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

# The Safety Fragment of Temporal Logics on Infinite Sequences

Lesson 3

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

April 11th, 2024

# Counter-free Automata over finite words

- Let $\mathcal{A} = \langle Q, \Sigma, I, \delta, F \rangle$ be a *deterministic* finite automaton (DFA).
- For each $\langle \sigma_0, \sigma_1, \ldots, \sigma_n \rangle \in \Sigma^*$ and for each $q \in Q$, we define

$$\delta^*(q, \langle \sigma_0, \sigma_1, \ldots, \sigma_n \rangle) = \begin{cases} \delta(q, \sigma_0) & \text{if } n = 0 \\ \delta(\delta^*(q, \langle \sigma_0, \ldots, \sigma_{n-1} \rangle), \sigma_n) & \text{otherwise} \end{cases}$$

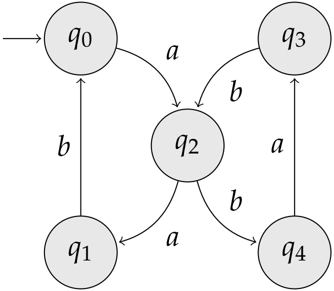
- For any word $\sigma \in \Sigma^*$ and any $i \in \mathbb{N}$, we define $(\sigma)^i$ as the word obtained from $i$ concatenations of $\sigma$.

## Definition (Nontrivial cycle)

A word $\sigma \in \Sigma^*$ (with $\sigma \neq \varepsilon$) defines a *nontrivial cycle* in $\mathcal{A}$ if there exists a state $q \in Q$ such that:

- $\delta^*(q, \sigma) \neq q$
- $\delta^*(q, (\sigma)^i) = q$.

for some $i > 1$.




## Definition (Counter-free DFA)

A DFA $\mathcal{A}$ is called *counter-free* if there are no words that define a nontrivial cycle. We denote this class by cf-DFA.

This automaton is *not* counter-free. The word $ab$ defines the nontrivial cycle:

$$q_0 \xrightarrow{ab} q_4 \xrightarrow{ab} q_2 \xrightarrow{ab} q_0.$$

- The definition of counter-free automaton requires a *deterministic* automaton.
- NBA are not closed under *determinization*.
- We change the type of automata over $\omega$-words which we work with.

$\Rightarrow$ Rabin Automata

**Definition (DRA)**

A *Deterministic Rabin Automaton* (DRA, for short) is a tuple $\langle Q, \Sigma, q_0, \delta, F\rangle$ where

$$F = \langle (A_1, B_1), \ldots, (A_n, B_n)\rangle$$

with $A_i, B_i \subseteq Q$.
A run $\pi := \langle q_0, q_1, \ldots\rangle \in Q^\omega$ is said to be *accepting* iff there exists some $i \in [1, n]$ such that

- $\mathsf{Inf}(\pi) \cap B_i \neq \varnothing$ and
- $\mathsf{Inf}(\pi) \cap A_i = \varnothing$.

# Counter-free Automata over infinite words

**Theorem**

*Deterministic Rabin Automata are equivalent to Nondeterministic Büchi Automata.*

**Definition (Counter-free DRA)**

A DRA $\mathcal{A}$ is called *counter-free* if there are no words that define a nontrivial cycle. We call cf-DRA this class.

**Definition (DRA)**

A *Deterministic Rabin Automaton* (DRA, for short) is a tuple $\langle Q, \Sigma, q_0, \delta, F \rangle$ where

$$F = \langle (A_1, B_1), \ldots, (A_n, B_n) \rangle$$

with $A_i, B_i \subseteq Q$.
A run $\pi := \langle q_0, q_1, \ldots \rangle \in Q^\omega$ is said to be *accepting* iff there exists some $i \in [1, n]$ such that

- $\mathsf{Inf}(\pi) \cap B_i \neq \varnothing$ and
- $\mathsf{Inf}(\pi) \cap A_i = \varnothing$.

