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

The Safety Fragment of Temporal Logics on Infinite Sequences

Lesson 7

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

April 22nd, 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

RECOGNIZING SAFETY

Algorithms & Complexity

Outline

In this part, we will answer to these questions:

- Can we effectively determine whether a NBA recognizes a safety property? If so, with which complexity?
- Can we effectively determine whether a LTL formula recognizes a safety property? If so, with which complexity?
- How complex is building the *automaton* for the set of *bad prefixes* of a safety ω-regular language?

Theorem (Alpern & Schneider (1987), Sistla (1994))

Given a NBA $\mathcal{A}$, *checking whether* $\mathcal{L}(\mathcal{A})$ *is safety is can be performed effectively.*

References:

- **Bowen Alpern and Fred B. Schneider (1987). "Recognizing Safety and Liveness".** In: *Distributed Comput.* 2.3, pp. 117–126. DOI: 10.1007/BF01782772. URL: https://doi.org/10.1007/BF01782772
- **A Prasad Sistla (1994). "Safety, liveness and fairness in temporal logic".** In: *Formal Aspects of Computing* 6.5, pp. 495–511. DOI: 10.1007/BF01211865

Recognizing Safety

Theorem (Alpern & Schneider (1987), Sistla (1994))

Given a NBA $\mathcal{A}$*, checking whether* $\mathcal{L}(\mathcal{A})$ *is safety is can be performed effectively.*

We prove this theorem.

Definition (Reduced NBA)

A NBA $\mathcal{A} = \langle Q, \Sigma, I, \Delta, F \rangle$ is *reduced* (rNBA, for short) iff from every state in Q there exists a path (of length at least 1) reaching a final state in F.

- Every NBA $\mathcal{A}$ can be turned into rNBA $\mathcal{A}'$ such that $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}')$, by removing the states (and its incoming transitions) from which no final state is reachable.
  - **Important:** this can add *undefined* transitions
- This can be done in *time* linear in $|Q|$ and in *space* nondeterministic logarithmic in $|Q|$ (Savitch's Theorem).

Definition (Closure of a rNBA)

Given a rNBA $\mathcal{A} = \langle Q, \Sigma, I, \Delta, F \rangle$, we define the *closure of* $\mathcal{A}$, denoted with $\mathtt{cl}(\mathcal{A})$, as the automaton $\mathtt{cl}(\mathcal{A}) = \langle Q, \Sigma, I, \Delta, Q \rangle$.

- We will use the automaton $\mathtt{cl}(\mathcal{A})$ to determine whether $\mathcal{L}(\mathcal{A})$ is a safety property.
- **Important:** the automaton $\mathtt{cl}(\mathcal{A})$ rejects a word in Σ^ω only by attempting an *undefined transition*.

Theorem

For any rNBA $\mathcal{A}$, *it holds that* $\mathcal{L}(\mathcal{A})$ *is a safety property iff* $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathtt{cl}(\mathcal{A}))$.

Proof.

$(\Rightarrow)$

- Suppose that $\mathcal{L}(\mathcal{A})$ is a safety property. We show that $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathtt{cl}(\mathcal{A}))$.
- $\mathcal{L}(\mathcal{A}) \subseteq \mathcal{L}(\mathtt{cl}(\mathcal{A}))$: trivial, because $\mathtt{cl}(\mathcal{A})$ is obtained from $\mathcal{A}$ by making all states as accepting.

Theorem

For any rNBA $\mathcal{A}$, *it holds that* $\mathcal{L}(\mathcal{A})$ *is a safety property iff* $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathtt{cl}(\mathcal{A}))$.

Proof.

($\Rightarrow$)

- We show that $\mathcal{L}(\mathtt{cl}(\mathcal{A})) \subseteq \mathcal{L}(\mathcal{A})$.
- Let $\sigma \in \mathcal{L}(\mathtt{cl}(\mathcal{A}))$.
- Observe that, since $\mathcal{A}$ and $\mathtt{cl}(\mathcal{A})$ have the same set of states and the same transition relation, if $\mathtt{cl}(\mathcal{A})$ reads σ (*i.e.*, without incurring in any undefined transition) then also $\mathcal{A}$ reads σ, and *vice versa*.
- Thus, now we focus on the automaton $\mathcal{A}$ reading σ.

Theorem

For any rNBA $\mathcal{A}$*, it holds that* $\mathcal{L}(\mathcal{A})$ *is a safety property iff* $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathtt{cl}(\mathcal{A}))$*.*

Proof.

($\Rightarrow$)

- Choose *any* prefix $\sigma_{[0,i]}$ and let q_i be any of the states reached by $\mathcal{A}$ after reading $\sigma_{[0,i]}$.
- Since $\mathcal{A}$ is *reduced*, there exists a final state q_{f_1} reachable from q_i when $\mathcal{A}$ reads some $\beta_0 \in \Sigma^+$.
- Similarly, since $\mathcal{A}$ is *reduced*, there exists a final state q_{f_2} reachable from q_{f_1} when $\mathcal{A}$ reads some $\beta_1 \in \Sigma^+$.
- ... and so on and so forth ...

