On the Model Checking of Timed and Weighted Temporal Logics

Patricia Bouyer

LSV, CNRS & ENS Cachan, France

On the Model Checking of Timed ~~and Weighted~~ Temporal Logics

Patricia Bouyer

LSV, CNRS & ENS Cachan, France

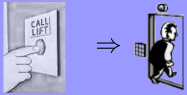
Outline

Outline

Model-checking

system:

property:

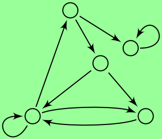


Model-checking

system:

property:

$\Rightarrow$



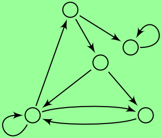
$\mathbf{G}\,(\texttt{request} \rightarrow \mathbf{F}\,\texttt{grant})$

Model-checking

system:

property:

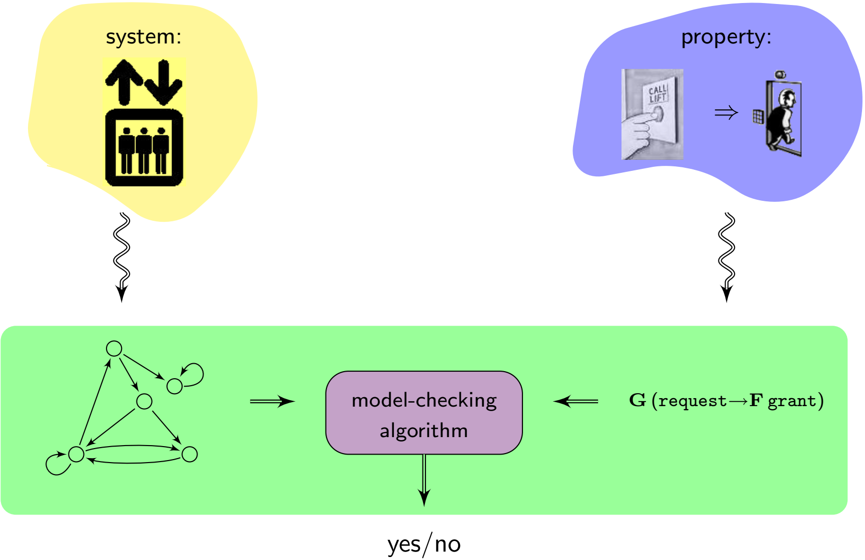
$\Rightarrow$



model-checking algorithm

$\mathbf{G}\,(\mathtt{request} \rightarrow \mathbf{F}\,\mathtt{grant})$

Model-checking



The untimed (linear-time) framework

Linear-time temporal logic [Pnu77]

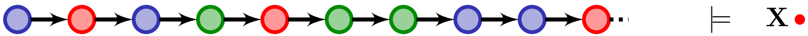
$$\text{LTL} \ni \varphi ::= p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \neg\varphi \mid \mathbf{X}\,\varphi \mid \varphi\,\mathbf{U}\,\varphi$$

[Pnu77] Pnueli. The temporal logic of programs (FOCS'77).

The untimed (linear-time) framework

Linear-time temporal logic [Pnu77]

$$\text{LTL} \ni \varphi ::= p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \neg\varphi \mid \mathbf{X}\,\varphi \mid \varphi\,\mathbf{U}\,\varphi$$



[Pnu77] Pnueli. The temporal logic of programs (FOCS'77).

The untimed (linear-time) framework

Linear-time temporal logic [Pnu77]

$$\text{LTL} \ni \varphi ::= p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \neg\varphi \mid \mathbf{X}\,\varphi \mid \varphi\,\mathbf{U}\,\varphi$$

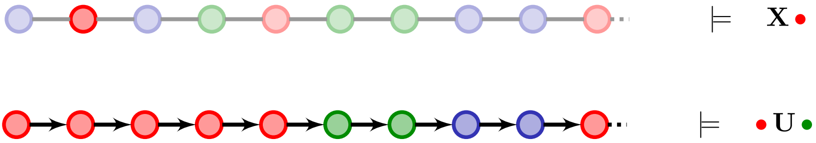
 $\models \mathbf{X}$ ●

[Pnu77] Pnueli. The temporal logic of programs (FOCS'77).

The untimed (linear-time) framework

Linear-time temporal logic [Pnu77]

$$\mathsf{LTL} \ni \varphi ::= p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \neg\varphi \mid \mathbf{X}\,\varphi \mid \varphi\,\mathbf{U}\,\varphi$$

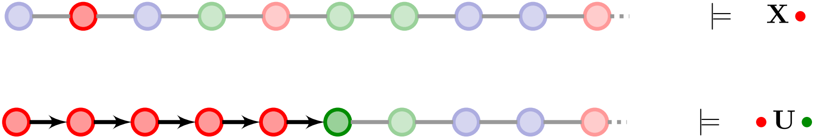


[Pnu77] Pnueli. The temporal logic of programs (FOCS'77).

The untimed (linear-time) framework

Linear-time temporal logic [Pnu77]

$$\mathsf{LTL} \ni \varphi ::= p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \neg\varphi \mid \mathbf{X}\,\varphi \mid \varphi\,\mathbf{U}\,\varphi$$

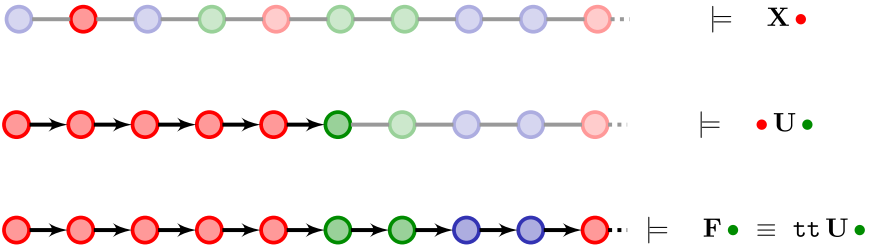


[Pnu77] Pnueli. The temporal logic of programs (FOCS'77).

The untimed (linear-time) framework

Linear-time temporal logic [Pnu77]

$$\mathsf{LTL} \ni \varphi ::= p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \neg\varphi \mid \mathbf{X}\,\varphi \mid \varphi\,\mathbf{U}\,\varphi$$

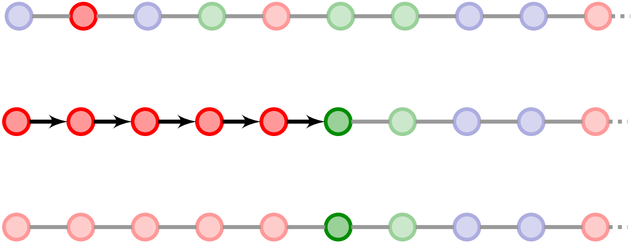


[Pnu77] Pnueli. The temporal logic of programs (FOCS'77).

The untimed (linear-time) framework

Linear-time temporal logic [Pnu77]

$$\mathsf{LTL} \ni \varphi ::= p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \neg\varphi \mid \mathbf{X}\,\varphi \mid \varphi\,\mathbf{U}\,\varphi$$



$\models \quad \mathbf{X} \bullet$

$\models \quad \bullet\,\mathbf{U}\,\bullet$

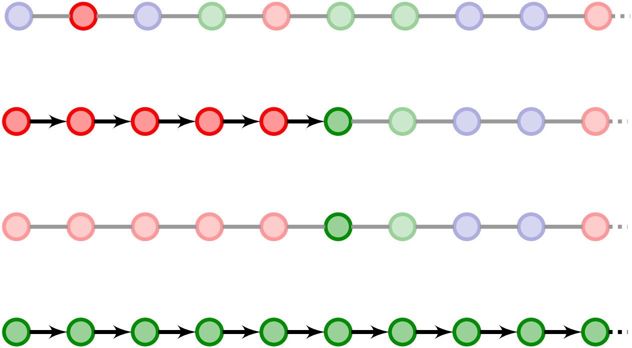
$\models \quad \mathbf{F}\bullet \equiv \mathtt{tt}\,\mathbf{U}\,\bullet$

[Pnu77] Pnueli. The temporal logic of programs (FOCS'77).

