Deciding Persistence or Recurrence Membership

Alexandre Gbaguidi Aïsse

LRDE

Supervisor: Alexandre Duret-Lutz

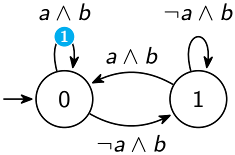
February 1, 2018

A significant portion of this presentation is inspired by Alexandre Duret-Lutz's contributions to the prcheck paper still on progress.

Context

1. Context
2. Deciding Recurrence via Deterministic Rabin Automata
3. Deciding Persistence via co-Büchi Automata
4. Comparison of both procedures

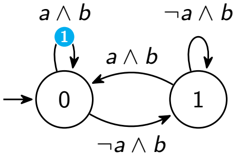
An ω-automaton $\mathcal{A} = \langle \Sigma, Q, q_0, M, \Delta, \phi \rangle$



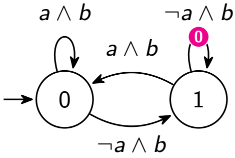
Büchi: Inf(1)

An ω-automaton $\mathcal{A} = \langle \Sigma, Q, q_0, M, \Delta, \phi \rangle$

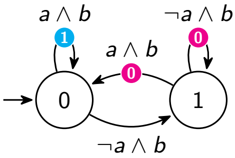
Büchi: Inf(1) co-Büchi: Fin(0)



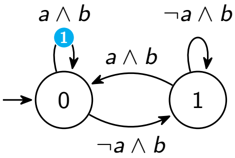
Büchi: Inf(1)



co-Büchi: Fin(0)

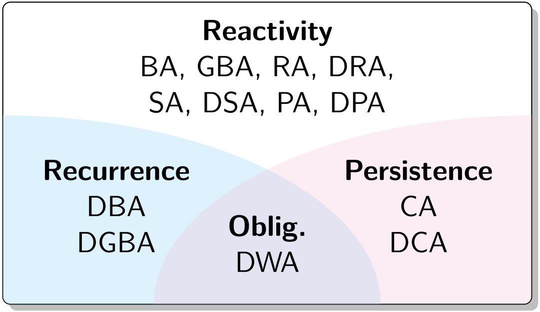


Streett: Fin(0) $\vee$ Inf(1)

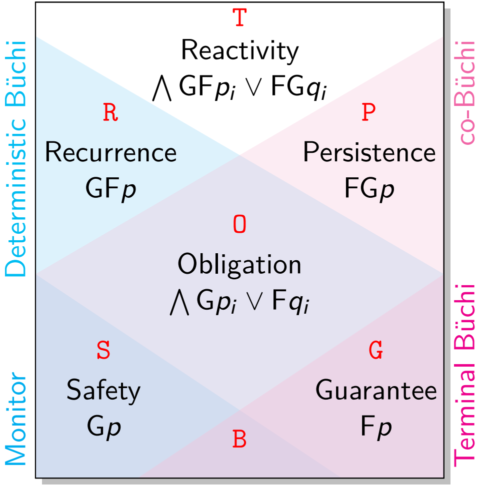


Rabin: Fin(0) $\wedge$ Inf(1)

Expressiveness



Temporal hierarchy of Manna & Pnueli



Manna and Pnueli [MP90]

A hierarchy of temporal properties.

1. Context
2. Deciding Recurrence via Deterministic Rabin Automata
3. Deciding Persistence via co-Büchi Automata
4. Comparison of both procedures

DBA-realizability

Recurrence class

The recurrence class contains all properties that can be recognized by a DBA.

DBA-realizability

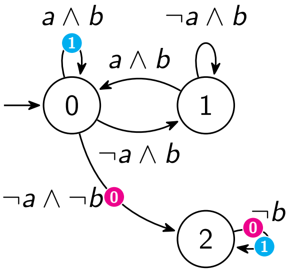
Recurrence class

The recurrence class contains all properties that can be recognized by a DBA.

