Parameterized Linear Temporal Logics Meet Costs: Still not Costlier than LTL

Martin Zimmermann

Saarland University

September 22nd, 2015

GandALF 2015, Genova, Italy

Motivation

Linear Temporal Logic (LTL) as specification language:

- Simple and variable-free syntax and intuitive semantics.
- Expressively equivalent to first-order logic on words.
- LTL model checking routinely applied in industrial settings.
- Desirable algorithmic properties.

Motivation

Linear Temporal Logic (LTL) as specification language:

- Simple and variable-free syntax and intuitive semantics.
- Expressively equivalent to first-order logic on words.
- LTL model checking routinely applied in industrial settings.
- Desirable algorithmic properties.

Shortcomings:

1. LTL cannot express timing constraints.

Motivation

Linear Temporal Logic (LTL) as specification language:

- Simple and variable-free syntax and intuitive semantics.
- Expressively equivalent to first-order logic on words.
- LTL model checking routinely applied in industrial settings.
- Desirable algorithmic properties.

Shortcomings:

1. LTL cannot express timing constraints.
2. LTL cannot express all ω-regular properties.

Motivation

Linear Temporal Logic (LTL) as specification language:

- Simple and variable-free syntax and intuitive semantics.
- Expressively equivalent to first-order logic on words.
- LTL model checking routinely applied in industrial settings.
- Desirable algorithmic properties.

Shortcomings:

1. LTL cannot express timing constraints.
    - Add $\mathbf{F}_{\leq k}$ for $k \in \mathbb{N}$.
2. LTL cannot express all ω-regular properties.

Motivation

Linear Temporal Logic (LTL) as specification language:

- Simple and variable-free syntax and intuitive semantics.
- Expressively equivalent to first-order logic on words.
- LTL model checking routinely applied in industrial settings.
- Desirable algorithmic properties.

Shortcomings:

1. LTL cannot express timing constraints.
   - Add $\mathbf{F}_{\leq k}$ for $k \in \mathbb{N}$. Not practical (i.e., which k is right?)
2. LTL cannot express all ω-regular properties.

Motivation

Linear Temporal Logic (LTL) as specification language:

- Simple and variable-free syntax and intuitive semantics.
- Expressively equivalent to first-order logic on words.
- LTL model checking routinely applied in industrial settings.
- Desirable algorithmic properties.

Shortcomings:

1. LTL cannot express timing constraints.
   - Add $\mathbf{F}_{\leq k}$ for $k \in \mathbb{N}$. Not practical (i.e., which k is right?)
   - Add $\mathbf{F}_{\leq x}$ for variable x.
2. LTL cannot express all ω-regular properties.

Motivation

Linear Temporal Logic (LTL) as specification language:

- Simple and variable-free syntax and intuitive semantics.
- Expressively equivalent to first-order logic on words.
- LTL model checking routinely applied in industrial settings.
- Desirable algorithmic properties.

Shortcomings:

1. LTL cannot express timing constraints.
   - Add $\mathbf{F}_{\leq k}$ for $k \in \mathbb{N}$. Not practical (i.e., which k is right?)
   - Add $\mathbf{F}_{\leq x}$ for variable x. Now: does there exist a valuation for x s.t. specification is satisfied?
2. LTL cannot express all ω-regular properties.

Motivation

Linear Temporal Logic (LTL) as specification language:

- Simple and variable-free syntax and intuitive semantics.
- Expressively equivalent to first-order logic on words.
- LTL model checking routinely applied in industrial settings.
- Desirable algorithmic properties.

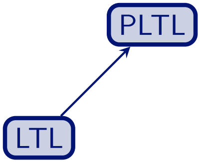
Shortcomings:

1. LTL cannot express timing constraints.
   - Add $\mathbf{F}_{\leq k}$ for $k \in \mathbb{N}$. Not practical (i.e., which k is right?)
   - Add $\mathbf{F}_{\leq x}$ for variable x. Now: does there exist a valuation for x s.t. specification is satisfied?
2. LTL cannot express all ω-regular properties.
   - Many extensions that are equivalent to ω-regular languages: add regular expression-, grammar-, or automata-operators to LTL.