**Theorem (Expressive Equivalence for cf-DRA)**

*For each $\omega$-language $\mathcal{L} \subseteq \Sigma^\omega$, it holds that:*

$$\mathcal{L} \textit{ is star-free}$$
$$\textit{iff}$$
$$\mathcal{L} = \mathcal{L}(\mathcal{A}) \textit{ for some } \textsf{cf-DRA } \mathcal{A}$$

**Theorem (Expressive Equivalence for cf-DFA)**

*For each language $\mathcal{L} \subseteq \Sigma^*$, it holds that:*

$$\mathcal{L} \textit{ is star-free}$$
$$\textit{iff}$$
$$\mathcal{L} = \mathcal{L}^{<\omega}(\mathcal{A}) \textit{ for some } \textsf{cf-DFA } \mathcal{A}$$

Reference:

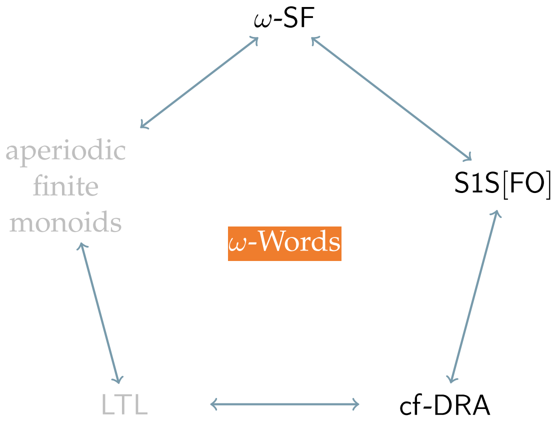
**Robert McNaughton and Seymour A Papert (1971).** ***Counter-Free Automata (MIT research monograph no. 65).*** **The MIT Press**

Reference:

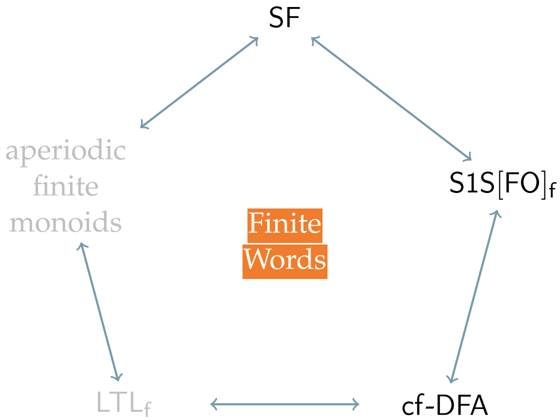
**Wolfgang Thomas (1979). "Star-free regular sets of $\omega$-sequences". In:** ***Information and Control*** **42.2, pp. 148–156. DOI: 10.1016/S0019-9958(79)90629-6**

Reference:

**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-SF
aperiodic finite monoids
S1S[FO]
ω-Words
LTL
cf-DRA


SF
aperiodic finite monoids
$S1S[FO]_f$
Finite Words
$LTL_f$
cf-DFA

# Temporal Logics

Temporal logic is the de-facto standard language for specifying properties of systems in *formal verification* and *artificial intelligence*.

- born in the ′50s as a tool for philosophical argumentation about time

Reference:

**Arthur N Prior (2003). *Time and modality*. John Locke Lecture**

- the idea of its use in formal verification can be traced back to the ′70s

Reference:

**Amir Pnueli (1977). "The temporal logic of programs". In: *18th Annual Symposium on Foundations of Computer Science (sfcs 1977)*. IEEE, pp. 46–57. DOI: 10.1109/SFCS.1977.32**

# Temporal logic in AI

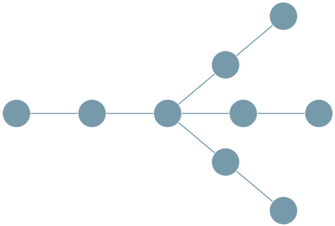
In *artificial intelligence*, when do we need to use *logic* to talk about *time*?