Theorem ([KPB94])

A DRA is DBA-realizable iff it is DBA-type.

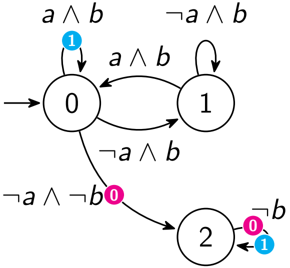
Converting a DRA into an equivalent DBA when it exists



Rabin: Fin(0) $\wedge$ Inf(1)

$T_B = \{ \quad \}$

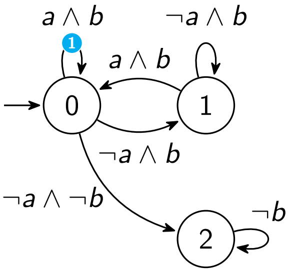
Converting a DRA into an equivalent DBA when it exists



Rabin: Fin(0) $\wedge$ Inf(1)

$$T_B = \{(0,0)\}$$

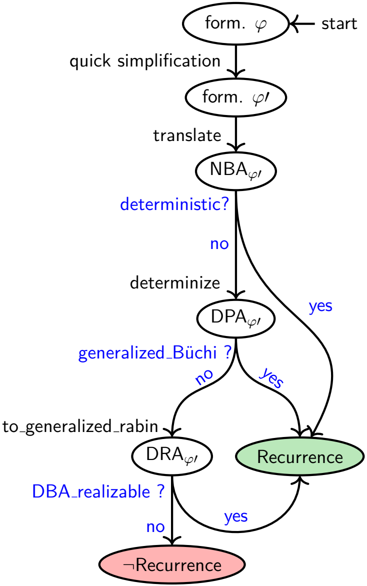
Converting a DRA into an equivalent DBA when it exists



Büchi: Inf(1)

$$T_B = \{(0,0)\}$$

Deciding Recurrence membership of $\varphi = f \vee \neg f$ via DRA



Our contributions

- Quick simplification
- DBA_realizable

Deciding Persistence via co-Büchi Automata

1. Context
2. Deciding Recurrence via Deterministic Rabin Automata
3. Deciding Persistence via co-Büchi Automata
4. Comparison of both procedures

A co-Büching Toolbox [AÏ7]

Persistence class

Persistence properties, are those that can be recognized by a co-Büchi automaton.

A co-Büching Toolbox [AÏ7]

Persistence class

Persistence properties, are those that can be recognized by a co-Büchi automaton.

$\varphi \rightarrow \mathcal{A}_\varphi \rightarrow \mathcal{A}_{NCA} \rightarrow \mathcal{A}_{DCA}$

- Based on [BK09, BK11],

A co-Büching Toolbox [AÏ7]

Persistence class

Persistence properties, are those that can be recognized by a co-Büchi automaton.

$\varphi \to \mathcal{A}_\varphi \to \mathcal{A}_{NCA} \to \mathcal{A}_{DCA}$

- Based on [BK09, BK11],
- Not always possible,

A co-Büching Toolbox [AÏ7]

Persistence class

Persistence properties, are those that can be recognized by a co-Büchi automaton.

$\varphi \to \mathcal{A}_\varphi \to \mathcal{A}_{NCA} \to \mathcal{A}_{DCA}$

- Based on [BK09, BK11],
- Not always possible,
- But, guarantees that $\mathscr{L}(\mathcal{A}_\varphi) \subseteq \mathscr{L}(\mathcal{A}_{NCA})$ and $\mathscr{L}(\mathcal{A}_\varphi) \subseteq \mathscr{L}(\mathcal{A}_{DCA})$

A co-Büching Toolbox [AÏ7]

Persistence class

Persistence properties, are those that can be recognized by a co-Büchi automaton.