Overview

LTL

Overview



Parametric LTL

Alur et al. '99: add parameterized operators to LTL

$$\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathbf{X}\varphi \mid \varphi \mathbf{U} \varphi \mid \varphi \mathbf{R} \varphi \mid \mathbf{F}_{\leq x}\varphi \mid \mathbf{G}_{\leq y}\varphi$$

with $x \in \mathcal{X}$, $y \in \mathcal{Y}$ ($\mathcal{X} \cap \mathcal{Y} = \emptyset$).

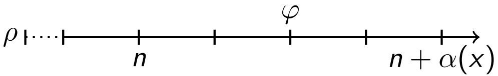
Parametric LTL

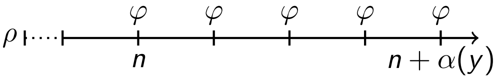
Alur et al. '99: add parameterized operators to LTL

$$\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathbf{X}\varphi \mid \varphi \mathbf{U} \varphi \mid \varphi \mathbf{R} \varphi \mid \mathbf{F}_{\leq x}\varphi \mid \mathbf{G}_{\leq y}\varphi$$

with $x \in \mathcal{X}$, $y \in \mathcal{Y}$ ($\mathcal{X} \cap \mathcal{Y} = \emptyset$).

Semantics w.r.t. variable valuation $\alpha \colon \mathcal{X} \cup \mathcal{Y} \to \mathbb{N}$:

- As usual for LTL operators.
- $(\rho, n, \alpha) \models \mathbf{F}_{\leq x}\varphi$: 

- $(\rho, n, \alpha) \models \mathbf{G}_{\leq y}\varphi$: 

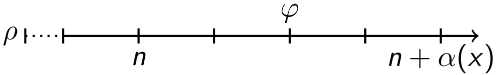
Parametric LTL

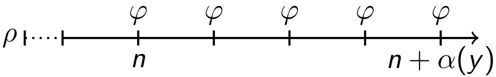
Alur et al. '99: add parameterized operators to LTL

$$\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathbf{X}\varphi \mid \varphi \mathbf{U} \varphi \mid \varphi \mathbf{R} \varphi \mid \mathbf{F}_{\leq x}\varphi \mid \mathbf{G}_{\leq y}\varphi$$

with $x \in \mathcal{X}$, $y \in \mathcal{Y}$ ($\mathcal{X} \cap \mathcal{Y} = \emptyset$).

Semantics w.r.t. variable valuation $\alpha \colon \mathcal{X} \cup \mathcal{Y} \to \mathbb{N}$:

- As usual for LTL operators.
- $(\rho, n, \alpha) \models \mathbf{F}_{\leq x}\varphi$: 

- $(\rho, n, \alpha) \models \mathbf{G}_{\leq y}\varphi$: 

Example:

$$\mathbf{G}(\mathit{req} \to \mathbf{F}_{\leq x}\mathit{resp})$$

Results

Model Checking: Does there exist an α such that every execution satisfies the specification w.r.t. α?

Results

Model Checking: Does there exist an α such that every execution satisfies the specification w.r.t. α?

Theorem (Alur et al. '99, Kupferman et al. 06')

PLTL model checking is $\textsc{PSpace}$*-complete.*

Results

Model Checking: Does there exist an α such that every execution satisfies the specification w.r.t. α?

Theorem (Alur et al. '99, Kupferman et al. 06')

PLTL model checking is $\textsc{PSpace}$*-complete.*

Infinite Games: Does there exist an α and a strategy σ for Player 0 such that every play that is consistent with σ satisfies the specification w.r.t. α?