Theorem

For any rNBA $\mathcal{A}$, *it holds that* $\mathcal{L}(\mathcal{A})$ *is a safety property iff* $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathtt{cl}(\mathcal{A}))$.

Proof.

$(\Rightarrow)$

- Let $\beta = \beta_0 \cdot \beta_1 \cdot \ldots$. Since, by construction, $\sigma_{[0,i]} \cdot \beta$ induces $\mathcal{A}$ to visit final state *infinitely often*, the word $\sigma_{[0,i]} \cdot \beta$ belongs to $\mathcal{L}(\mathcal{A})$.
- We have proved that, for any $\sigma \in \mathcal{L}(\mathtt{cl}(\mathcal{A}))$, it holds that:

$$\forall i \geq 0 \,.\, \exists \sigma' \in \Sigma^\omega \,.\, \sigma_{[0,i]} \cdot \sigma' \in \mathcal{L}(\mathcal{A})$$

Theorem

For any rNBA $\mathcal{A}$, *it holds that* $\mathcal{L}(\mathcal{A})$ *is a safety property iff* $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathtt{cl}(\mathcal{A}))$.

Proof.

($\Rightarrow$)

- Since by hypothesis $\mathcal{L}(\mathcal{A})$ is a safety property, for all $\sigma \in \Sigma^\omega$, we have that,

$$\sigma \notin \mathcal{L}(\mathcal{A}) \leftrightarrow \exists i \geq 0 \;.\; \forall \sigma' \in \Sigma^\omega \;.\; \sigma_{[0,i]} \cdot \sigma' \notin \mathcal{L}(\mathcal{A})$$

- Since before we proved that the rightmost part of the above equation is false for any $\sigma \in \mathcal{L}(\mathtt{cl}(\mathcal{A}))$, we have that $\sigma \in \mathcal{L}(\mathcal{A})$.

Theorem

For any rNBA $\mathcal{A}$, *it holds that* $\mathcal{L}(\mathcal{A})$ *is a safety property iff* $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathtt{cl}(\mathcal{A}))$.

Proof.

($\Leftarrow$)

- Suppose that $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathtt{cl}(\mathcal{A}))$.
- We prove that, for all $\sigma \in \Sigma^\omega$, it holds:

$$\sigma \notin \mathcal{L}(\mathcal{A}) \leftrightarrow \exists i \geq 0 \,.\, \forall \sigma' \in \Sigma^\omega \,.\, \sigma_{[0,i]} \cdot \sigma' \notin \mathcal{L}(\mathcal{A})$$

Theorem

For any rNBA $\mathcal{A}$, *it holds that* $\mathcal{L}(\mathcal{A})$ *is a safety property iff* $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathtt{cl}(\mathcal{A}))$.

Proof.

$(\Leftarrow)$

- The right-to-left direction

$$\forall \sigma \in \Sigma^\omega \ . \qquad \big(\sigma \notin \mathcal{L}(\mathcal{A}) \leftarrow \exists i \geq 0 \ . \ \forall \sigma' \in \Sigma^\omega \ . \ \sigma_{[0,i]} \cdot \sigma' \notin \mathcal{L}(\mathcal{A})\big)$$

holds for every language: it suffices to take $\sigma' := \sigma_{[i+1,\infty)}$.

Theorem

For any rNBA $\mathcal{A}$*, it holds that* $\mathcal{L}(\mathcal{A})$ *is a safety property iff* $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathtt{cl}(\mathcal{A}))$.

Proof.

$(\Leftarrow)$

- We prove the left-to-right direction:

$$\forall \sigma \in \Sigma^\omega \ . \qquad (\sigma \notin \mathcal{L}(\mathcal{A}) \rightarrow \exists i \geq 0 \ . \ \forall \sigma' \in \Sigma^\omega \ . \ \sigma_{[0,i]} \cdot \sigma' \notin \mathcal{L}(\mathcal{A}))$$

- Since by hypothesis $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathtt{cl}(\mathcal{A}))$, it is equivalent to prove:

$$\forall \sigma \in \Sigma^\omega \ . \qquad (\sigma \notin \mathcal{L}(\mathtt{cl}(\mathcal{A})) \rightarrow \exists i \geq 0 \ . \ \forall \sigma' \in \Sigma^\omega \ . \ \sigma_{[0,i]} \cdot \sigma' \notin \mathcal{L}(\mathtt{cl}(\mathcal{A})))$$

Theorem

For any rNBA $\mathcal{A}$, *it holds that* $\mathcal{L}(\mathcal{A})$ *is a safety property iff* $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathtt{cl}(\mathcal{A}))$.

Proof.