The untimed (linear-time) framework

Linear-time temporal logic [Pnu77]

$$\text{LTL} \ni \varphi ::= p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \neg\varphi \mid \mathbf{X}\,\varphi \mid \varphi\,\mathbf{U}\,\varphi$$



$\models \mathbf{X}\,\bullet$

$\models \bullet\,\mathbf{U}\,\bullet$

$\models \mathbf{F}\,\bullet \equiv \mathtt{tt}\,\mathbf{U}\,\bullet$

$\models \mathbf{G}\,\bullet \equiv \neg\mathbf{F}\,\neg\bullet$

[Pnu77] Pnueli. The temporal logic of programs (FOCS'77).

The untimed (linear-time) framework

Linear-time temporal logic [Pnu77]

$$\mathsf{LTL} \ni \varphi \ ::= \ p \ \mid \ \varphi \wedge \varphi \ \mid \ \varphi \vee \varphi \ \mid \ \neg\varphi \ \mid \ \mathbf{X}\,\varphi \ \mid \ \varphi\,\mathbf{U}\,\varphi$$

- response property:

$$\mathbf{G}\,(\bullet \rightarrow \mathbf{F}\,\bullet)$$

[Pnu77] Pnueli. The temporal logic of programs (FOCS'77).

The untimed (linear-time) framework

Linear-time temporal logic [Pnu77]

$$\mathsf{LTL} \ni \varphi \ ::= \ p \ \mid \ \varphi \wedge \varphi \ \mid \ \varphi \vee \varphi \ \mid \ \neg\varphi \ \mid \ \mathbf{X}\,\varphi \ \mid \ \varphi\,\mathbf{U}\,\varphi$$

- response property:

$$\mathbf{G}\,({\color{red}\bullet} \rightarrow \mathbf{F}\,{\color{green}\bullet})$$

- liveness property:

$$\mathbf{G}\,\mathbf{F}\,{\color{green}\bullet}$$

[Pnu77] Pnueli. The temporal logic of programs (FOCS'77).

The untimed (linear-time) framework

Linear-time temporal logic [Pnu77]

$$\text{LTL} \ni \varphi ::= p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \neg\varphi \mid \mathbf{X}\,\varphi \mid \varphi\,\mathbf{U}\,\varphi$$

- response property:

$$\mathbf{G}\,(\bullet \rightarrow \mathbf{F}\,\bullet)$$

- liveness property:

$$\mathbf{G}\,\mathbf{F}\,\bullet$$

- safety property:

$$\mathbf{G}\,\neg\bullet$$

[Pnu77] Pnueli. The temporal logic of programs (FOCS'77).

The untimed (linear-time) framework

Linear-time temporal logic [Pnu77]

$$\mathsf{LTL} \ni \varphi \ ::= \ p \ \mid \ \varphi \wedge \varphi \ \mid \ \varphi \vee \varphi \ \mid \ \neg\varphi \ \mid \ \mathbf{X}\,\varphi \ \mid \ \varphi\,\mathbf{U}\,\varphi$$

- response property:

$$\mathbf{G}\,(\textcolor{red}{\bullet} \rightarrow \mathbf{F}\,\textcolor{green}{\bullet})$$

- liveness property:

$$\mathbf{G}\,\mathbf{F}\,\textcolor{green}{\bullet}$$

- safety property:

$$\mathbf{G}\,\neg\textcolor{red}{\bullet}$$

- a more complex property:

$$(\textcolor{green}{\bullet} \wedge (\mathbf{F}\,\textcolor{blue}{\bullet} \vee \mathbf{G}\,\textcolor{orange}{\bullet}))\,\mathbf{U}\,\textcolor{red}{\bullet}$$

[Pnu77] Pnueli. The temporal logic of programs (FOCS'77).

Adding timing requirements

- Need for timed models
  - the behaviour of most systems depends on time;
  - faithful modelling has to take time into account.

☞ timed automata, time(d) Petri nets, timed process algebras...

Adding timing requirements

- Need for timed models
  - the behaviour of most systems depends on time;
  - faithful modelling has to take time into account.

  ☞ timed automata, time(d) Petri nets, timed process algebras...

- Need for timed specification languages
  - the behaviour of most systems depends on time;
  - untimed specifications are not sufficient (for instance, bounded response timed, etc...)

  ☞ TCTL, MTL, TPTL, timed μ-calculus...

Outline

Metric Temporal Logic (MTL)

[Koy90]

$$\mathsf{MTL} \ni \varphi ::= a \mid \neg\varphi \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \varphi \, \mathbf{U}_I \, \varphi$$

where I is an interval with integral bounds.

[Koy90] Koymans. Specifying real-time properties with metric temporal logic (Real-time systems, 1990).

Metric Temporal Logic (MTL)

[Koy90]

$$\mathsf{MTL} \ni \varphi ::= a \mid \neg\varphi \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \varphi \mathbf{U}_I \varphi$$

where I is an interval with integral bounds.

- This is a timed extension of LTL

[Koy90] Koymans. Specifying real-time properties with metric temporal logic (Real-time systems, 1990).

Metric Temporal Logic (MTL)

[Koy90]

$$\text{MTL} \ni \varphi ::= a \mid \neg\varphi \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \varphi \, \mathbf{U}_I \, \varphi$$

where I is an interval with integral bounds.

- This is a timed extension of LTL
- Can be interpreted over timed words, or over signals
  - this distinction is fundamental

[Koy90] Koymans. Specifying real-time properties with metric temporal logic (Real-time systems, 1990).

Metric Temporal Logic (MTL)

[Koy90]

$$\mathsf{MTL} \ni \varphi ::= a \mid \neg\varphi \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \varphi \,\mathbf{U}_I\, \varphi$$

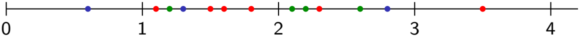
where I is an interval with integral bounds.

- This is a timed extension of LTL
- Can be interpreted over timed words, or over signals
  - this distinction is fundamental
- Can be interpreted over finite or infinite behaviours
  - this distinction is fundamental

[Koy90] Koymans. Specifying real-time properties with metric temporal logic (Real-time systems, 1990).

The pointwise semantics

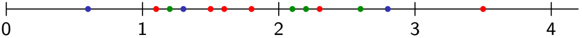
MTL formulas are interpreted over timed words:



$$(\bullet, .6)(\bullet, 1.1)(\bullet, 1.2)(\bullet, 1.3)\ldots$$

The pointwise semantics

MTL formulas are interpreted over timed words:

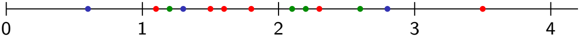


$$(\bullet, .6)(\bullet, 1.1)(\bullet, 1.2)(\bullet, 1.3)\ldots$$

☞ the system is observed only when actions happen

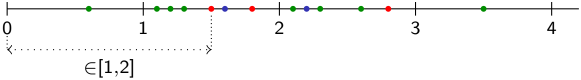
The pointwise semantics

MTL formulas are interpreted over timed words:



$$(\bullet, .6)(\bullet, 1.1)(\bullet, 1.2)(\bullet, 1.3)\ldots$$

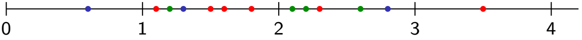
☞ the system is observed only when actions happen



$\models \bullet\, \mathbf{U}_{[1,2]} \bullet$

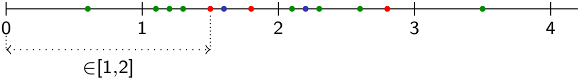
The pointwise semantics

MTL formulas are interpreted over timed words:

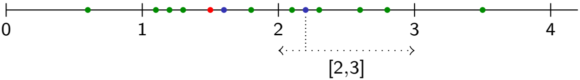


$$(\bullet, .6)(\bullet, 1.1)(\bullet, 1.2)(\bullet, 1.3)\ldots$$

☞ the system is observed only when actions happen



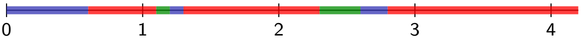
$\models \bullet\ \mathbf{U}_{[1,2]}\ \bullet$



$\not\models\ \mathbf{G}_{[2,3]}\ \bullet$

The continuous semantics

MTL formulas are interpreted over (finitely variable) signals:

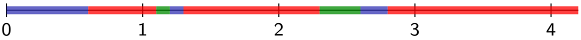


$t \in [0, .6] \mapsto$ ■ $t \in (.6, 1.1) \mapsto$ ■ $t \in [1.1, 1.2) \mapsto$ ■

…

The continuous semantics

MTL formulas are interpreted over (finitely variable) signals:



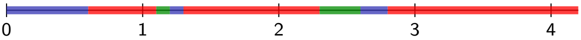
$t \in [0,.6] \mapsto$ ■ $t \in (.6,1.1) \mapsto$ ■ $t \in [1.1,1.2) \mapsto$ ■

...