Results

Model Checking: Does there exist an α such that every execution satisfies the specification w.r.t. α?

Theorem (Alur et al. '99, Kupferman et al. 06')

PLTL model checking is $\textsc{PSpace}$*-complete.*

Infinite Games: Does there exist an α and a strategy σ for Player 0 such that every play that is consistent with σ satisfies the specification w.r.t. α?

Theorem (Kupferman et al. 06', Z. '11)

Solving PLTL games is $2\textsc{ExpTime}$*-complete.*

Results

Model Checking: Does there exist an α such that every execution satisfies the specification w.r.t. α?

Theorem (Alur et al. '99, Kupferman et al. 06')

PLTL model checking is $\textsc{PSpace}$*-complete.*

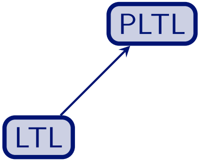
Infinite Games: Does there exist an α and a strategy σ for Player 0 such that every play that is consistent with σ satisfies the specification w.r.t. α?

Theorem (Kupferman et al. 06', Z. '11)

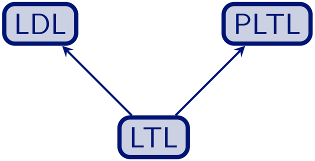
Solving PLTL games is $2\textsc{ExpTime}$*-complete.*

Parameterized operators can be added for free!

Overview



Overview



Linear Dynamic Logic

Vardi '11: Another extension of LTL expressing exactly the ω-regular languages: use PDL-like operators

$$\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \langle r \rangle \varphi \mid [r] \varphi$$
$$r ::= \phi \mid \varphi ? \mid r + r \mid r \,;\, r \mid r^*$$

where ϕ ranges over boolean formulas over atomic propositions.

Linear Dynamic Logic

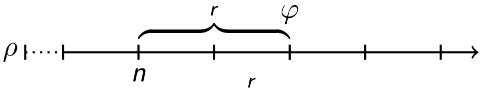
Vardi '11: Another extension of LTL expressing exactly the ω-regular languages: use PDL-like operators

$$\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \langle r \rangle \varphi \mid [r]\varphi$$
$$r ::= \phi \mid \varphi? \mid r + r \mid r\,;r \mid r^*$$

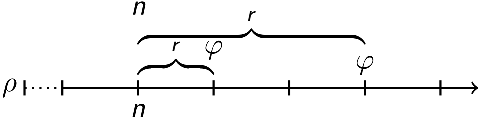
where ϕ ranges over boolean formulas over atomic propositions.

Semantics:

- $(\rho, n) \models \langle r \rangle \varphi$:



- $(\rho, n) \models [r]\varphi$:



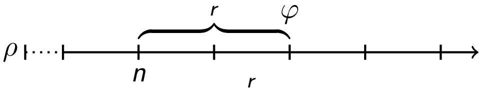
Linear Dynamic Logic

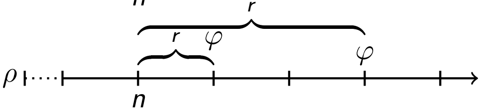
Vardi '11: Another extension of LTL expressing exactly the ω-regular languages: use PDL-like operators

$$\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \langle r \rangle \varphi \mid [r] \varphi$$
$$r ::= \phi \mid \varphi? \mid r + r \mid r ; r \mid r^*$$

where ϕ ranges over boolean formulas over atomic propositions.

Semantics:

- $(\rho, n) \models \langle r \rangle \varphi$: 

- $(\rho, n) \models [r] \varphi$: 

Example:

$$[\texttt{tt}^*] (req \rightarrow \langle (\texttt{tt} ; \texttt{tt})^* \rangle\, resp)$$

Results

Theorem (Vardi '11)

LDL defines exactly the ω-regular languages.

Results

Theorem (Vardi '11)

LDL defines exactly the ω-regular languages.

Theorem (Vardi '11)

LDL can be translated into linearly-sized alternating automata.