$(\Leftarrow)$

- $\forall \sigma \in \Sigma^\omega \,.\, \big(\sigma \notin \mathcal{L}(\mathtt{cl}(\mathcal{A})) \to \exists i \geq 0 \,.\, \forall \sigma' \in \Sigma^\omega \,.\, \sigma_{[0,i]} \cdot \sigma' \notin \mathcal{L}(\mathtt{cl}(\mathcal{A}))\big)$
- Suppose $\sigma \notin \mathcal{L}(\mathtt{cl}(\mathcal{A}))$. Thus the automaton $\mathtt{cl}(\mathcal{A})$ *rejects* σ.
- Since by hypothesis $\mathtt{cl}(\mathcal{A})$ is a *reduced* Büchi automatom, $\mathtt{cl}(\mathcal{A})$ can reject σ only by attempting an *undefined* transition.

Theorem

For any rNBA $\mathcal{A}$, *it holds that* $\mathcal{L}(\mathcal{A})$ *is a safety property iff* $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathtt{cl}(\mathcal{A}))$.

Proof.

$(\Leftarrow)$

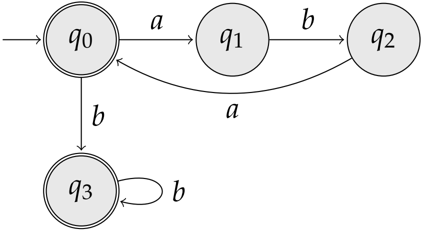
- $\forall \sigma \in \Sigma^\omega \;. \quad (\sigma \notin \mathcal{L}(\mathtt{cl}(\mathcal{A})) \rightarrow \exists i \geq 0 \;.\; \forall \sigma' \in \Sigma^\omega \;.\; \sigma_{[0,i]} \cdot \sigma' \notin \mathcal{L}(\mathtt{cl}(\mathcal{A})))$
- Let i be the position of σ at which $\mathtt{cl}(\mathcal{A})$ takes the undefined transition.
- Clearly, it holds that:

$$\forall \sigma' \in \Sigma^\omega \;.\; \sigma_{[0,i]} \cdot \sigma' \notin \mathcal{L}(\mathtt{cl}(\mathcal{A}))$$

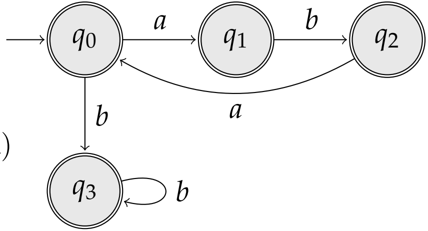
- Thus $\mathtt{cl}(\mathcal{A})$ (and $\mathcal{A}$ as well) specify a safety property. $\square$

$\Sigma = \{a, b\}$, $\mathcal{L} = (a \cdot b \cdot a)^\omega \cup (a \cdot b \cdot a)^* \cdot b^\omega$

$\mathcal{A}$



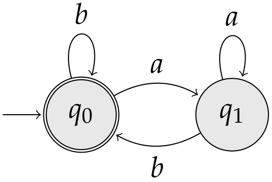
$\mathtt{cl}(\mathcal{A})$



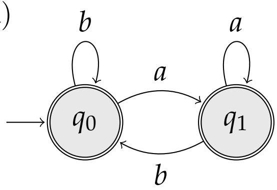
The language $\mathcal{L}$ is *safety* because $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathtt{cl}(\mathcal{A}))$.

$\Sigma = \{a, b\}$, $\mathcal{L} = \{\sigma \in \Sigma^\omega \mid$ each 'a' is eventually followed by 'b'$\}$

$\mathcal{A}$



$\mathsf{cl}(\mathcal{A})$



The language $\mathcal{L}$ is <u>*not*</u> safety because $\mathcal{L}(\mathcal{A}) \neq \mathcal{L}(\mathsf{cl}(\mathcal{A}))$.

- $a^\omega \in \mathcal{L}(\mathsf{cl}(\mathcal{A}))$ but $a^\omega \notin \mathcal{L}(\mathcal{A})$

Complexity of the procedure

Checking whether $\mathcal{L}(\mathtt{cl}(\mathcal{A})) = \mathcal{L}(\mathcal{A})$ is done by checking whether:

$$\mathcal{L}(\mathtt{cl}(\mathcal{A})) \subseteq \mathcal{L}(\mathcal{A}) \ \wedge \ \mathcal{L}(\mathcal{A}) \subseteq \mathcal{L}(\mathtt{cl}(\mathcal{A}))$$

which in turn is equivalent to check whether:

$$\mathcal{L}(\mathtt{cl}(\mathcal{A})) \cap \overline{\mathcal{L}(\mathcal{A})} = \varnothing \ \wedge \ \mathcal{L}(\mathcal{A}) \cap \overline{\mathcal{L}(\mathtt{cl}(\mathcal{A}))} = \varnothing$$

- Complementation of NBA is needed.
- Complexity of Büchi complementation (n = number of states):
  - upper bound: $\mathcal{O}(0.96n)^n$
  - lower bound: $\Omega(0.76n)^n$
- **Sven Schewe (2009). "Büchi Complementation Made Tight".** In: *26th International Symposium on Theoretical Aspects of Computer Science, STACS 2009, February 26-28, 2009, Freiburg, Germany, Proceedings*. Ed. by Susanne Albers and Jean-Yves Marion. Vol. 3. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Germany, pp. 661–672. DOI: 10.4230/LIPIcs.STACS.2009.1854. URL: https://doi.org/10.4230/LIPIcs.STACS.2009.1854