☞ the system is observed continuously

The continuous semantics

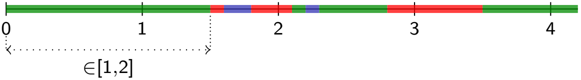
MTL formulas are interpreted over (finitely variable) signals:



$t \in [0,.6] \mapsto$ ■ $t \in (.6,1.1) \mapsto$ ■ $t \in [1.1,1.2) \mapsto$ ■

...

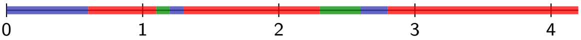
☞ the system is observed continuously



$\models \bullet\ \mathbf{U}_{[1,2]} \bullet$

The continuous semantics

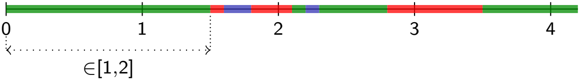
MTL formulas are interpreted over (finitely variable) signals:



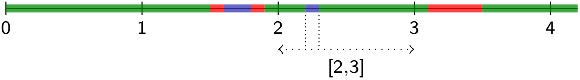
$t \in [0,.6] \mapsto$ ■ $\quad t \in (.6,1.1) \mapsto$ ■ $\quad t \in [1.1,1.2) \mapsto$ ■

...

☞ the system is observed continuously



$\models \bullet\ \mathbf{U}_{[1,2]}\ \bullet$



$\not\models\ \mathbf{G}_{[2,3]}\ \bullet$

Some examples

- "Every problem is followed within 56 time units by an alarm"

$$\mathbf{G}\left(\texttt{problem} \rightarrow \mathbf{F}_{\leqslant 56}\,\texttt{alarm}\right)$$

Some examples

- "Every problem is followed within 56 time units by an alarm"

$$\mathbf{G}\,(\texttt{problem} \rightarrow \mathbf{F}_{\leqslant 56}\,\texttt{alarm})$$

- "Each time there is a problem, it is either repaired within the next 15 time units, or an alarm rings during 3 time units 12 time units later"

$$\mathbf{G}\,(\texttt{problem} \rightarrow (\mathbf{F}_{\leqslant 15}\,\texttt{repair} \vee \mathbf{G}_{[12,15)}\,\texttt{alarm}))$$

Some examples

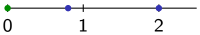
- "Every problem is followed within 56 time units by an alarm"

$$\mathbf{G}\,(\texttt{problem} \rightarrow \mathbf{F}_{\leqslant 56}\,\texttt{alarm})$$

- "Each time there is a problem, it is either repaired within the next 15 time units, or an alarm rings during 3 time units 12 time units later"

$$\mathbf{G}\,(\texttt{problem} \rightarrow (\mathbf{F}_{\leqslant 15}\,\texttt{repair} \vee \mathbf{G}_{[12,15)}\,\texttt{alarm}))$$

- $\mathbf{F}_{=2}\,\texttt{repair}$ *vs* $\mathbf{F}_{=1}\,(\mathbf{F}_{=1}\,\texttt{repair})$



Some examples

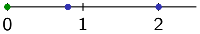
- "Every problem is followed within 56 time units by an alarm"

$$\mathbf{G}\,(\texttt{problem} \rightarrow \mathbf{F}_{\leqslant 56}\,\texttt{alarm})$$

- "Each time there is a problem, it is either repaired within the next 15 time units, or an alarm rings during 3 time units 12 time units later"

$$\mathbf{G}\,(\texttt{problem} \rightarrow (\mathbf{F}_{\leqslant 15}\,\texttt{repair} \vee \mathbf{G}_{[12,15)}\,\texttt{alarm}))$$

- $\mathbf{F}_{=2}\,\texttt{repair}$ *vs* $\mathbf{F}_{=1}\,(\mathbf{F}_{=1}\,\texttt{repair})$



$\models \mathbf{F}_{=2}\,\bullet$ $\not\models \mathbf{F}_{=1}\,(\mathbf{F}_{=1}\,\bullet)$

Some examples

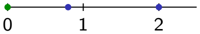
- "Every problem is followed within 56 time units by an alarm"

$$\mathbf{G}\,(\texttt{problem} \rightarrow \mathbf{F}_{\leqslant 56}\,\texttt{alarm})$$

- "Each time there is a problem, it is either repaired within the next 15 time units, or an alarm rings during 3 time units 12 time units later"

$$\mathbf{G}\,(\texttt{problem} \rightarrow (\mathbf{F}_{\leqslant 15}\,\texttt{repair} \vee \mathbf{G}_{[12,15)}\,\texttt{alarm}))$$

- $\mathbf{F}_{=2}\,\texttt{repair}$ *vs* $\mathbf{F}_{=1}\,(\mathbf{F}_{=1}\,\texttt{repair})$



$\models \mathbf{F}_{=2}\,\bullet \quad \not\models \mathbf{F}_{=1}\,(\mathbf{F}_{=1}\,\bullet)$

$\models \mathbf{F}_{=2}\,\bullet \quad \models \mathbf{F}_{=1}\,(\mathbf{F}_{=1}\,\bullet)$

Some examples

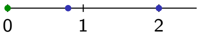
- "Every problem is followed within 56 time units by an alarm"

$$\mathbf{G}\,(\texttt{problem} \rightarrow \mathbf{F}_{\leqslant 56}\,\texttt{alarm})$$

- "Each time there is a problem, it is either repaired within the next 15 time units, or an alarm rings during 3 time units 12 time units later"

$$\mathbf{G}\,(\texttt{problem} \rightarrow (\mathbf{F}_{\leqslant 15}\,\texttt{repair} \vee \mathbf{G}_{[12,15)}\,\texttt{alarm}))$$

- $\mathbf{F}_{=2}\,\texttt{repair}$ *vs* $\mathbf{F}_{=1}\,(\mathbf{F}_{=1}\,\texttt{repair})$



$\models \mathbf{F}_{=2}\,\bullet$ $\quad\not\models \mathbf{F}_{=1}\,(\mathbf{F}_{=1}\,\bullet)$

$\models \mathbf{F}_{=2}\,\bullet$ $\quad\models \mathbf{F}_{=1}\,(\mathbf{F}_{=1}\,\bullet)$

  - in the pointwise semantics, $\mathbf{F}_{=2}\,\bullet \not\equiv \mathbf{F}_{=1}\,\mathbf{F}_{=1}\,\bullet$
  - in the continuous semantics, $\mathbf{F}_{=2}\,\bullet \equiv \mathbf{F}_{=1}\,\mathbf{F}_{=1}\,\bullet$

Some further extensions

- Timed Propositional Temporal Logic (TPTL) [AH89]

  TPTL = LTL + clock variables + clock constraints

[AH89] Alur, Henzinger. A really temporal logic (FOCS'89).