Results

Theorem (Vardi '11)

LDL defines exactly the ω-regular languages.

Theorem (Vardi '11)

LDL can be translated into linearly-sized alternating automata.

Corollary

1. *LDL model checking is* $\textsc{PSpace}$*-complete.*
2. *Solving LDL games is* $2\textsc{ExpTime}$*-complete.*

Results

Theorem (Vardi '11)

LDL defines exactly the ω-regular languages.

Theorem (Vardi '11)

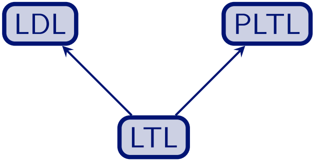
LDL can be translated into linearly-sized alternating automata.

Corollary

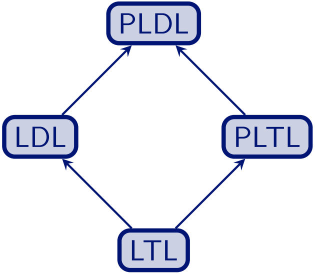
1. *LDL model checking is* $\textsc{PSpace}$*-complete.*
2. *Solving LDL games is* $2\textsc{ExpTime}$*-complete.*

Expressivity can be increased for free!

Overview



Overview



Parametric LDL

Faymonville, Z. '14: add parameterized operators to LDL.

$$\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \langle r \rangle \varphi \mid [r] \varphi \mid \langle r \rangle_{\leq x} \varphi \mid [r]_{\leq y} \varphi$$

$$r ::= \phi \mid \varphi? \mid r + r \mid r \,;\, r \mid r^*$$

Parametric LDL

Faymonville, Z. '14: add parameterized operators to LDL.

$$\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \langle r \rangle \varphi \mid [r]\varphi \mid \langle r \rangle_{\leq x}\varphi \mid [r]_{\leq y}\varphi$$

$$r ::= \phi \mid \varphi? \mid r + r \mid r\,;r \mid r^*$$

Example:

$$[\mathtt{tt}^*]\,(req \rightarrow \langle(\mathtt{tt}\,;\mathtt{tt})^*\rangle_{\leq x} resp)$$

Parametric LDL

Faymonville, Z. '14: add parameterized operators to LDL.

$$\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \langle r \rangle \varphi \mid [r] \varphi \mid \langle r \rangle_{\leq x} \varphi \mid [r]_{\leq y} \varphi$$

$$r ::= \phi \mid \varphi ? \mid r + r \mid r \,;\, r \mid r^*$$

Example:

$$[\mathtt{tt}^*] (req \rightarrow \langle (\mathtt{tt}\,;\mathtt{tt})^* \rangle_{\leq x} resp)$$

Theorem (Faymonville, Z. '14)

PLDL model checking is $\mathrm{PS}\textsc{pace}$*-complete.*

Parametric LDL

Faymonville, Z. '14: add parameterized operators to LDL.

$$\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \langle r \rangle \varphi \mid [r] \varphi \mid \langle r \rangle_{\leq x} \varphi \mid [r]_{\leq y} \varphi$$

$$r ::= \phi \mid \varphi? \mid r + r \mid r\,;r \mid r^*$$

Example:

$$[\mathtt{tt}^*](req \rightarrow \langle(\mathtt{tt}\,;\mathtt{tt})^*\rangle_{\leq x} resp)$$

Theorem (Faymonville, Z. '14)

PLDL model checking is PSPACE*-complete.*

Theorem (Faymonville, Z. '14)

Solving PLDL games is 2EXPTIME*-complete.*

Parametric LDL

Faymonville, Z. '14: add parameterized operators to LDL.

$$\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \langle r \rangle \varphi \mid [r] \varphi \mid \langle r \rangle_{\leq x} \varphi \mid [r]_{\leq y} \varphi$$

$$r ::= \phi \mid \varphi? \mid r + r \mid r\,;r \mid r^*$$

Example:

$$[\mathtt{tt}^*](req \rightarrow \langle (\mathtt{tt}\,;\mathtt{tt})^* \rangle_{\leq x} resp)$$