$\varphi \rightarrow \mathcal{A}_\varphi \rightarrow \mathcal{A}_{NCA} \rightarrow \mathcal{A}_{DCA}$

- Based on [BK09, BK11],
- Not always possible,
- But, guarantees that $\mathscr{L}(\mathcal{A}_\varphi) \subseteq \mathscr{L}(\mathcal{A}_{NCA})$ and $\mathscr{L}(\mathcal{A}_\varphi) \subseteq \mathscr{L}(\mathcal{A}_{DCA})$
- $\mathcal{A}_{NCA}$ constructed on top of $\mathcal{A}_\varphi \times \text{subset\_cons}(\mathcal{A}_\varphi)$

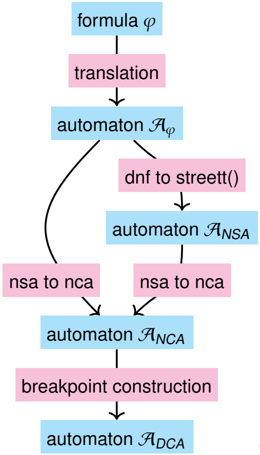
A co-Büching Toolbox [AÏ17]

Persistence class

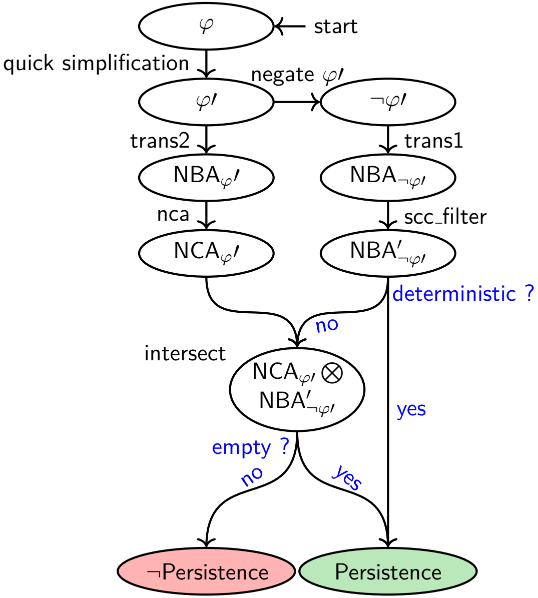
Persistence properties, are those that can be recognized by a co-Büchi automaton.

$\varphi \to \mathcal{A}_\varphi \to \mathcal{A}_{NCA} \to \mathcal{A}_{DCA}$

- Based on [BK09, BK11],
- Not always possible,
- But, guarantees that $\mathscr{L}(\mathcal{A}_\varphi) \subseteq \mathscr{L}(\mathcal{A}_{NCA})$ and $\mathscr{L}(\mathcal{A}_\varphi) \subseteq \mathscr{L}(\mathcal{A}_{DCA})$
- $\mathcal{A}_{NCA}$ constructed on top of $\mathcal{A}_\varphi \times$ subset_cons($\mathcal{A}_\varphi$)



Deciding Persistence membership of $\varphi = f \vee \neg f$ via NCA



Our contributions

- Quick simplification
- Determinism check
- Avoid Aug. subset cons. when possible
- Avoid dead SCCs

Summary: pros and cons

	Method 1 (via CA)	Method 2 (via DRA)
Translations	$\varphi' \rightarrow \text{NBA}_{\varphi'}$ $\neg\varphi' \rightarrow \text{NBA}_{\neg\varphi'}$	$\varphi' \rightarrow \text{NBA}_{\varphi'}$ -
Conversions	$\text{NBA}_{\varphi'} \rightarrow \text{NCA}_{\varphi'}$ -	$\text{DPA}_{\varphi'} \rightarrow \text{DRA}_{\varphi'}$ $\text{DRA}_{\varphi'} \rightarrow \text{DBA}_{\varphi'}$
Determinization	-	$\text{NBA}_{\varphi'} \rightarrow \text{DPA}_{\varphi'}$
Powerset	In $\text{NBA}_{\varphi'} \rightarrow \text{NCA}_{\varphi'}$	-
Acceptance sets	State-based	Transition-based