Complexity of the procedure

Checking whether $\mathcal{L}(\mathtt{cl}(\mathcal{A})) = \mathcal{L}(\mathcal{A})$ is done by checking whether:

$$\mathcal{L}(\mathtt{cl}(\mathcal{A})) \subseteq \mathcal{L}(\mathcal{A}) \ \wedge \ \mathcal{L}(\mathcal{A}) \subseteq \mathcal{L}(\mathtt{cl}(\mathcal{A}))$$

which in turn is equivalent to check whether:

$$\mathcal{L}(\mathtt{cl}(\mathcal{A})) \cap \overline{\mathcal{L}(\mathcal{A})} = \varnothing \ \wedge \ \mathcal{L}(\mathcal{A}) \cap \overline{\mathcal{L}(\mathtt{cl}(\mathcal{A}))} = \varnothing$$

- The emptiness check can be performed *on-the-fly* during the construction of the automata.
- Total Complexity: polynomial space (PSPACE)

Complexity of the problem

Theorem

The set of NBA *recognizing safety properties is* PSPACE.

Open Question:

Is PSPACE-complete?

Complexity for the deterministic case

Theorem

Given a DBA $\mathcal{A}$ *with* n *states, checking whether* $\mathcal{L}(\mathcal{A})$ *is safety can be done in time polynomial in* n*.*

Proof.

We have to check these two conditions:

- $\mathcal{L}(\mathtt{cl}(\mathcal{A})) \cap \overline{\mathcal{L}(\mathcal{A})} = \varnothing$
- $\mathcal{L}(\mathcal{A}) \cap \overline{\mathcal{L}(\mathtt{cl}(\mathcal{A}))} = \varnothing$

Both require *complementation*, which is problematic (exponential) even for *deterministic* Büchi automata:

- swapping final states with nonfinal ones does *not* work

Complexity for the deterministic case

Theorem

Given a DBA $\mathcal{A}$ with n states, checking whether $\mathcal{L}(\mathcal{A})$ is safety can be done in time polynomial in n.

Proof.

We resort to *deterministic Rabin automata* (DRA).
Rabin acceptance condition:

- Final state condition: $\Omega = \{(A_i, B_i)\}_{i=1}^{n}$
- A run π is accepting iff, for some $i \in \{1, \ldots, n\}$, it holds that $Inf(\pi) \cap A_i = \varnothing$ and $Inf(\pi) \cap B_i \neq \varnothing$.

Any DBA $\mathcal{A} := \langle Q, \Sigma, q_0, \delta, F\rangle$ is equivalent to the DRA $\mathcal{A}' := \langle Q, \Sigma, q_0, \delta, \{(\varnothing, F)\}\rangle$.

Complexity for the deterministic case

Theorem

Given a DBA $\mathcal{A}$ *with* n *states, checking whether* $\mathcal{L}(\mathcal{A})$ *is safety can be done in time polynomial in* n.

Proof.

Any DRA $\mathcal{A}' := \langle Q, \Sigma, q_0, \delta, \Omega \rangle$ with $\Omega = \{(A_i, B_i)\}_{i=1}^{n}$ can be complemented *without* an exponential blow-up into a *deterministic Streett automaton*.

Streett acceptance condition:

- Final state condition: $\Omega = \{(A_i, B_i)\}_{i=1}^{n}$
- A run π is accepting iff, for all $i \in \{1, \ldots, n\}$, either $\mathit{Inf}(\pi) \cap A_i = \varnothing$ or $\mathit{Inf}(\pi) \cap B_i \neq \varnothing$.

Complexity for the deterministic case

Theorem

Given a DBA $\mathcal{A}$ *with* n *states, checking whether* $\mathcal{L}(\mathcal{A})$ *is safety can be done in time polynomial in* n*.*

Proof.

Any DRA $\mathcal{A}' := \langle Q, \Sigma, q_0, \delta, \Omega \rangle$ with $\Omega = \{(A_i, B_i)\}_{i=1}^{n}$ can be complemented *without* an exponential blow-up into a *deterministic Streett automaton*.

- We define the deterministic Streett automaton $\overline{\mathcal{A}'}$ as $\langle Q, \Sigma, q_0, \delta, \Omega' \rangle$ where

$$\Omega' := \{(B, A) \mid (A, B) \in \Omega\}$$

- $\overline{\mathcal{A}'}$ recognizes the complement language of $\mathcal{A}'$.

Complexity for the deterministic case

Theorem

Given a DBA $\mathcal{A}$ *with* n *states, checking whether* $\mathcal{L}(\mathcal{A})$ *is safety can be done in time polynomial in* n.

Proof.

