *<u>not</u> informative for* $\phi := \big(\mathsf{G}(q \vee \mathsf{FG}p) \wedge \mathsf{G}(r \vee \mathsf{FG}\neg p)\big) \vee \mathsf{G}q \vee \mathsf{G}r$.

$$\{p\} \quad \{p\} \quad \{p\} \quad \{p\} \quad \varnothing \quad \varnothing$$

$$\mathtt{nnf}(\neg\phi) := \big(\mathsf{F}(\neg q \wedge \mathsf{GF}\neg p) \vee \mathsf{F}(\neg r \wedge \mathsf{GF}p)\big) \wedge \mathsf{F}\neg q \wedge \mathsf{F}\neg r$$

Definition (Informative Prefix)

Let ϕ be an LTL formula over $\mathcal{AP}$ and let $\sigma \in (2^{\mathcal{AP}})^+$ be a finite trace over $2^{\mathcal{AP}}$.

$$\sigma \text{ is an } \textit{informative prefix for } \phi$$
$$\text{iff}$$
$$\sigma \models_{\mathrm{KV}} \mathtt{nnf}(\neg\phi)$$

Example:

This prefix is *not informative for* $\phi := (\mathsf{G}(q \vee \mathsf{FG}p) \wedge \mathsf{G}(r \vee \mathsf{FG}\neg p)) \vee \mathsf{G}q \vee \mathsf{G}r$.

$\mathsf{G}(\dots)$ is always false under $\models_{\mathrm{KV}}$: no prefix is informative for ϕ

$\mathtt{nnf}(\neg\phi) := (\mathsf{F}(\neg q \wedge \mathsf{GF}\neg p) \vee \mathsf{F}(\neg r \wedge \mathsf{GF}p)) \wedge \mathsf{F}\neg q \wedge \mathsf{F}\neg r$

Definition (Informative Prefix)

Let ϕ be an LTL formula over $\mathcal{AP}$ and let $\sigma \in (2^{\mathcal{AP}})^+$ be a finite trace over $2^{\mathcal{AP}}$.

σ is an *informative prefix for* ϕ
iff
$\sigma \models_{\mathrm{KV}} \mathtt{nnf}(\neg\phi)$

Remark:

Given σ and ϕ, checking whether $\sigma \models_{\mathrm{KV}} \phi$ can be done in time $\mathcal{O}(|\sigma| \cdot |\phi|)$.

Let ϕ be any LTL formula such that $\mathcal{L}(\phi)$ is a safety language. The definition of informative prefix is used to classify such formulas ϕ into three types:

1. intentionally safe
2. accidentally safe
3. pathologically safe

Let ϕ be any LTL formula such that $\mathcal{L}(\phi)$ is a safety language. The definition of informative prefix is used to classify such formulas ϕ into three types:

1. intentionally safe

ϕ is intentionally safe iff *all* bad prefixes are informative.

For example:

- the formula $\mathsf{G}(p)$ is intentionally safe.
- the formula $\mathsf{G}(p \vee (\mathsf{X}q \wedge \mathsf{X}\neg q))$ is *not* intentionally safe, because $\langle \{p\}, \{p\}, \{p\}, \{p\}, \varnothing \rangle$ is a bad prefix but it is not informative.

2. accidentally safe
3. pathologically safe

Let ϕ be any LTL formula such that $\mathcal{L}(\phi)$ is a safety language. The definition of informative prefix is used to classify such formulas ϕ into three types:

1. intentionally safe
2. accidentally safe

ϕ is accidentally safe iff *(i)* not all the bad prefixes of ψ are informative, but *(ii)* every $\sigma \in (2^{AP})^\omega$ that violates ϕ has an informative bad prefix.

For example:

- $\mathsf{G}(p \vee (\mathsf{X}q \wedge \mathsf{X}\neg q))$ is accidentally safe: $\langle \{p\}, \{p\}, \{p\}, \{p\}, \varnothing \rangle$ is a bad prefix but it is not informative. However, every infinite trace violating the formula has an informative prefix of type $\{p\}^* \cdot \varnothing \cdot \varnothing$.

3. pathologically safe

Let ϕ be any LTL formula such that $\mathcal{L}(\phi)$ is a safety language. The definition of informative prefix is used to classify such formulas ϕ into three types:

1. intentionally safe
2. accidentally safe
3. pathologically safe

ϕ is pathologically safe iff there is a $\sigma \in (2^{\mathcal{AP}})^\omega$ that violates ϕ and has no informative bad prefixes.