Some further extensions

- Timed Propositional Temporal Logic (TPTL) [AH89]

  TPTL = LTL + clock variables + clock constraints

$$\mathbf{G}\,(\texttt{problem} \rightarrow \mathbf{F}_{\leqslant 56}\,\texttt{alarm}) \quad \equiv \quad \mathbf{G}\,(\texttt{problem} \rightarrow x.\mathbf{F}\,(\texttt{alarm} \wedge x \leqslant 56))$$

[AH89] Alur, Henzinger. A really temporal logic (FOCS'89).

Some further extensions

- Timed Propositional Temporal Logic (TPTL) [AH89]

  TPTL = LTL + clock variables + clock constraints

$$\mathbf{G}\,(\texttt{problem} \rightarrow \mathbf{F}_{\leqslant 56}\,\texttt{alarm}) \quad \equiv \quad \mathbf{G}\,(\texttt{problem} \rightarrow x.\mathbf{F}\,(\texttt{alarm} \wedge x \leqslant 56))$$

$$\mathbf{G}\,(\texttt{problem} \rightarrow x.\mathbf{F}\,(\texttt{alarm} \wedge \mathbf{F}\,(\texttt{failsafe} \wedge x \leqslant 56)))$$

[AH89] Alur, Henzinger. A really temporal logic (FOCS'89).

Some further extensions

- Timed Propositional Temporal Logic (TPTL) [AH89]

  TPTL = LTL + clock variables + clock constraints

$$\mathbf{G}\,(\texttt{problem} \to \mathbf{F}_{\leqslant 56}\,\texttt{alarm}) \quad \equiv \quad \mathbf{G}\,(\texttt{problem} \to x.\mathbf{F}\,(\texttt{alarm} \wedge x \leqslant 56))$$

$$\mathbf{G}\,(\texttt{problem} \to x.\mathbf{F}\,(\texttt{alarm} \wedge \mathbf{F}\,(\texttt{failsafe} \wedge x \leqslant 56)))$$

- MTL+Past: add past-time modalities [AH92]

[AH89] Alur, Henzinger. A really temporal logic (FOCS'89).
[AH92] Alur, Henzinger. Back to the future: towards a theory of timed regular languages (FOCS'92).

Some further extensions

- Timed Propositional Temporal Logic (TPTL) [AH89]

  TPTL = LTL + clock variables + clock constraints

$$\mathbf{G}\,(\mathtt{problem} \rightarrow \mathbf{F}_{\leqslant 56}\,\mathtt{alarm}) \quad \equiv \quad \mathbf{G}\,(\mathtt{problem} \rightarrow x.\mathbf{F}\,(\mathtt{alarm} \wedge x \leqslant 56))$$

$$\mathbf{G}\,(\mathtt{problem} \rightarrow x.\mathbf{F}\,(\mathtt{alarm} \wedge \mathbf{F}\,(\mathtt{failsafe} \wedge x \leqslant 56)))$$

- MTL+Past: add past-time modalities [AH92]

$$\mathbf{G}\,(\mathtt{alarm} \rightarrow \mathbf{F}^{-1}_{\leqslant 56}\,\mathtt{problem})$$

[AH89] Alur, Henzinger. A really temporal logic (FOCS'89).
[AH92] Alur, Henzinger. Back to the future: towards a theory of timed regular languages (FOCS'92).

A note on the expressiveness

Theorem

LTL+Past is as expressive as LTL [Kam68,GPSS80].

[Kam68] Kamp. Tense logic and the theory of linear order (PhD Thesis UCLA 1968).
[GPSS80] Gabbay, Pnueli, Shelah, Stavi. On the temporal analysis of fairness (POPL'80).

A note on the expressiveness

Theorem

LTL+Past is as expressive as LTL [Kam68,GPSS80].

Theorem

MTL is strictly less expressive than MTL+Past and TPTL [BCM05].

[Kam68] Kamp. Tense logic and the theory of linear order (PhD Thesis UCLA 1968).
[GPSS80] Gabbay, Pnueli, Shelah, Stavi. On the temporal analysis of fairness (POPL'80).
[BCM05] Bouyer, Chevalier, Markey. On the expressiveness of MTL and TPTL (FSTTCS'05).

A note on the expressiveness

Theorem

LTL+Past is as expressive as LTL [Kam68,GPSS80].

Theorem

MTL is strictly less expressive than MTL+Past and TPTL [BCM05].

Conjecture in 1990: the TPTL formula

$$\mathbf{G}\,(\bullet \rightarrow x.\mathbf{F}\,(\bullet \wedge \mathbf{F}\,(\bullet \wedge x \leqslant 2)))$$

cannot be expressed in MTL.

[Kam68] Kamp. Tense logic and the theory of linear order (PhD Thesis UCLA 1968).
[GPSS80] Gabbay, Pnueli, Shelah, Stavi. On the temporal analysis of fairness (POPL'80).
[BCM05] Bouyer, Chevalier, Markey. On the expressiveness of MTL and TPTL (FSTTCS'05).

A note on the expressiveness

Theorem

LTL+Past is as expressive as LTL [Kam68,GPSS80].

Theorem

MTL is strictly less expressive than MTL+Past and TPTL [BCM05].

Conjecture in 1990: the TPTL formula

$$\mathbf{G}\,(\bullet \rightarrow x.\mathbf{F}\,(\bullet \wedge \mathbf{F}\,(\bullet \wedge x \leqslant 2)))$$

cannot be expressed in MTL.

- This is true in the pointwise semantics.

[Kam68] Kamp. Tense logic and the theory of linear order (PhD Thesis UCLA 1968).
[GPSS80] Gabbay, Pnueli, Shelah, Stavi. On the temporal analysis of fairness (POPL'80).
[BCM05] Bouyer, Chevalier, Markey. On the expressiveness of MTL and TPTL (FSTTCS'05).

A note on the expressiveness

Theorem

LTL+Past is as expressive as LTL [Kam68,GPSS80].

Theorem

MTL is strictly less expressive than MTL+Past and TPTL [BCM05].

Conjecture in 1990: the TPTL formula

$$\mathbf{G}\,(\bullet \rightarrow x.\mathbf{F}\,(\bullet \wedge \mathbf{F}\,(\bullet \wedge x \leqslant 2)))$$

cannot be expressed in MTL.

- This is true in the pointwise semantics.
- This is wrong in the continuous semantics!

[Kam68] Kamp. Tense logic and the theory of linear order (PhD Thesis UCLA 1968).
[GPSS80] Gabbay, Pnueli, Shelah, Stavi. On the temporal analysis of fairness (POPL'80).
[BCM05] Bouyer, Chevalier, Markey. On the expressiveness of MTL and TPTL (FSTTCS'05).

The TPTL formula

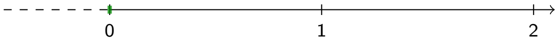
$$\mathbf{G}\left({\color{green}\bullet} \rightarrow x.\mathbf{F}\left({\color{red}\bullet} \wedge \mathbf{F}\left({\color{blue}\bullet} \wedge x \leqslant 2\right)\right)\right)$$

can be expressed in MTL in the continuous semantics

The TPTL formula

$$\mathbf{G}\left(\textcolor{green}{\bullet} \rightarrow x.\mathbf{F}\left(\textcolor{red}{\bullet} \wedge \mathbf{F}\left(\textcolor{blue}{\bullet} \wedge x \leqslant 2\right)\right)\right)$$

can be expressed in MTL in the continuous semantics

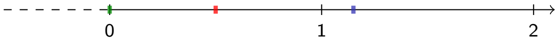


$$\mathbf{G}\ \textcolor{green}{\bullet} \rightarrow \left\{ \right.$$

The TPTL formula

$$\mathbf{G}\left({\color{green}\bullet} \rightarrow x.\mathbf{F}\left({\color{red}\bullet} \wedge \mathbf{F}\left({\color{blue}\bullet} \wedge x \leqslant 2\right)\right)\right)$$

can be expressed in MTL in the continuous semantics

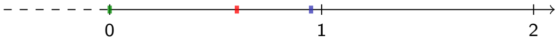


$$\mathbf{G}\ {\color{green}\bullet} \ \rightarrow \begin{cases} \mathbf{F}_{\leqslant 1}\ {\color{red}\bullet} \ \wedge \ \mathbf{F}_{[1,2]}\ {\color{blue}\bullet} \\ \\ \\ \end{cases}$$