Consider the two conditions:

- $\mathcal{L}(\mathsf{cl}(\mathcal{A})) \cap \overline{\mathcal{L}(\mathcal{A})} = \varnothing$
- $\mathcal{L}(\mathcal{A}) \cap \overline{\mathcal{L}(\mathsf{cl}(\mathcal{A}))} = \varnothing$

We perform the construction from Büchi to Streett (for the complement language) for the automata $\mathcal{L}(\mathcal{A})$ and $\mathcal{L}(\mathsf{cl}(\mathcal{A}))$.

Complexity for the deterministic case

Theorem

Given a DBA $\mathcal{A}$ with n states, checking whether $\mathcal{L}(\mathcal{A})$ is safety can be done in time polynomial in n.

Proof.

Consider the two conditions:

- $\mathcal{L}(\mathsf{cl}(\mathcal{A})) \cap \overline{\mathcal{L}(\mathcal{A})} = \varnothing$
- $\mathcal{L}(\mathcal{A}) \cap \overline{\mathcal{L}(\mathsf{cl}(\mathcal{A}))} = \varnothing$

Every Büchi automaton $\mathcal{A} = \langle Q, \Sigma, I, \Delta, F \rangle$ is also a Streett automaton.

$$\mathcal{A}' := \langle Q, \Sigma, I, \Delta, \{(Q, F)\} \rangle$$

Note the role of Q in (Q, F) to always violate the first disjunct of Streett condition.

Complexity for the deterministic case

Theorem

Given a DBA $\mathcal{A}$ *with* n *states, checking whether* $\mathcal{L}(\mathcal{A})$ *is safety can be done in time polynomial in* n.

Proof.

Consider the two conditions:

- $\mathcal{L}(\mathsf{cl}(\mathcal{A})) \cap \overline{\mathcal{L}(\mathcal{A})} = \varnothing$
- $\mathcal{L}(\mathcal{A}) \cap \overline{\mathcal{L}(\mathsf{cl}(\mathcal{A}))} = \varnothing$

Streett automata are closed under Boolean operations.

Complexity for the deterministic case

Theorem

Given a DBA $\mathcal{A}$ *with* n *states, checking whether* $\mathcal{L}(\mathcal{A})$ *is safety can be done in time polynomial in* n*.*

Proof.

Consider the two conditions:

- $\mathcal{L}(\mathsf{cl}(\mathcal{A})) \cap \overline{\mathcal{L}(\mathcal{A})} = \varnothing$
- $\mathcal{L}(\mathcal{A}) \cap \overline{\mathcal{L}(\mathsf{cl}(\mathcal{A}))} = \varnothing$

Since their emptiness can be solved in nondeterministic logarithmic space, this proves that the set of safety DBA is PTIME. $\square$

Theorem (Sistla (1994))

The set of LTL *formulas* ϕ *such that* $\mathcal{L}(\phi)$ *is safety is* PSPACE*-complete.*

Or equivalently: Given a LTL formula ϕ, the problem of establishing whether $\mathcal{L}(\phi)$ is safety is PSPACE-complete.

Theorem

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

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

Reference

Moshe Y Vardi and Pierre Wolper (1986). "An automata-theoretic approach to automatic program verification". In: *Proceedings of the First Symposium on Logic in Computer Science*. IEEE Computer Society, pp. 322–331

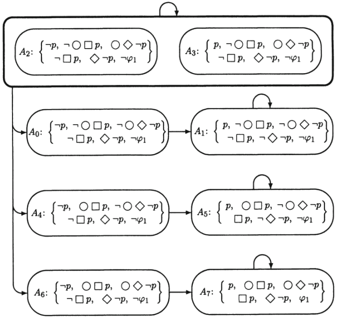
Reference

Moshe Y Vardi (1996). "An automata-theoretic approach to linear temporal logic". In: *Logics for concurrency*. Springer, pp. 238–266

Theorem

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

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



Picture taken from

Zohar Manna and Amir Pnueli (1995). ***Temporal verification of reactive systems - safety.*** **Springer. ISBN: 978-0-387-94459-3**

Theorem (Sistla (1994))

The set of LTL *formulas* ϕ *such that* $\mathcal{L}(\phi)$ *is safety is* PSPACE*-complete.*

Or equivalently: Given a LTL formula ϕ, the problem of establishing whether $\mathcal{L}(\phi)$ is safety is PSPACE-complete.

Theorem (Sistla (1994))

The set of LTL *formulas* ϕ *such that* $\mathcal{L}(\phi)$ *is safety is* PSPACE*-complete.*

Proof.

- Let $\phi \in$ LTL.
- We can effectively build a NBA $\mathcal{A}_\phi$ such that $\mathcal{L}(\mathcal{A}_\phi) = \mathcal{L}(\phi)$ and $|\mathcal{A}_\phi| = 2^{\mathcal{O}(n)}$.
- In *space polynomial in* n, we can turn $\mathcal{A}_\phi$ into an equivalent rNBA $\mathcal{A}'_\phi$.
- Let $\mathtt{cl}(\mathcal{A}'_\phi)$ be its *closure*.
- $\mathcal{L}(\phi)$ is safety iff:
  - $\mathcal{L}(\mathcal{A}'_\phi) \subseteq \mathcal{L}(\mathtt{cl}(\mathcal{A}'_\phi))$ and
  - $\mathcal{L}(\mathtt{cl}(\mathcal{A}'_\phi)) \subseteq \mathcal{L}(\mathcal{A}'_\phi)$

Since the 1st point is always true, it suffices to prove the second.