For example:

- $(\mathsf{G}(q \vee \mathsf{FG}p) \wedge \mathsf{G}(r \vee \mathsf{FG}\neg p)) \vee \mathsf{G}q \vee \mathsf{G}r$
  - the computation $\varnothing^\omega$ violates the formula

    $$\varnothing^\omega \models (\mathsf{F}(\neg q \wedge \mathsf{GF}\neg p) \vee \mathsf{F}(\neg r \wedge \mathsf{GF}p)) \wedge \mathsf{F}(\neg q) \wedge \mathsf{F}(\neg r)$$

  - but each of its prefixes σ is *not informative* because $\sigma \not\models_{\mathrm{KV}} (\mathsf{F}(\neg q \wedge \mathsf{GF}\neg p) \vee \mathsf{F}(\neg r \wedge \mathsf{GF}p)) \wedge \mathsf{F}(\neg q) \wedge \mathsf{F}(\neg r)$, but no finite prefix is such.

Classification of Safety Properties
by Kupferman and Vardi

Let ϕ be any LTL formula such that $\mathcal{L}(\phi)$ is a safety language. The definition of informative prefix is used to classify such formulas ϕ into three types:

1. intentionally safe
2. accidentally safe
3. pathologically safe

Formulas that are accidentally safe or pathologically safe are *needlessly complicated*:

- They contain a redundancy that can be eliminated.
- If a user wrote a pathologically safe formula, then probably he/she didn't mean to write a safety formula.
- This classification helps in detecting inconsistent or redundant specifications.

Theorem

For any formula ϕ of SafetyLTL*, it holds that ϕ is either intentionally or accidentally safe.*

Proof.

- By the duality between SafetyLTL and coSafetyLTL, we have that $\mathtt{nnf}(\neg\phi)$ is a formula of coSafetyLTL and is equivalent to ϕ. Let $\psi := \mathtt{nnf}(\neg\phi)$.
- Let $\sigma = \langle\sigma_0, \sigma_1, \ldots\rangle$ be an infinite trace that satisfies ψ, that is $\sigma \models \psi$.
- Since, by definition of coSafetyLTL, ψ contains only X and U as temporal operators, there exists a furthermost time point i such that $\sigma_{[0,i]} \models \psi$ (under finite traces semantics).

Theorem

For any formula ϕ of SafetyLTL*, it holds that ϕ is either intentionally or accidentally safe.*

Proof.

- Since on the operators X and U the definitions of $\models$ and $\models_{\mathrm{KV}}$ coincide, we have also that $\sigma_{[0,i]} \models_{\mathrm{KV}} \psi$. Therefore, by definition, $\sigma_{[0,i]}$ is an *informative prefix*.
- It follows that every infinite trace that violates ϕ has an informative prefix, thus ϕ is either intentionally or accidentally safe. $\square$

As we will see, this classification is exploited for having efficient verification algorithms.

- An automaton that recognizes only the bad prefixes that are *informative* can be built exponentially more efficiently than the automaton for *all* the bad prefixes.
- Moreover, in practice, almost all the benefits that one can obtain from an automaton for the bad prefixes can also be obtained from an automaton for the *informative* bad prefixes.
  - for example, we can perform *model checking* algorithms considering only the informative bad prefixes
  - since there may be bad prefixes that are not informative but may become informative if extended, *minimality* of counterexamples is the only thing that is sacrified when dealing with informative bad prefixes.

REFERENCES

Bibliography I

Armin Biere et al. (2003). "Bounded model checking". In: *Adv. Comput.* 58, pp. 117–148. DOI: 10.1016/S0065-2458(03)58003-2. URL: https://doi.org/10.1016/S0065-2458(03)58003-2.

Edward Y. Chang, Zohar Manna, and Amir Pnueli (1992). "Characterization of Temporal Property Classes". In: *Proceedings of the 19th International Colloquium on Automata, Languages and Programming*. Ed. by Werner Kuich. Vol. 623. Lecture Notes in Computer Science. Springer, pp. 474–486. DOI: 10.1007/3-540-55719-9_97.

Bibliography II

Alessandro Cimatti et al. (2022). "A first-order logic characterisation of safety and co-safety languages". In: *Foundations of Software Science and Computation Structures - 25th International Conference, FOSSACS 2022, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2022, Munich, Germany, April 2-7, 2022, Proceedings*. Ed. by Patricia Bouyer and Lutz Schröder. Vol. 13242. Lecture Notes in Computer Science. Springer, pp. 244–263. DOI: 10.1007/978-3-030-99253-8_13. URL: https://doi.org/10.1007/978-3-030-99253-8%5C_13.

Orna Kupferman and Moshe Y Vardi (2001). "Model checking of safety properties". In: *Formal Methods in System Design* 19.3, pp. 291–314. DOI: 10.1023/A:1011254632723.

Zohar Manna and Amir Pnueli (1990). "A hierarchy of temporal properties (invited paper, 1989)". In: *Proceedings of the 9th annual ACM symposium on Principles of distributed computing*, pp. 377–410. DOI: 10.1145/93385.93442.

Bibliography III

Ina Schiering and Wolfgang Thomas (1996). "Counter-free automata, first-order logic, and star-free expressions extended by prefix oracles". In: *Developments in Language Theory, II (Magdeburg, 1995), Worl Sci. Publishing, River Edge, NJ*, pp. 166–175.