1 Context

2 Deciding Recurrence via Deterministic Rabin Automata

3 Deciding Persistence via co-Büchi Automata

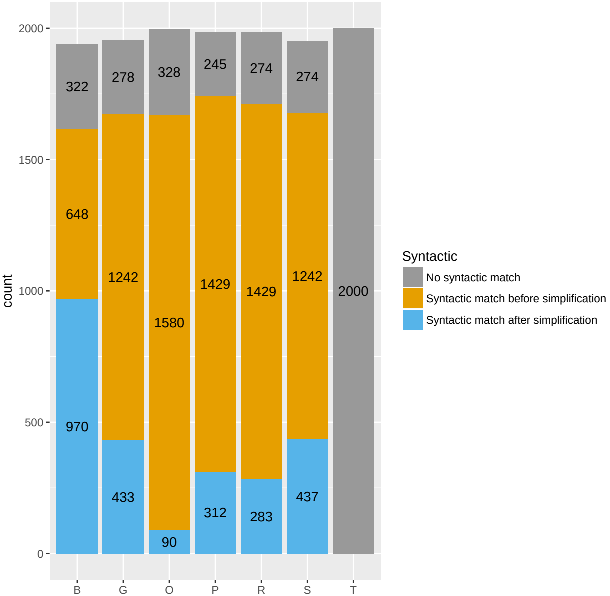
4 Comparison of both procedures

Benchmark's datasets

Two datasets generated using Spot [DLLF+16]

- 13816 random formulas obtained with `randltl` and `ltlfilt`,
- 406 pattern formulas obtained with `genltl` and `ltlfilt`.

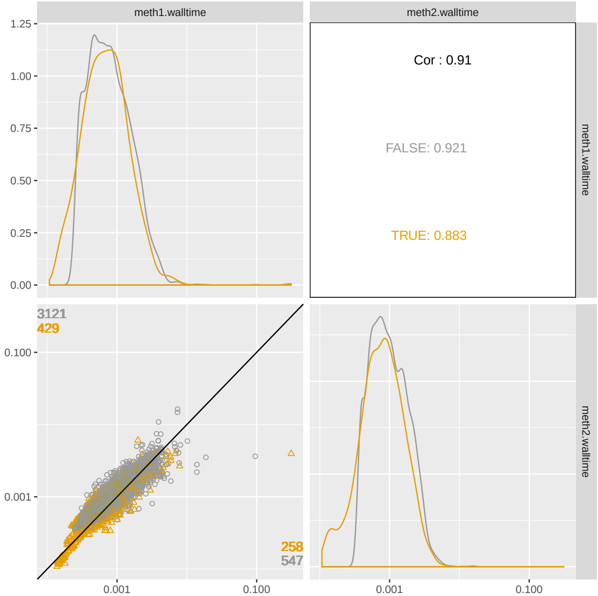
Distribution of random formulas by classes



Random formulas not matched syntactically

- 4746 for persistence,
- 4803 for recurrence.