Theorem (Sistla (1994))

The set of LTL *formulas* ϕ *such that* $\mathcal{L}(\phi)$ *is safety is* PSPACE*-complete.*

Proof.

- $\mathcal{L}(\mathrm{cl}(\mathcal{A}'_\phi)) \subseteq \mathcal{L}(\mathcal{A}'_\phi)$ is equivalent to $\mathcal{L}(\mathrm{cl}(\mathcal{A}'_\phi)) \cap \overline{\mathcal{L}(\mathcal{A}'_\phi)}$
- ... but instead of complementing $\mathcal{A}'_\phi$ (which is difficult) we complement the formula ϕ (which has a trivial, constant complexity)
- We can effectively build a NBA $\mathcal{A}_{\neg\phi}$ such that $\mathcal{L}(\mathcal{A}_{\neg\phi}) = \mathcal{L}(\neg\phi)$ and $|\,\mathcal{A}_{\neg\phi}\,| = 2^{\mathcal{O}(n)}$.
- We have that $\mathcal{L}(\mathcal{A}_{\neg\phi}) = \overline{\mathcal{L}(\mathcal{A}'_\phi)}$.

Theorem (Sistla (1994))

The set of LTL *formulas* ϕ *such that* $\mathcal{L}(\phi)$ *is safety is* PSPACE*-complete.*

Proof.

- $\mathcal{L}(\phi)$ is safety iff $\mathcal{L}(\mathtt{cl}(\mathcal{A}'_\phi)) \cap \mathcal{L}(\mathcal{A}_{\neg\phi}) = \varnothing$.
- Check *emptiness* of $\mathtt{cl}(\mathcal{A}'_\phi) \times \mathcal{A}_{\neg\phi}$:
  - $\mathtt{cl}(\mathcal{A}'_\phi) \times \mathcal{A}_{\neg\phi}$ is of size $2^{\mathcal{O}(n)}$
  - Emptiness: nondeterministic *logarithmic* space in the number of states of the automaton.
  - It can be performed *on-the-fly* during the construction of $\mathtt{cl}(\mathcal{A}'_\phi) \times \mathcal{A}_{\neg\phi}$.
  - Total Complexity: Polynomial Space (PSPACE)

Theorem (Sistla (1994))

The set of LTL *formulas* ϕ *such that* $\mathcal{L}(\phi)$ *is safety is* PSPACE*-complete.*

Proof.

- We prove that the problem is PSPACE-hard.
- Reduction from the LTL validity problem, which is PSPACE-complete.
- Let $\phi \in$ LTL over the atomic propositions $\mathcal{AP}$ and let $p \notin \mathcal{AP}$ a *fresh* proposition.
- It holds that: ϕ is *valid* iff $\mathcal{L}(\phi \vee \mathsf{F}p)$ is *safety*.

Theorem (Sistla (1994))

The set of LTL *formulas* ϕ *such that* $\mathcal{L}(\phi)$ *is safety is* PSPACE-*complete.*

Proof.

- We prove: if ϕ is *valid* then $\mathcal{L}(\phi \vee \mathsf{F}p)$ is *safety*.
- Suppose that ϕ is valid.
- Then $\phi \vee \mathsf{F}p$ is valid as well, that is $\mathcal{L}(\phi \vee \mathsf{F}p) = (2^{\mathcal{AP}})^\omega$.
- Clearly, $(2^{\mathcal{AP}})^\omega$ is a *safety* language, because every violation (there are none) is irremediable.

Theorem (Sistla (1994))

The set of LTL *formulas* ϕ *such that* $\mathcal{L}(\phi)$ *is safety is* PSPACE*-complete.*

Proof.

- We prove: if $\mathcal{L}(\phi \vee \mathsf{F}p)$ is *safety* then ϕ is *valid*.
- Suppose there exists a violation of $\mathcal{L}(\phi \vee \mathsf{F}p)$, that is a trace $\sigma \in (2^{\mathcal{AP} \cup \{p\}})^\omega$ such that $\sigma \models \neg\phi \wedge \mathsf{G}\neg p$.
- Since by hypothesis $\mathcal{L}(\phi \vee \mathsf{F}p)$ is *safety*, this violation must be *irremediable*, that is $\exists i \geq 0 \,.\, \forall \sigma' \,.\, \sigma_{[0,i]} \cdot \sigma' \models \neg\phi \wedge \mathsf{G}\neg p$.
- Because $\sigma_{[0,i]} \cdot \sigma'$ has also to satisfy $\mathsf{G}\neg p$ for all σ', there exists no such i.
- This means that there are no violations of $\phi \vee \mathsf{F}p$ (this formula is valid).
- Since p doesn't occur in ϕ, this means that ϕ is valid.