- automated planning
  - temporally extended goals (Bacchus and Kabanza 1998)
  - temporal planning (Fox and Long 2003)
  - timeline-based planning (Della Monica et al. 2017)
- automated synthesis (Jacobs et al. 2017)
- autonomy under uncertainty (Brafman and De Giacomo 2019)
  - specification of goals for planning over MDPs and POMDPs
- reinforcement learning (De Giacomo et al. 2020; Hammond et al. 2021)
  - specification of reward functions and safety conditions
- knowledge representation
  - temporal description logics (Artale et al. 2014)
- multi-agent systems
  - temporal epistemic logics (van Benthem et al. 2009)

There are many choices to be made for the representation of *time*.

**Linear**

**Branching**



# Representing time

There are many choices to be made for the representation of *time*.

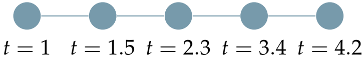
**Infinite**

**Finite**

There are many choices to be made for the representation of *time*.

**Qualitative**

**Real-time**



# Representing time

There are many choices to be made for the representation of *time*.

**Discrete**

**Dense**

# Representing time

There are many choices to be made for the representation of *time*.

We focus here on:

- *linear*-time
- *discrete*-time
- *qualitative*-time
- *infinite*-time
  - sometimes also *finite*-time

*Linear Temporal Logic with Past* (LTL+P, for short) is a *modal* logic.

- introduced by Pnueli in the '70s
- interpreted over discrete, infinite *state sequences* (infinite words)
- it extends classical *propositional* logic
- temporal *operators* are used to talk about how propositions change over time

Let $\mathcal{AP} \coloneqq \{p, q, r, \dots\}$ be a set of *atomic propositions*. The syntax of LTL+P is defined as follows:

| | |
|---|---|
| $\phi \coloneqq p \mid \neg\phi \mid \phi \vee \phi$ | Boolean Modalities |
| $\mid \mathsf{X}\phi \mid \phi \,\mathsf{U}\, \phi$ | Future Temporal Modalities |
| $\mid \mathsf{Y}\phi \mid \phi \,\mathsf{S}\, \phi$ | Past Temporal Modalities |

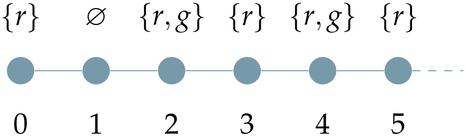
where $p \in \mathcal{AP}$.

- $\mathsf{X}$ is called *tomorrow* (or *next*)
- $\mathsf{U}$ is called *until*
- $\mathsf{Y}$ is called *yesterday* (or *previous*)
- $\mathsf{S}$ is called *since*

# Linear Temporal Logic with Past
## LTL+P Semantics

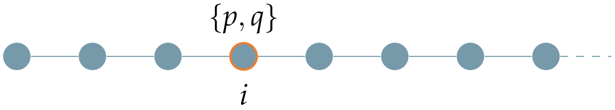
- We focus on the *infinite-time* interpretation of LTL+P.
- Given a set of atomic propositions $\mathcal{AP}$, any LTL+P formula defined over $\mathcal{AP}$ is interpreted over *infinite words* $\sigma \in (2^{\mathcal{AP}})^\omega$.
- In this context, sequences in $(2^{\mathcal{AP}})^\omega$ are also called state sequences or traces.

$$\mathcal{AP} := \{r, g\}$$



We say that $\sigma$ satisfies at position $i$ the LTL+P formula $\phi$, written $\sigma, i \models \phi$, iff:

- $\sigma, i \models p$ iff $p \in \sigma_i$

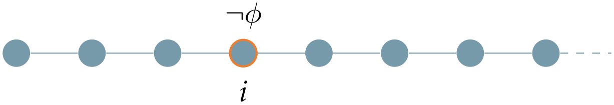



$p$ holds at position $i$

# Linear Temporal Logic with Past
## LTL+P Semantics

We say that $\sigma$ satisfies at position $i$ the LTL+P formula $\phi$, written $\sigma, i \models \phi$, iff:

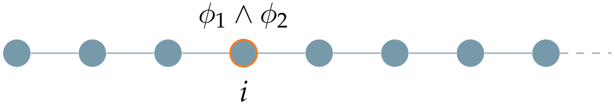
- $\sigma, i \models \neg\phi$ iff $\sigma, i \not\models \phi$