Theorem (Faymonville, Z. '14)

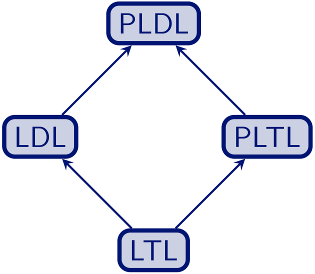
PLDL model checking is PSPACE*-complete.*

Theorem (Faymonville, Z. '14)

Solving PLDL games is 2EXPTIME*-complete.*

Parameterized operators can be added and expressivity can be increased for free!

Overview



Beyond Bounding Time: Costs

- Model checking and solving games for PLTL and PLDL are boundedness problems.
- Recently, boundedness problems have received a lot of attention:
  - Automata with counters and quantitative logics
  - finitary parity, parity with costs, energy-parity, etc.

Beyond Bounding Time: Costs

- Model checking and solving games for PLTL and PLDL are boundedness problems.
- Recently, boundedness problems have received a lot of attention:
  - Automata with counters and quantitative logics
  - finitary parity, parity with costs, energy-parity, etc.

Example: Parity games with costs:

- Label arena with costs, i.e., $\mathrm{cst}\colon E \to \mathbb{N}$.
- Condition: there exists a b s.t. almost every occurrence of some odd color is followed by occurrence of larger even color s.t. cost between occurrences is at most b.

Beyond Bounding Time: Costs

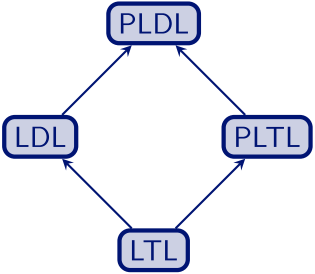
- Model checking and solving games for PLTL and PLDL are boundedness problems.
- Recently, boundedness problems have received a lot of attention:
  - Automata with counters and quantitative logics
  - finitary parity, parity with costs, energy-parity, etc.

Example: Parity games with costs:

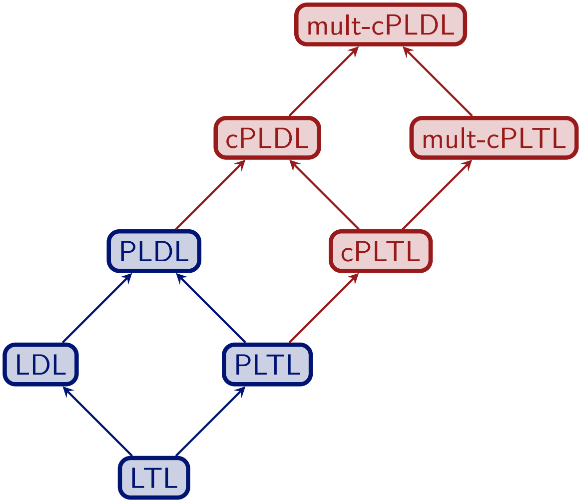
- Label arena with costs, i.e., $\mathrm{cst}\colon E \to \mathbb{N}$.
- Condition: there exists a b s.t. almost every occurrence of some odd color is followed by occurrence of larger even color s.t. cost between occurrences is at most b.

This is not expressible in PLTL or PLDL.

Overview



Overview

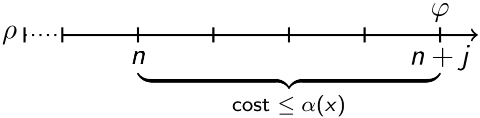


PLTL and PLDL with Costs

Syntax: As for PLTL respectively PLDL.