Deciding persistence on random formulas

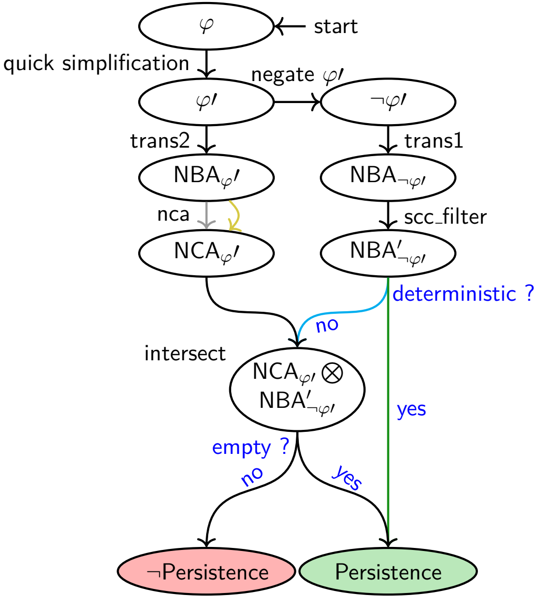


Legend

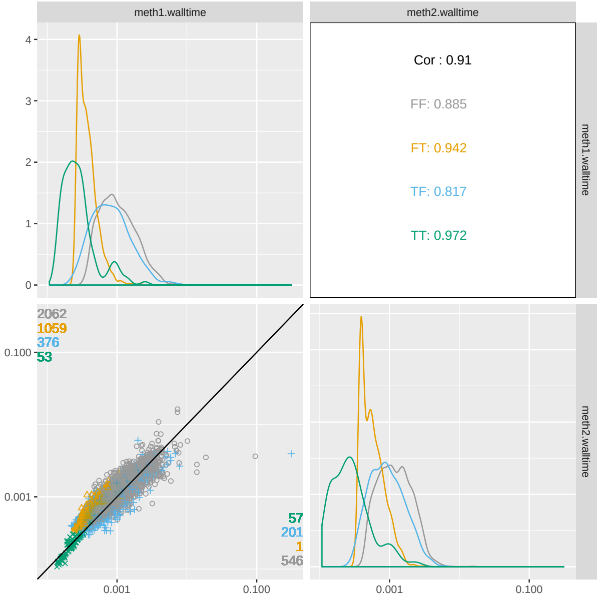
False: not pers.

True: pers.

Processing chains



Highlighting Points



Legend

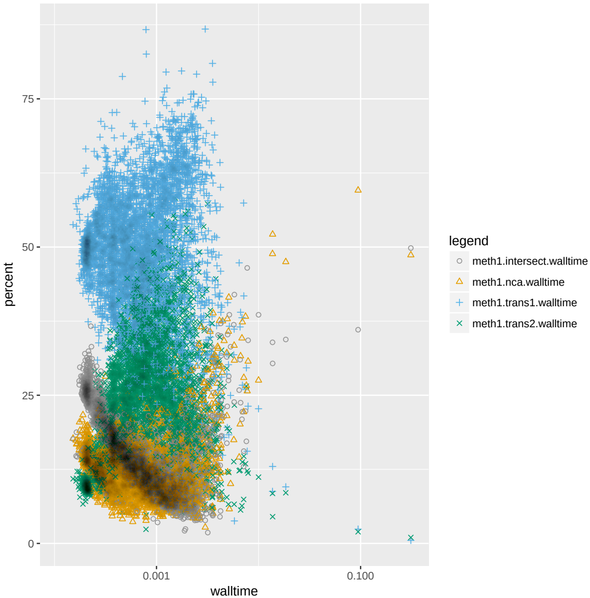
FF: $\varphi\prime$ not pers. & $\mathcal{A}_{\varphi\prime}$ not det.

FT: $\varphi\prime$ not pers. & $\mathcal{A}_{\varphi\prime}$ det.

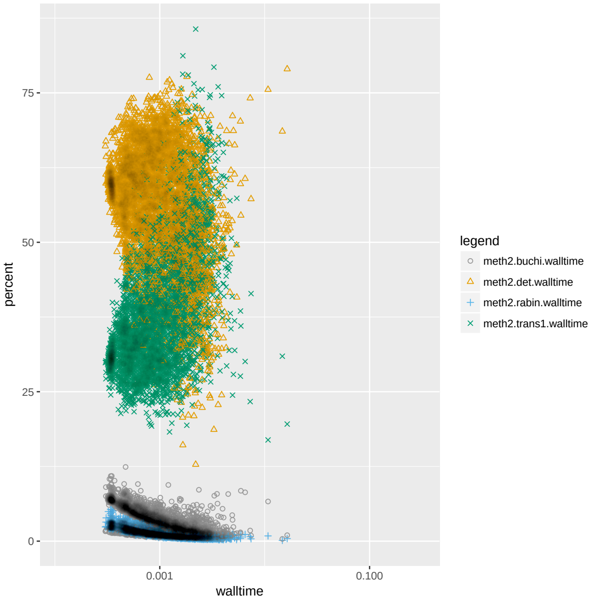
TF: $\varphi\prime$ pers. & $\mathcal{A}_{\neg\varphi\prime}$ not det.