$\phi$ does not hold at position $i$

We say that $\sigma$ satisfies at position $i$ the LTL+P formula $\phi$, written $\sigma, i \models \phi$, iff:

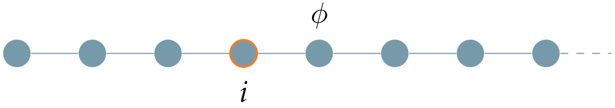
- $\sigma, i \models \phi_1 \wedge \phi_2$ iff $\sigma, i \models \phi_1$ and $\sigma, i \models \phi_2$




$\phi_1$ and $\phi_2$ hold at position $i$

We say that $\sigma$ satisfies at position $i$ the LTL+P formula $\phi$, written $\sigma, i \models \phi$, iff:

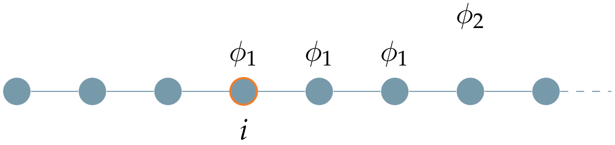
- $\sigma, i \models \mathsf{X}\phi$ iff $\sigma, i+1 \models \phi$




$\phi$ holds at the *next* position of $i$

We say that $\sigma$ satisfies at position $i$ the LTL+P formula $\phi$, written $\sigma, i \models \phi$, iff:

- $\sigma, i \models \phi_1 \mathsf{U} \phi_2$ iff $\exists j \geq i \,.\, \sigma, j \models \phi_2$ and $\forall i \leq k < j \,.\, \sigma, k \models \phi_1$

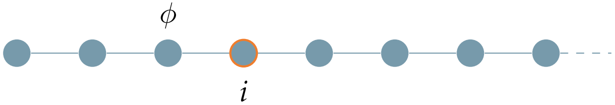



$\phi_1$ holds *until* $\phi_2$ holds

# Linear Temporal Logic with Past
## LTL+P Semantics

We say that $\sigma$ satisfies at position $i$ the LTL+P formula $\phi$, written $\sigma, i \models \phi$, iff:

- $\sigma, i \models \mathsf{Y}\phi$ iff $i > 0$ and $\sigma, i-1 \models \phi$

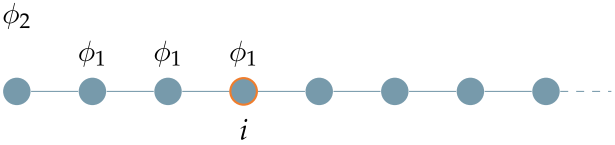



position $i$ has a predecessor and $\phi$ holds at the *previous* position of $i$

**Note:** $\sigma, 0 \models \mathsf{Y}\phi$ is always false.

We say that $\sigma$ satisfies at position $i$ the LTL+P formula $\phi$, written $\sigma, i \models \phi$, iff:

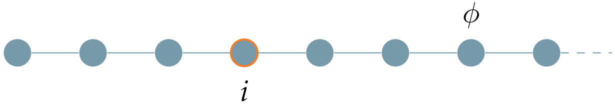
- $\sigma, i \models \phi_1 \mathsf{S}\, \phi_2$ iff $\exists j \leq i \,.\, \sigma, j \models \phi_2$ and $\forall j < k \leq i \,.\, \sigma, k \models \phi_1$




$\phi_1$ holds *since* $\phi_2$ held

*Shortcuts:*

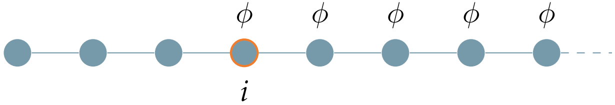
- (*eventually*) $\mathsf{F}\phi \equiv \top \mathsf{U} \phi$




$\phi$ will *eventually* hold

*Shortcuts:*

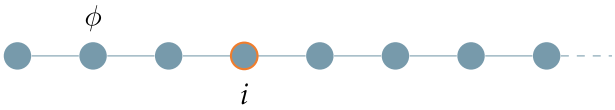
- (*globally*) $\mathsf{G}\phi \equiv \neg\mathsf{F}\neg\phi$