DETECTING BAD PREFIXES

Algorithms & Complexity

For problems like *model checking* and *reactive synthesis*, given a safety property:

- one doesn't want to build a NBA
- but rather to reason on finite words and to build a DFA.

In particular, we consider the automaton over finite words for the set of bad prefixes.

Reasoning over finite words is simpler than reasoning over infinite words.

Task:

Given a NBA $\mathcal{A}$, to give an algorithm for building the automaton recognizing exactly the set of bad prefixes of $\mathcal{L}(\mathcal{A})$ and to analyze its complexity.

For problems like *model checking* and *reactive synthesis*, given a safety property:

- one doesn't want to build a NBA
- but rather to reason on finite words and to build a DFA.

In particular, we consider the automaton over finite words for the set of bad prefixes.

Reasoning over finite words is simpler than reasoning over infinite words.

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

Definition (Safety Property)

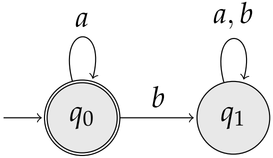
$\mathcal{L} \subseteq \Sigma^\omega$ is a *safety property* iff, for all $\sigma \notin \mathcal{L}$, there exists an position $i \in \mathbb{N}$ such that $\sigma_{[0,i]} \cdot \sigma' \notin \mathcal{L}$, for all $\sigma' \in \Sigma^\omega$.

- $\sigma_{[0,i]}$ is called the *bad prefix* of σ.
- We denote with $\mathtt{bad}(\mathcal{L})$ the set of bad prefixes of $\mathcal{L}$.
- $\mathtt{bad}(\mathcal{L})$ is a language of finite words, that is $\mathtt{bad}(\mathcal{L}) \subseteq \Sigma^*$.

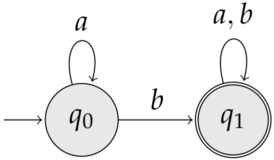
The Deterministic Case

If $\mathcal{A}$ is a DBA (Deterministic Büchi Automaton), then building the automaton for $\mathtt{bad}(\mathcal{L}(\mathcal{A}))$ is straightforward

- nondeterministic polynomial space and linear time.



$\mathcal{A}$

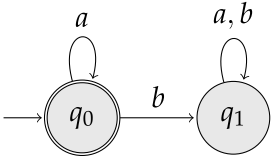


$\mathcal{A}_{bad}$

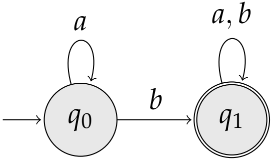
The Deterministic Case

If $\mathcal{A}$ is a DBA (Deterministic Büchi Automaton), then building the automaton for $\mathtt{bad}(\mathcal{L}(\mathcal{A}))$ is straightforward

- Given a set of states S of $\mathcal{A}$, we denote with $\mathcal{A}^S$ the automaton obtained from $\mathcal{A}$ by defining the set of initial states to be S.
- Let $\mathcal{A}_{bad}$ be the DFA obtained from $\mathcal{A}$ by defining a state q to be *final* iff $\mathcal{A}^{\{q\}}$ recognizes the empty set.
- It holds that $\mathcal{L}(\mathcal{A}_{bad}) = \mathtt{bad}(\mathcal{L}(\mathcal{A}))$.



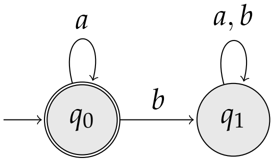
$\mathcal{A}$



$\mathcal{A}_{bad}$

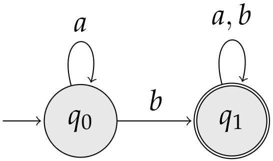
The Deterministic Case

- $\mathcal{L}(\mathcal{A}) = a^\omega$



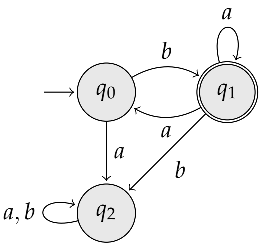
$\mathcal{A}$

- $\texttt{bad}(\mathcal{L}(\mathcal{A})) = a^* \cdot b \cdot \Sigma^*$

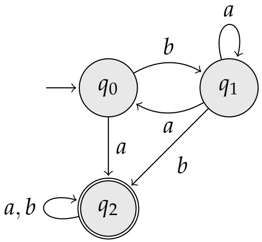


$\mathcal{A}_{bad}$

- The nondeterministic case is more involved.
- The previous algorithm for the deterministic case does not work in the nondeterministic case.
- Counterexample:
  - $\mathcal{L}(\mathcal{A}) = b \cdot a^\omega \ \cup \ (b \cdot a^+)^\omega \ \cup \ (b \cdot a^+)^* \cdot a^\omega$
  - The automaton $\mathcal{A}_{bad}$ recognizes also the word "*bab*" which is not a bad prefix.
- We need another way to build $\mathcal{A}_{bad}$.