The TPTL formula

$$\mathbf{G}\left({\color{green}\bullet} \rightarrow x.\mathbf{F}\left({\color{red}\bullet} \wedge \mathbf{F}\left({\color{blue}\bullet} \wedge x \leqslant 2\right)\right)\right)$$

can be expressed in MTL in the continuous semantics

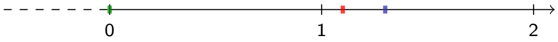


$$\mathbf{G}\ {\color{green}\bullet} \rightarrow \begin{cases} \quad \mathbf{F}_{\leqslant 1}\ {\color{red}\bullet} \ \wedge\ \mathbf{F}_{[1,2]}\ {\color{blue}\bullet} \\ \vee \\ \quad \mathbf{F}_{\leqslant 1}\left({\color{red}\bullet} \ \wedge\ \mathbf{F}_{\leqslant 1}{\color{blue}\bullet}\right) \\ \end{cases}$$

The TPTL formula

$$\mathbf{G}\left({\color{green}\bullet} \rightarrow x.\mathbf{F}\left({\color{red}\bullet} \wedge \mathbf{F}\left({\color{blue}\bullet} \wedge x \leqslant 2\right)\right)\right)$$

can be expressed in MTL in the continuous semantics

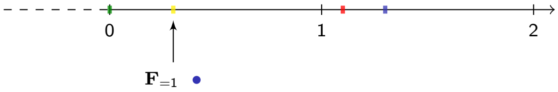


$$\mathbf{G}\ {\color{green}\bullet} \ \rightarrow \begin{cases} \quad \mathbf{F}_{\leqslant 1}\ {\color{red}\bullet} \ \wedge\ \mathbf{F}_{[1,2]}\ {\color{blue}\bullet} \\ \vee \\ \quad \mathbf{F}_{\leqslant 1}\left({\color{red}\bullet} \ \wedge\ \mathbf{F}_{\leqslant 1}{\color{blue}\bullet}\right) \\ \vee \\ \end{cases}$$

The TPTL formula

$$\mathbf{G}\,({\color{green}\bullet} \rightarrow x.\mathbf{F}\,({\color{red}\bullet} \wedge \mathbf{F}\,({\color{blue}\bullet} \wedge x \leqslant 2)))$$

can be expressed in MTL in the continuous semantics

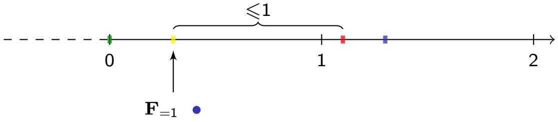


$$\mathbf{G}\ {\color{green}\bullet} \ \rightarrow \left\{ \begin{array}{ll} & \mathbf{F}_{\leqslant 1}\ {\color{red}\bullet} \ \wedge\ \mathbf{F}_{[1,2]}\ {\color{blue}\bullet} \\ \vee & \\ & \mathbf{F}_{\leqslant 1}\,(\ {\color{red}\bullet} \ \wedge\ \mathbf{F}_{\leqslant 1}\,{\color{blue}\bullet}\ ) \\ \vee & \\ & \end{array} \right.$$

The TPTL formula

$$\mathbf{G}\left({\color{green}\bullet} \rightarrow x.\mathbf{F}\left({\color{red}\bullet} \wedge \mathbf{F}\left({\color{blue}\bullet} \wedge x \leqslant 2\right)\right)\right)$$

can be expressed in MTL in the continuous semantics

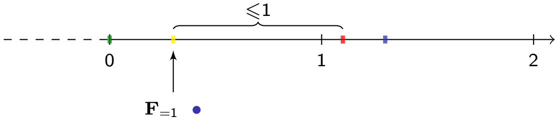


$$\mathbf{G}\ {\color{green}\bullet} \ \rightarrow \ \left\{ \begin{array}{ll} & \mathbf{F}_{\leqslant 1}\ {\color{red}\bullet}\ \wedge\ \mathbf{F}_{[1,2]}\ {\color{blue}\bullet} \\ \vee & \\ & \mathbf{F}_{\leqslant 1}\left(\ {\color{red}\bullet}\ \wedge\ \mathbf{F}_{\leqslant 1}\ {\color{blue}\bullet}\ \right) \\ \vee & \\ & \end{array} \right.$$

The TPTL formula

$$\mathbf{G}\left(\textcolor{green}{\bullet} \rightarrow x.\mathbf{F}\left(\textcolor{red}{\bullet} \wedge \mathbf{F}\left(\textcolor{blue}{\bullet} \wedge x \leqslant 2\right)\right)\right)$$

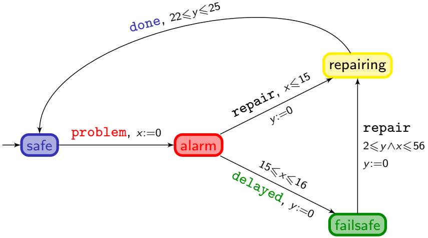
can be expressed in MTL in the continuous semantics



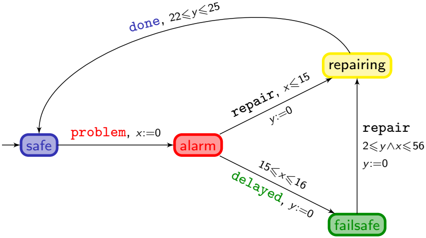
$$\mathbf{G}\ \textcolor{green}{\bullet} \rightarrow \begin{cases} \mathbf{F}_{\leqslant 1}\ \textcolor{red}{\bullet} \wedge \mathbf{F}_{[1,2]}\ \textcolor{blue}{\bullet} \\ \vee \\ \mathbf{F}_{\leqslant 1}\left(\textcolor{red}{\bullet} \wedge \mathbf{F}_{\leqslant 1}\textcolor{blue}{\bullet}\right) \\ \vee \\ \mathbf{F}_{\leqslant 1}\left(\mathbf{F}_{\leqslant 1}\ \textcolor{red}{\bullet} \wedge \mathbf{F}_{=1}\ \textcolor{blue}{\bullet}\right) \end{cases}$$

Outline

Timed automata

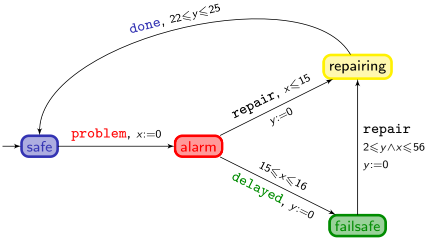


Timed automata



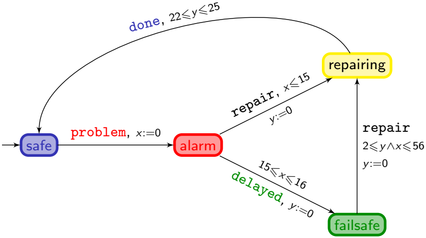
	safe
x	0
y	0

Timed automata



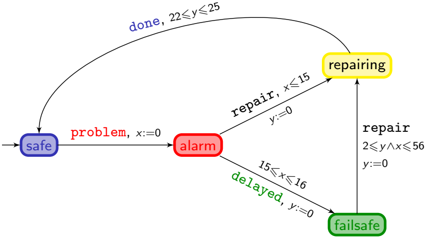
	safe	$\xrightarrow{23}$	safe
x	0		23
y	0		23