$\phi$ holds *always*

*Shortcuts:*

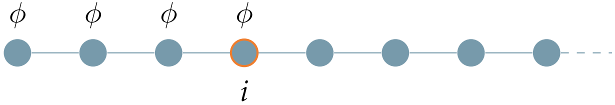
- (*once*) $\mathsf{O}\phi \equiv \top \mathsf{S}\, \phi$




$\phi$ *once* held

*Shortcuts:*

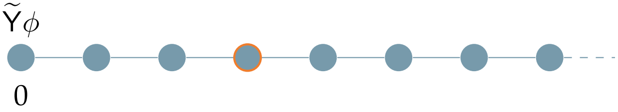
- (*historically*) $\mathsf{H}\phi \equiv \neg\mathsf{O}\neg\phi$




$\phi$ holds *always in the past*

*Shortcuts:*

- (*weak yesterday*) $\widetilde{\mathsf{Y}}\phi \equiv \neg\mathsf{Y}\neg\phi$




$\phi$ holds at the *previous* position of $i$, *if any*

**Note:** $\sigma, i \models \widetilde{\mathsf{Y}}\bot$ is true iff $i = 0$.

## Definition (Negation Normal Form)

We define the $\mathtt{nnf}(\cdot) : \mathsf{LTL} \to \mathsf{LTL}$ (*Negation Normal Form*) function as follows:

- $\mathtt{nnf}(p) = p$
- $\mathtt{nnf}(\phi_1 \wedge \phi_2) = \mathtt{nnf}(\phi_1) \wedge \mathtt{nnf}(\phi_2)$
- $\mathtt{nnf}(\phi_1 \vee \phi_2) = \mathtt{nnf}(\phi_1) \vee \mathtt{nnf}(\phi_2)$
- $\mathtt{nnf}(\mathsf{X}\phi) = \mathsf{X}(\mathtt{nnf}(\phi))$
- $\mathtt{nnf}(\phi_1 \mathbin{\mathsf{U}} \phi_2) = (\mathtt{nnf}(\phi_1)) \mathbin{\mathsf{U}} (\mathtt{nnf}(\phi_2))$
- $\mathtt{nnf}(\phi_1 \mathbin{\mathsf{R}} \phi_2) = (\mathtt{nnf}(\phi_1)) \mathbin{\mathsf{R}} (\mathtt{nnf}(\phi_2))$

The release ($\mathsf{R}$) operator is defined as the negation of the until ($\mathsf{U}$):
$\phi_1 \mathbin{\mathsf{R}} \phi_2 \equiv \neg((\neg\phi_1) \mathbin{\mathsf{U}} (\neg\phi_2))$.
For any $\phi \in \mathsf{LTL}$, the formula $\mathtt{nnf}(\phi)$ has *negation only applied to atomic propositions*.

## Definition (Negation Normal Form)

We define the $\mathtt{nnf}(\cdot) : \mathsf{LTL} \to \mathsf{LTL}$ (*Negation Normal Form*) function as follows:

- $\mathtt{nnf}(\neg p) = \neg p$
- $\mathtt{nnf}(\neg\neg\phi) = \mathtt{nnf}(\phi)$
- $\mathtt{nnf}(\neg(\phi_1 \wedge \phi_2)) = \mathtt{nnf}(\neg\phi_1) \vee \mathtt{nnf}(\neg\phi_2)$
- $\mathtt{nnf}(\neg(\phi_1 \vee \phi_2)) = \mathtt{nnf}(\neg\phi_1) \wedge \mathtt{nnf}(\neg\phi_2)$
- $\mathtt{nnf}(\neg\mathsf{X}\phi) = \mathsf{X}(\mathtt{nnf}(\neg\phi))$
- $\mathtt{nnf}(\neg(\phi_1 \,\mathsf{U}\, \phi_2)) = (\mathtt{nnf}(\neg\phi_1)) \,\mathsf{R}\, (\mathtt{nnf}(\neg\phi_2))$
- $\mathtt{nnf}(\neg(\phi_1 \,\mathsf{R}\, \phi_2)) = (\mathtt{nnf}(\neg\phi_1)) \,\mathsf{U}\, (\mathtt{nnf}(\neg\phi_2))$

For any $\phi \in \mathsf{LTL}$, the formula $\mathtt{nnf}(\phi)$ has *negation only applied to atomic propositions*.