Semantics: Label edges by costs, i.e., cst: $E \to \mathbb{N}$, and bound cost instead of time, e.g.,

$(\rho, n, \alpha) \models \mathbf{F}_{\leq x}\varphi$:

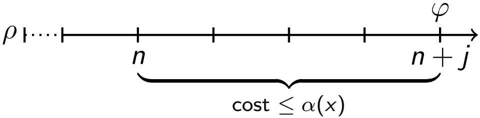


PLTL and PLDL with Costs

Syntax: As for PLTL respectively PLDL.

Semantics: Label edges by costs, i.e., $\mathrm{cst}\colon E \to \mathbb{N}$, and bound cost instead of time, e.g.,

$(\rho, n, \alpha) \models \mathbf{F}_{\leq x}\varphi$:



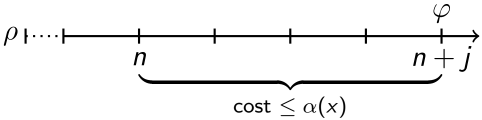
Note: j might be arbitrarily large, as we allow cost zero.

PLTL and PLDL with Costs

Syntax: As for PLTL respectively PLDL.

Semantics: Label edges by costs, i.e., $\mathrm{cst}\colon E \to \mathbb{N}$, and bound cost instead of time, e.g.,

$(\rho, n, \alpha) \models \mathbf{F}_{\leq x}\varphi$:



Note: j might be arbitrarily large, as we allow cost zero.

A multi-dimensional setting: mult-cPLTL and mult-cPLDL

- $\mathrm{cst}\colon E \to \mathbb{N}^d$, $d \in \mathbb{N}$.
- Label parameterized operators with coordinate $i \in \{1, \ldots, d\}$, e.g., $\mathbf{F}_{\leq_i x}$ and $\langle r \rangle_{\leq_i x}$

Results

Let $\mathcal{L} \in \{\text{cPLTL}, \text{cPLDL}, \text{mult-cPLTL}, \text{mult-cPLDL}\}$.

Theorem

$\mathcal{L}$ model checking is $\textsc{PSpace}$*-complete.*

Results

Let $\mathcal{L} \in \{\text{cPLTL}, \text{cPLDL}, \text{mult-cPLTL}, \text{mult-cPLDL}\}$.

Theorem

$\mathcal{L}$ model checking is $\textsc{PSpace}$*-complete.*

Theorem

Solving $\mathcal{L}$ games is $2\textsc{ExpTime}$*-complete.*

Results

Let $\mathcal{L} \in \{\text{cPLTL}, \text{cPLDL}, \text{mult-cPLTL}, \text{mult-cPLDL}\}$.

Theorem

$\mathcal{L}$ model checking is $\textsc{PSpace}$-complete.

Theorem

Solving $\mathcal{L}$ games is $2\textsc{ExpTime}$-complete.

Remark: The running times are independent of the largest cost, as we consider boundedness problems.

Results

Let $\mathcal{L} \in \{\text{cPLTL}, \text{cPLDL}, \text{mult-cPLTL}, \text{mult-cPLDL}\}$.

Theorem

$\mathcal{L}$ model checking is $\textsc{PSpace}$-complete.

Theorem

Solving $\mathcal{L}$ games is $2\textsc{ExpTime}$-complete.

Remark: The running times are independent of the largest cost, as we consider boundedness problems.

Going from bounding time to bounding (multi-dimensional) costs for free!