Timed automata



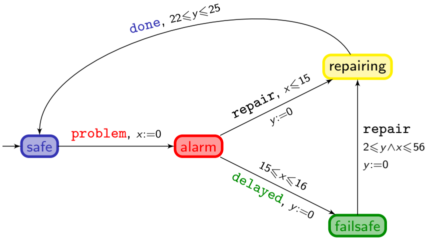
	safe	$\xrightarrow{23}$	safe	$\xrightarrow{\text{problem}}$	alarm
x	0		23		0
y	0		23		23

Timed automata



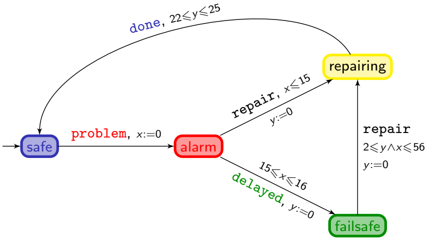
	safe	$\xrightarrow{23}$	safe	$\xrightarrow{\text{problem}}$	alarm	$\xrightarrow{15.6}$	alarm
x	0		23		0		15.6
y	0		23		23		38.6

Timed automata



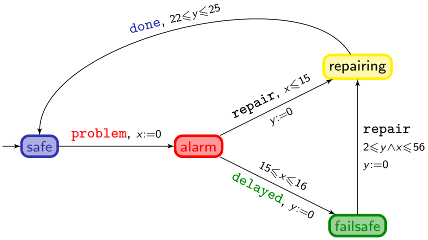
	safe	$\xrightarrow{23}$	safe	$\xrightarrow{\texttt{problem}}$	alarm	$\xrightarrow{15.6}$	alarm	$\xrightarrow{\texttt{delayed}}$	failsafe
x	0		23		0		15.6		15.6
y	0		23		23		38.6		0

Timed automata



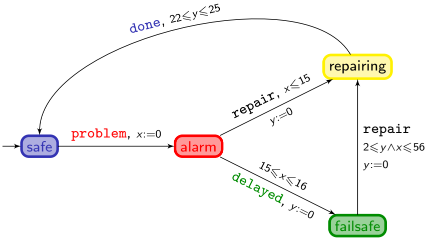
	safe	$\xrightarrow{23}$	safe	$\xrightarrow{\texttt{problem}}$	alarm	$\xrightarrow{15.6}$	alarm	$\xrightarrow{\texttt{delayed}}$	failsafe	$\xrightarrow{2.3}$	failsafe
x	0		23		0		15.6		15.6		17.9
y	0		23		23		38.6		0		2.3

Timed automata



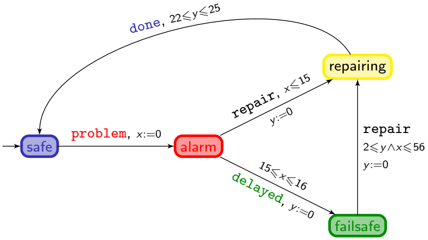
	safe	$\xrightarrow{23}$	safe	$\xrightarrow{\text{problem}}$	alarm	$\xrightarrow{15.6}$	alarm	$\xrightarrow{\text{delayed}}$	failsafe	$\xrightarrow{2.3}$	failsafe	$\xrightarrow{\text{repair}}$	reparation
x	0		23		0		15.6		15.6		17.9		17.9
y	0		23		23		38.6		0		2.3		0

Timed automata



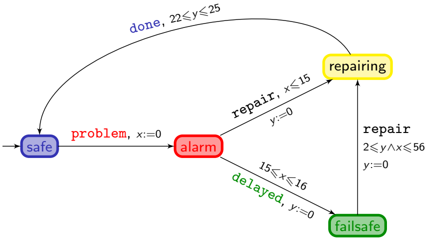
	safe	$\xrightarrow{23}$	safe	$\xrightarrow{\text{problem}}$	alarm	$\xrightarrow{15.6}$	alarm	$\xrightarrow{\text{delayed}}$	failsafe	$\xrightarrow{2.3}$	failsafe	$\xrightarrow{\text{repair}}$	reparation	$\xrightarrow{22.1}$	reparation
x	0		23		0		15.6		15.6		17.9		17.9		40
y	0		23		23		38.6		0		2.3		0		22.1

Timed automata



	safe	$\xrightarrow{23}$	safe	$\xrightarrow{\texttt{problem}}$	alarm	$\xrightarrow{15.6}$	alarm	$\xrightarrow{\texttt{delayed}}$	failsafe	$\xrightarrow{2.3}$	failsafe	$\xrightarrow{\texttt{repair}}$	reparation	$\xrightarrow{22.1}$	reparation	$\xrightarrow{\texttt{done}}$ safe
x	0		23		0		15.6		15.6		17.9		17.9		40	40
y	0		23		23		38.6		0		2.3		0		22.1	22.1

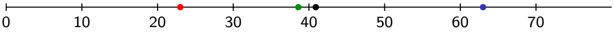
Timed automata



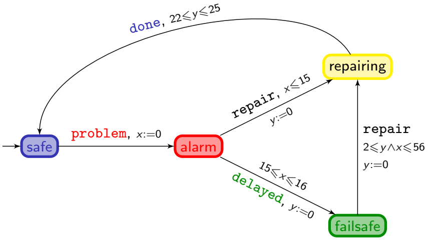
	safe	$\xrightarrow{23}$	safe	$\xrightarrow{\text{problem}}$	alarm	$\xrightarrow{15.6}$	alarm	$\xrightarrow{\text{delayed}}$	failsafe	$\xrightarrow{2.3}$	failsafe	$\xrightarrow{\text{repair}}$	reparation	$\xrightarrow{22.1}$	reparation	$\xrightarrow{\text{done}}$	safe
x	0		23		0		15.6		15.6		17.9		17.9		40		40
y	0		23		23		38.6		0		2.3		0		22.1		22.1

Can be viewed:

- as the timed word (problem,23)(delayed,38.6)(repair,40.9)(done,63)



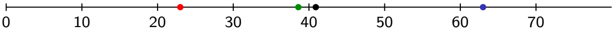
Timed automata



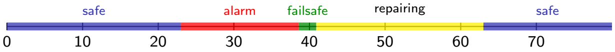
	safe	$\xrightarrow{23}$	safe	$\xrightarrow{\text{problem}}$	alarm	$\xrightarrow{15.6}$	alarm	$\xrightarrow{\text{delayed}}$	failsafe	$\xrightarrow{2.3}$	failsafe	$\xrightarrow{\text{repair}}$	reparation	$\xrightarrow{22.1}$	reparation	$\xrightarrow{\text{done}}$	safe
x	0		23		0		15.6		15.6		17.9		17.9		40		40
y	0		23		23		38.6		0		2.3		0		22.1		22.1

Can be viewed:

- as the timed word (problem,23)(delayed,38.6)(repair,40.9)(done,63)



- as the signal



Basic result on timed automata

Theorem

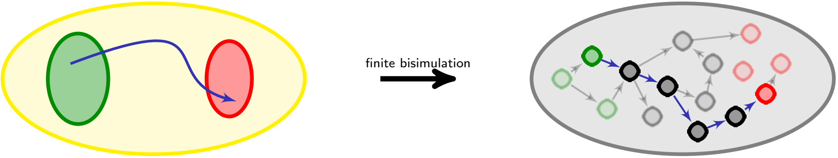
The reachability problem is decidable (and PSPACE-complete) for timed automata [AD94].

[AD94] Alur, Dill. A theory of timed automata (TCS, 1994).

Basic result on timed automata

Theorem

The reachability problem is decidable (and PSPACE-complete) for timed automata [AD94].



timed automaton

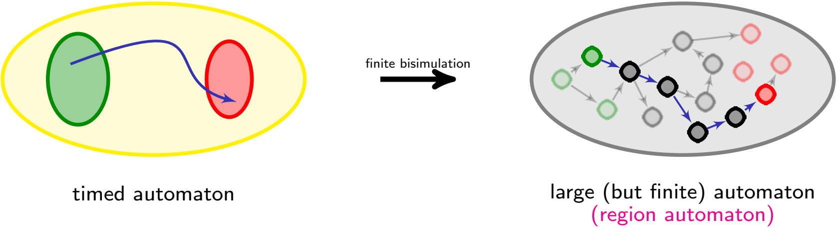
large (but finite) automaton
(region automaton)

[AD94] Alur, Dill. A theory of timed automata (TCS, 1994).