TT: $\varphi\prime$ pers. & $\mathcal{A}_{\neg\varphi\prime}$ det.

Time consumption of method 1



Time consumption of method 2



Conclusion

Our Work

- Quick Simplification (both methods)

Conclusion

Our Work

- Quick Simplification (both methods)
- DBA_realizable (via DRA)

Conclusion

Our Work

- Quick Simplification (both methods)
- DBA_realizable (via DRA)
- Determinism check (via CA)

Conclusion

Our Work

- Quick Simplification (both methods)
- DBA_realizable (via DRA)
- Determinism check (via CA)
- Avoid aug. subset construction when possible (via CA)

Conclusion

Our Work

- Quick Simplification (both methods)
- DBA_realizable (via DRA)
- Determinism check (via CA)
- Avoid aug. subset construction when possible (via CA)
- Avoid dead SCCs (via CA)

Conclusion

Our Work

- Quick Simplification (both methods)
- DBA_realizable (via DRA)
- Determinism check (via CA)
- Avoid aug. subset construction when possible (via CA)
- Avoid dead SCCs (via CA)

Future work

- Extend method via CA to transition-based acceptance

Conclusion

Our Work

- Quick Simplification (both methods)
- DBA_realizable (via DRA)
- Determinism check (via CA)
- Avoid aug. subset construction when possible (via CA)
- Avoid dead SCCs (via CA)

Future work

- Extend method via CA to transition-based acceptance
- Benchmark constructions of DBA with both methods,

Conclusion

Our Work

- Quick Simplification (both methods)
- DBA_realizable (via DRA)
- Determinism check (via CA)
- Avoid aug. subset construction when possible (via CA)
- Avoid dead SCCs (via CA)

Future work

- Extend method via CA to transition-based acceptance
- Benchmark constructions of DBA with both methods,
- Decide Persistence/Recurrence at a syntactical level by rewriting φ.

Bibliography I

[AÏ7] Alexandre Gbaguidi Aïsse. A co-büching toolbox. Technical Report 1705, EPITA Research and Development Laboratory (LRDE), 2017.

[BK09] U. Boker and O. Kupferman. Co-ing Büchi made tight and useful. In *Logic in Computer Science (LICS) 09*, pages 245–254, 2009.

[BK11] U. Boker and O. Kupferman. Co-Büching them all. In *Foundations of Software Science and Computation Structures (FoSSaCS) 11*, Lecture Notes in Computer Science (LNCS). springer, 2011.

[DLLF+16] Alexandre Duret-Lutz, Alexandre Lewkowicz, Amaury Fauchille, Thibaud Michaud, Etienne Renault, and Laurent Xu. Spot 2.0 — a framework for LTL and ω-automata manipulation. In *Proceedings of the 14th International Symposium on Automated Technology for Verification and Analysis (ATVA'16)*, volume 9938 of *Lecture Notes in Computer Science*, pages 122–129. Springer, October 2016.

[KPB94] Sriram C. Krishnan, Anuj Puri, and Robert K. Brayton. *Deterministic ω automata vis-a-vis deterministic Büchi automata*, pages 378–386. Springer Berlin Heidelberg, Berlin, Heidelberg, 1994.

[MP90] Zohar Manna and Amir Pnueli. A hierarchy of temporal properties (invited paper, 1989). In *Proceedings of the Ninth Annual ACM Symposium on Principles of Distributed Computing*, PODC '90, pages 377–410, New York, NY, USA, 1990. ACM.