Optimization Problems

- Unipolar formulas: at most one type of parameterized operator
- Then: ask for optimal variable valuations
  - For $\mathbf{F}_{\leq x}$ and $\langle r \rangle_{\leq x}$: minimize $\alpha(x)$
  - For $\mathbf{G}_{\leq y}$ and $[r]_{\leq y}$: maximize $\alpha(y)$

Optimization Problems

- Unipolar formulas: at most one type of parameterized operator
- Then: ask for optimal variable valuations
  - For $\mathbf{F}_{\leq x}$ and $\langle r \rangle_{\leq x}$: minimize $\alpha(x)$
  - For $\mathbf{G}_{\leq y}$ and $[r]_{\leq y}$: maximize $\alpha(y)$

Theorem

1. *Tight exponential upper/lower bounds on optimal α for unipolar cPLDL model checking.*
2. *Tight doubly-exponential upper/lower bounds on optimal α for unipolar cPLDL games.*

Optimization Problems

- Unipolar formulas: at most one type of parameterized operator
- Then: ask for optimal variable valuations
  - For $\mathbf{F}_{\leq x}$ and $\langle r \rangle_{\leq x}$: minimize $\alpha(x)$
  - For $\mathbf{G}_{\leq y}$ and $[r]_{\leq y}$: maximize $\alpha(y)$

Theorem

1. *Tight exponential upper/lower bounds on optimal α for unipolar cPLDL model checking.*
2. *Tight doubly-exponential upper/lower bounds on optimal α for unipolar cPLDL games.*

Corollary

1. *Model checking optimization in polynomial space.*
2. *Game optimization in triply-exponential time.*

Proof Sketch (for PLTL Games)

1. Replacing $\mathbf{G}_{\leq y}\psi$ by ψ preserves satisfiability (monotonicity).
2. Apply alternating color technique **(Kupferman et al. '06)**:
   - Add new proposition p and replace every $\mathbf{F}_{\leq x}\psi$ by

$$(p \rightarrow p\mathbf{U}(\neg p\mathbf{U}\psi)) \wedge (\neg p \rightarrow \neg p\mathbf{U}(p\mathbf{U}\psi))$$

     (ψ satisfied within one color change), obtain $c(\varphi)$.

Proof Sketch (for PLTL Games)

1. Replacing $\mathbf{G}_{\leq y}\psi$ by ψ preserves satisfiability (monotonicity).
2. Apply alternating color technique **(Kupferman et al. '06)**:
   - Add new proposition p and replace every $\mathbf{F}_{\leq x}\psi$ by

     $$(p \rightarrow p\mathbf{U}(\neg p\mathbf{U}\psi)) \wedge (\neg p \rightarrow \neg p\mathbf{U}(p\mathbf{U}\psi))$$

     (ψ satisfied within one color change), obtain $c(\varphi)$.

Lemma

φ and $c(\varphi)$ "equivalent" on traces where distance between color changes is bounded.

Proof Sketch (for PLTL Games)

1. Replacing $\mathbf{G}_{\leq y}\psi$ by ψ preserves satisfiability (monotonicity).
2. Apply alternating color technique **(Kupferman et al. '06)**:
   - Add new proposition p and replace every $\mathbf{F}_{\leq x}\psi$ by

$$(p \rightarrow p\mathbf{U}(\neg p\mathbf{U}\psi)) \wedge (\neg p \rightarrow \neg p\mathbf{U}(p\mathbf{U}\psi))$$

(ψ satisfied within one color change), obtain $c(\varphi)$.

Lemma

φ and $c(\varphi)$ "equivalent" on traces where distance between color changes is bounded.

3. Emptiness for game with condition φ equivalent to Player 0 winning LTL game with condition $c(\varphi) \wedge \mathbf{GF}p \wedge \mathbf{GF}\neg p$, as finite state strategies bound distance between color changes.
4. Yields doubly-exponential upper bound.

Conclusion

Weighted extensions of parameterized linear temporal logics that retain the attractive algorithmic properties of LTL:

- Model checking $\textsc{PSpace}$-complete.
- Solving games $2\textsc{ExpTime}$-complete.

Also (in the one-dimensional case):

- Model checking optimization in polynomial space.
- Game optimization in triply-exponential time.

Conclusion

Weighted extensions of parameterized linear temporal logics that retain the attractive algorithmic properties of LTL:

- Model checking PSPACE-complete.
- Solving games 2EXPTIME-complete.