Basic result on timed automata

Theorem

The reachability problem is decidable (and PSPACE-complete) for timed automata [AD94].



☞ It can be extended to model-check TCTL [ACD93].

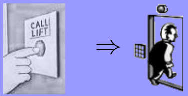
[AD94] Alur, Dill. A theory of timed automata (TCS, 1994).
[ACD93] Alur, Courcoubetis, Dill. Model-checking in dense real-time (I&C, 1993).

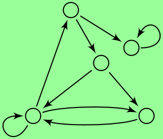
Outline

Back to the model-checking problem

system:

property:

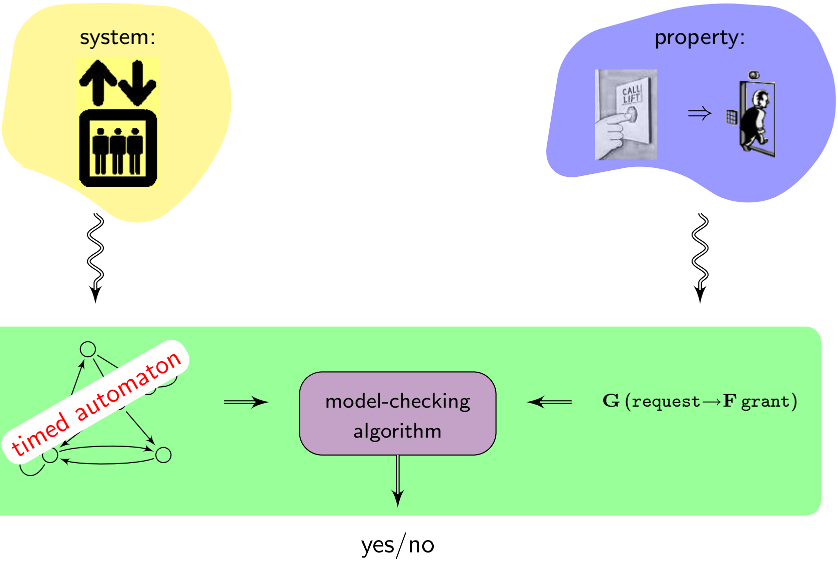




$\Longrightarrow$ model-checking algorithm $\Longleftarrow$ $\mathbf{G}\,(\mathtt{request} \rightarrow \mathbf{F}\,\mathtt{grant})$

yes/no

Back to the model-checking problem

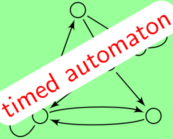


Back to the model-checking problem

system:

property:

$\Rightarrow$



$\Longrightarrow$ model-checking algorithm $\Longleftarrow$ **G** (requ … **F** grant)

MTL formula

yes/no

Results

Theorem

Over finite runs, the model-checking problem is:

	pointwise sem.	continuous sem.
MTL	decidable, NPR [OW05]	undecidable [AFH96]
MTL+Past	undecidable	undecidable
TPTL	undecidable [AH94]	undecidable [AH94]

[OW05] Ouaknine, Worrell. On the decidability of metric temporal logic (LICS'05).
[AFH96] Alur, Feder, Henzinger. The benefits of relaxing punctuality (Journal of the ACM, 1996).
[AH94] Alur, Henzinger. A really temporal logic (Journal of the ACM, 1994).

Results

Theorem

Over finite runs, the model-checking problem is:

	pointwise sem.	continuous sem.
MTL	decidable, NPR [OW05]	undecidable [AFH96]
MTL+Past	undecidable	undecidable
TPTL	undecidable [AH94]	undecidable [AH94]

- Model-checking linear-time timed temporal logics is hard!

[OW05] Ouaknine, Worrell. On the decidability of metric temporal logic (LICS'05).
[AFH96] Alur, Feder, Henzinger. The benefits of relaxing punctuality (Journal of the ACM, 1996).
[AH94] Alur, Henzinger. A really temporal logic (Journal of the ACM, 1994).

Results

Theorem

Over finite runs, the model-checking problem is:

	pointwise sem.	continuous sem.
MTL	decidable, NPR [OW05]	undecidable [AFH96]
MTL+Past	undecidable	undecidable
TPTL	undecidable [AH94]	undecidable [AH94]

- Model-checking linear-time timed temporal logics is hard!
- The gap between branching-time and linear-time dramatically increases in the timed framework...

  (reminder: model-checking TCTL is PSPACE-complete)

[OW05] Ouaknine, Worrell. On the decidability of metric temporal logic (LICS'05).
[AFH96] Alur, Feder, Henzinger. The benefits of relaxing punctuality (Journal of the ACM, 1996).
[AH94] Alur, Henzinger. A really temporal logic (Journal of the ACM, 1994).

Results

Theorem

Over finite runs, the model-checking problem is:

	pointwise sem.	continuous sem.
MTL	decidable, NPR [OW05]	undecidable [AFH96]
MTL+Past	undecidable	undecidable
TPTL	undecidable [AH94]	undecidable [AH94]

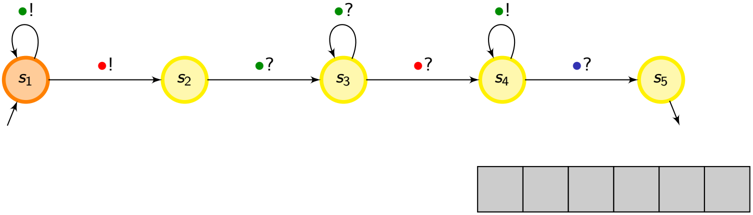
- Model-checking linear-time timed temporal logics is hard!
- The gap between branching-time and linear-time dramatically increases in the timed framework...
  (reminder: model-checking TCTL is PSPACE-complete)

☞ we will explain this high complexity, following [Che07]

[OW05] Ouaknine, Worrell. On the decidability of metric temporal logic (LICS'05).
[AFH96] Alur, Feder, Henzinger. The benefits of relaxing punctuality (Journal of the ACM, 1996).
[AH94] Alur, Henzinger. A really temporal logic (Journal of the ACM, 1994).
[Che07] Chevalier. Logiques pour les systèmes temporisés : contrôle et expressivité (PhD Thesis ENS Cachan, June 2007).

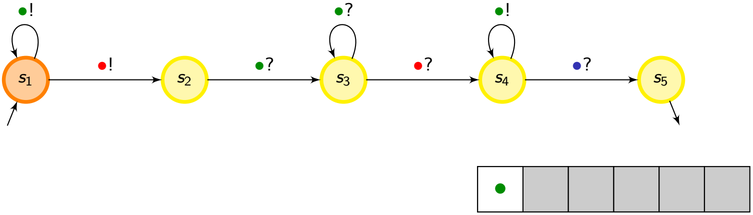
A short visit to channel machines (1)

A channel machine $=$ a finite automaton $+$ a FIFO channel



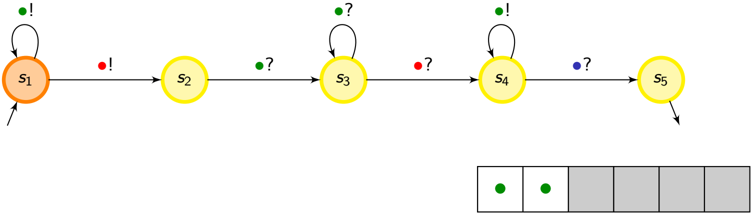
A short visit to channel machines (1)