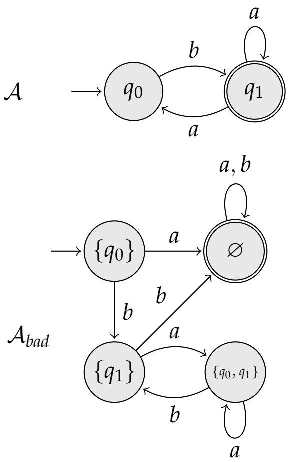
$\mathcal{A}$



$\mathcal{A}_{bad}$

- Let $\mathcal{A} = \langle Q, \Sigma, I, \Delta, F \rangle$ be NBA.
- We define $\mathcal{A}_{bad}$ as the DFA $\langle 2^Q, \Sigma, q_0', \delta', F' \rangle$ such that:
  - $q_0' := I$
  - for every $S \in 2^Q$ and every $\sigma \in \Sigma$, $\delta(S, \sigma) := \bigcup_{q \in S} \delta(q, \sigma)$.
  - $F := \{S \in 2^Q \mid \mathcal{L}(\mathcal{A}^S) = \varnothing\}$.
- Complexity: $|\mathcal{A}_{bad}| \in 2^{\mathcal{O}(n)}$ where $n = |Q|$.

> *"The detection of bad prefixes with a nondeterministic Büchi automaton has the flavor of determinization."*



- Let $\mathcal{A} = \langle Q, \Sigma, I, \Delta, F \rangle$ be NBA.
- We define $\mathcal{A}_{bad}$ as the DFA $\langle 2^Q, \Sigma, q_0', \delta', F' \rangle$ such that:
  - $q_0' := I$
  - for every $S \in 2^Q$ and every $\sigma \in \Sigma$, $\delta(S, \sigma) := \bigcup_{q \in S} \delta(q, \sigma)$.
  - $F := \{S \in 2^Q \mid \mathcal{L}(\mathcal{A}^S) = \varnothing\}$.
- Complexity: $|\mathcal{A}_{bad}| \in 2^{\mathcal{O}(n)}$ where $n = |Q|$.

> *"The detection of bad prefixes with a nondeterministic Büchi automaton has the flavor of determinization."*

This is a *lowerbound*.

- There exists an NFA $\mathcal{A}$ with n states such that
  - all states are accepting
  - its complement $\overline{\mathcal{A}}$ has $2^{\Theta(n)}$ states.
- Let $\mathcal{A}'$ be the NBA obtained by considering $\mathcal{A}$ as a Büchi automaton.
- Since both $\mathcal{A}$ and $\mathcal{A}'$ can reject a word only by attempting an undefined transition, it holds that $\mathtt{bad}(\mathcal{A}') = \overline{\mathcal{A}}$.
- It follows that the automaton for $\mathtt{bad}(\mathcal{A})$ has $2^{\Theta(n)}$ states.

An analogous result holds for the cosafety case.

Theorem

Given a NBA $\mathcal{A}$ *with* n *states such that* $\mathcal{L}(\mathcal{A})$ *is cosafety, the size of an automaton for* $\mathtt{good}(\mathcal{A})$ *is* $2^{\Theta(n)}$.

Detecting bad prefixing of an LTL formula recognizing a safety language is doubly exponential.

Theorem

Given an LTL *formula* ϕ *such that* $\mathcal{L}(\phi)$ *is safety and* $|\phi| = n$*, the size of an automaton for* $\mathtt{bad}(\mathcal{L}(\phi))$ *is* $2^{2^{\mathcal{O}(n)}}$ *and* $2^{2^{\Omega(\sqrt{n})}}$.

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

REFERENCES

Bowen Alpern and Fred B. Schneider (1987). "Recognizing Safety and Liveness". In: *Distributed Comput.* 2.3, pp. 117–126. DOI: 10.1007/BF01782772. URL: https://doi.org/10.1007/BF01782772.

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 (1995). *Temporal verification of reactive systems - safety*. Springer. ISBN: 978-0-387-94459-3.

Sven Schewe (2009). "Büchi Complementation Made Tight". In: *26th International Symposium on Theoretical Aspects of Computer Science, STACS 2009, February 26-28, 2009, Freiburg, Germany, Proceedings*. Ed. by Susanne Albers and Jean-Yves Marion. Vol. 3. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Germany, pp. 661–672. DOI: 10.4230/LIPIcs.STACS.2009.1854. URL: https://doi.org/10.4230/LIPIcs.STACS.2009.1854.

A Prasad Sistla (1994). "Safety, liveness and fairness in temporal logic". In: *Formal Aspects of Computing* 6.5, pp. 495–511. DOI: 10.1007/BF01211865.

Moshe Y Vardi (1996). "An automata-theoretic approach to linear temporal logic". In: *Logics for concurrency*. Springer, pp. 238–266.

Moshe Y Vardi and Pierre Wolper (1986). "An automata-theoretic approach to automatic program verification". In: *Proceedings of the First Symposium on Logic in Computer Science*. IEEE Computer Society, pp. 322–331.