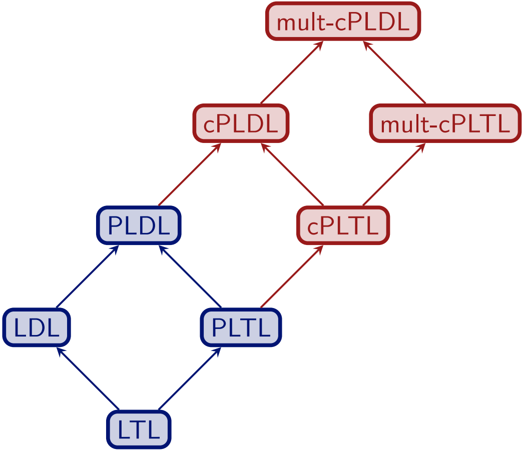
Also (in the one-dimensional case):

- Model checking optimization in polynomial space.
- Game optimization in triply-exponential time.

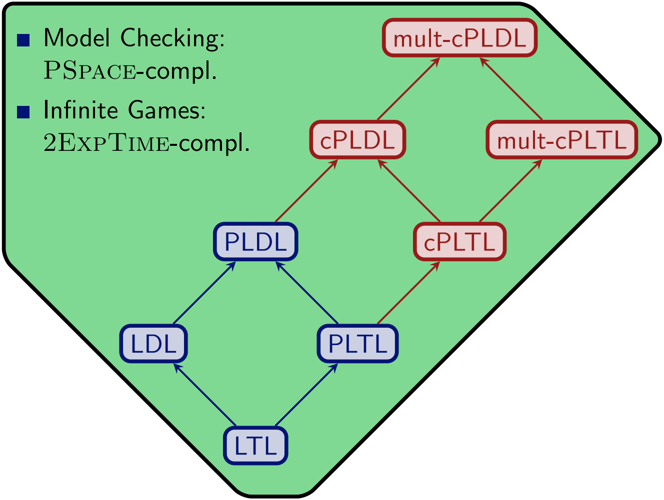
Open problems:

- Game optimization in doubly-exponential time.
- Multi-dimensional optimization problems.
- More general weight structures, e.g., negative weights, semi-rings, etc.

Overview

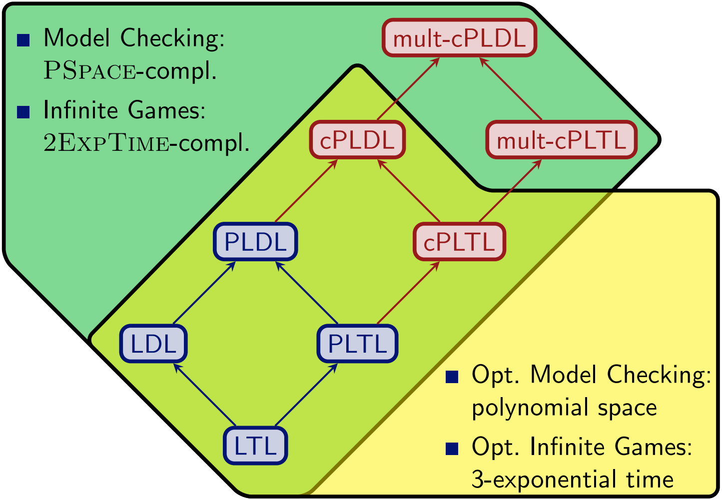


Overview



Model Checking: PSpace-compl.
Infinite Games: 2ExpTime-compl.
mult-cPLDL
cPLDL
mult-cPLTL
PLDL
cPLTL
LDL
PLTL
LTL

Overview


Model Checking: PSpace-compl.
Infinite Games: 2ExpTime-compl.
mult-cPLDL
cPLDL
mult-cPLTL
PLDL
cPLTL
LDL
PLTL
LTL
Opt. Model Checking: polynomial space
Opt. Infinite Games: 3-exponential time