- We say that $\sigma$ *satisfies* $\phi$ (written $\sigma \models \phi$) iff $\sigma, 0 \models \phi$.
- For any LTL+P formula $\phi$, we define *the language of $\phi$ over infinite words* as:

$$\mathcal{L}(\phi) = \{\sigma \in (2^{\mathcal{AP}})^\omega \mid \sigma \models \phi\}$$

- We say that $\phi$ is satisfiable iff $\mathcal{L}(\phi) \neq \varnothing$.
- We say that $\phi$ is valid iff $\mathcal{L}(\phi) = (2^{\mathcal{AP}})^\omega$.

**Example:**

*Each request (r) is eventually followed by a grant (g).*

$$\mathsf{G}(r \rightarrow \mathsf{F}(g))$$

**Example:**

*Each grant (g) is preceeded by a request (r).*

$$\mathsf{G}(g \rightarrow \mathsf{O}(r)))$$

# REFERENCES

**Alessandro Artale et al. (2014). "A Cookbook for Temporal Conceptual Data Modelling with Description Logics".** In: *ACM Trans. Comput. Log.* 15.3, 25:1–25:50. DOI: 10.1145/2629565.

**Fahiem Bacchus and Froduald Kabanza (1998). "Planning for Temporally Extended Goals".** In: *Annals of Mathematics in Artificial Intelligence* 22.1-2, pp. 5–27.

**Ronen I. Brafman and Giuseppe De Giacomo (2019). "Planning for LTLf /LDLf Goals in Non-Markovian Fully Observable Nondeterministic Domains".** In: *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. Ed. by Sarit Kraus. ijcai.org, pp. 1602–1608. DOI: 10.24963/ijcai.2019/222.

**Giuseppe De Giacomo et al. (2020). "Imitation Learning over Heterogeneous Agents with Restraining Bolts".** In: *Proceedings of the 13th International Conference on Automated Planning and Scheduling*. AAAI Press, pp. 517–521.

D. Della Monica et al. (2017). "Bounded Timed Propositional Temporal Logic with Past Captures Timeline-based Planning with Bounded Constraints". In: *Proc. of the 26th International Joint Conference on Artificial Intelligence*, pp. 1008–1014. DOI: 10.24963/ijcai.2017/140.

Maria Fox and Derek Long (2003). "PDDL2.1: An Extension to PDDL for Expressing Temporal Planning Domains". In: *J. Artif. Intell. Res.* 20, pp. 61–124. DOI: 10.1613/jair.1129.

Lewis Hammond et al. (2021). "Multi-Agent Reinforcement Learning with Temporal Logic Specifications". In: *Proceedings of the 20th International Conference on Autonomous Agents and Multiagent Systems*. ACM, pp. 583–592. DOI: 10.5555/3463952.3464024.

Swen Jacobs et al. (2017). "The first reactive synthesis competition (SYNTCOMP 2014)". In: *Int. J. Softw. Tools Technol. Transf.* 19.3, pp. 367–390. DOI: 10.1007/s10009-016-0416-3.

**Robert McNaughton and Seymour A Papert (1971).** ***Counter-Free Automata (MIT research monograph no. 65)*.** The MIT Press.

**Amir Pnueli (1977). "The temporal logic of programs".** In: *18th Annual Symposium on Foundations of Computer Science (sfcs 1977)*. IEEE, pp. 46–57. DOI: 10.1109/SFCS.1977.32.

**Arthur N Prior (2003).** ***Time and modality*.** John Locke Lecture.

**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.

**Wolfgang Thomas (1979). "Star-free regular sets of $\omega$-sequences".** In: *Information and Control* 42.2, pp. 148–156. DOI: 10.1016/S0019-9958(79)90629-6.

**Johan van Benthem et al. (2009). "Merging Frameworks for Interaction".** In: *J. Philos. Log.* 38.5, pp. 491–526. DOI: 10.1007/s10992-008-9099-x.