A channel machine $=$ a finite automaton $+$ a FIFO channel



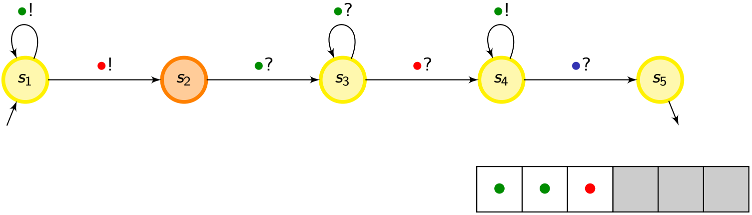
A short visit to channel machines (1)

A channel machine $=$ a finite automaton $+$ a FIFO channel



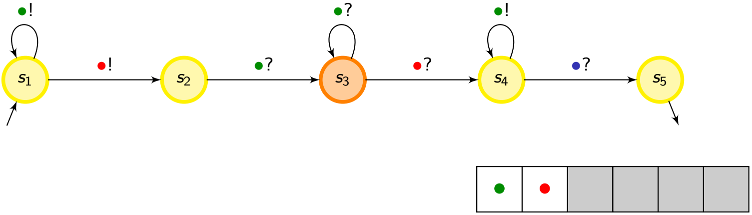
A short visit to channel machines (1)

A channel machine $=$ a finite automaton $+$ a FIFO channel



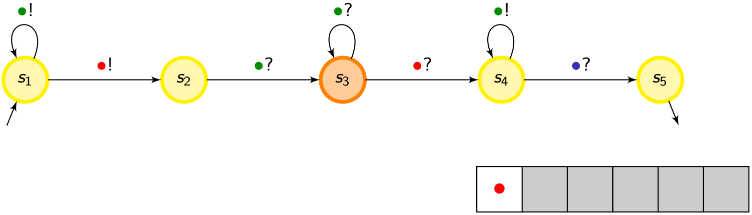
A short visit to channel machines (1)

A channel machine $=$ a finite automaton $+$ a FIFO channel



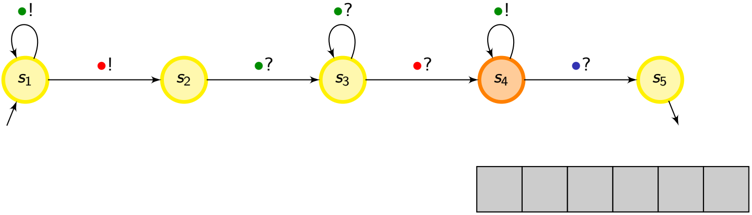
A short visit to channel machines (1)

A channel machine = a finite automaton + a FIFO channel



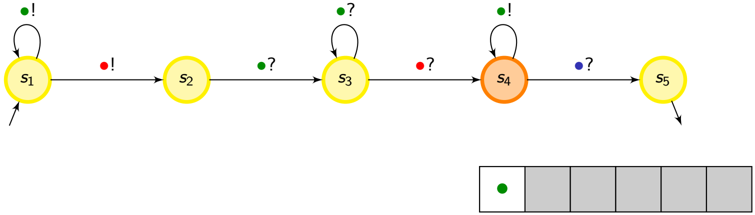
A short visit to channel machines (1)

A channel machine $=$ a finite automaton $+$ a FIFO channel



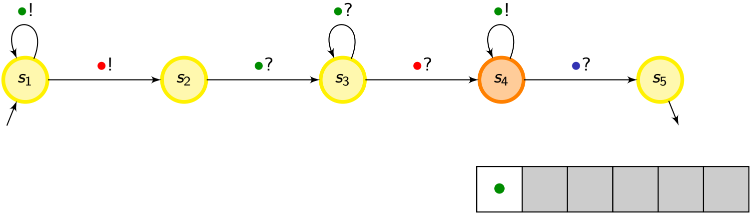
A short visit to channel machines (1)

A channel machine = a finite automaton + a FIFO channel



A short visit to channel machines (1)

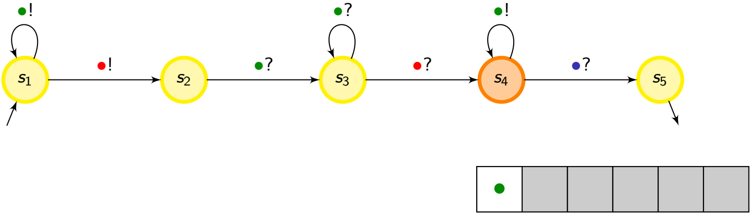
A channel machine = a finite automaton + a FIFO channel



s_5 is not reachable

A short visit to channel machines (1)

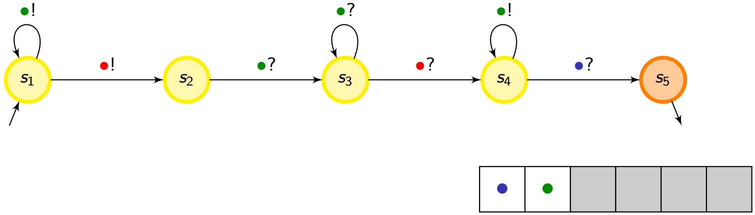
A channel machine = a finite automaton + a FIFO channel



- insertion errors: any letter can appear on the channel at any time

A short visit to channel machines (1)

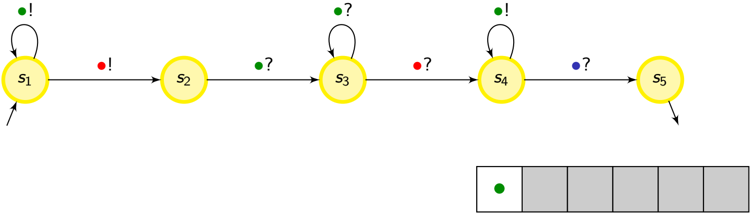
A channel machine = a finite automaton + a FIFO channel



- insertion errors: any letter can appear on the channel at any time

A short visit to channel machines (1)

A channel machine $=$ a finite automaton $+$ a FIFO channel



- insertion errors: any letter can appear on the channel at any time

s_5 is reachable

A short visit to channel machines (2)

Halting problem: is there an execution ending in a halting state?

[BZ83] Brand, Zafiropulo. On communicating finite-state machines (Journal of the ACM, 1983).
[Sch02] Schnoebelen. Verifying lossy channel systems has non-primitive recursive complexity (IPL, 2002).

A short visit to channel machines (2)

Halting problem: is there an execution ending in a halting state?

Proposition

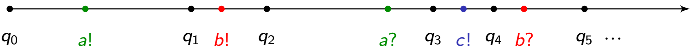
- The halting problem is undecidable for channel machines [BZ83].
- The halting problem is decidable but NPR for channel machines with insertion errors [Sch02].

[BZ83] Brand, Zafiropulo. On communicating finite-state machines (Journal of the ACM, 1983).
[Sch02] Schnoebelen. Verifying lossy channel systems has non-primitive recursive complexity (IPL, 2002).

Channel machines and timed words

We encode an execution of a channel machine as a timed word:

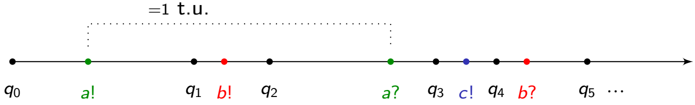
$$(q_0, \varepsilon) \xrightarrow{a!} (q_1, a) \xrightarrow{b!} (q_2, ab) \xrightarrow{a?} (q_3, b) \xrightarrow{c!} (q_4, bc) \xrightarrow{b?} (q_5, c) \cdots$$



Channel machines and timed words

We encode an execution of a channel machine as a timed word:

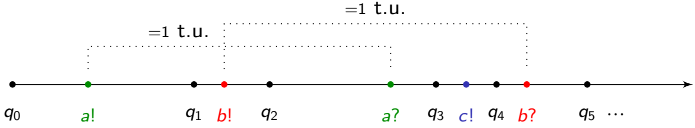
$$(q_0, \varepsilon) \xrightarrow{a!} (q_1, a) \xrightarrow{b!} (q_2, ab) \xrightarrow{a?} (q_3, b) \xrightarrow{c!} (q_4, bc) \xrightarrow{b?} (q_5, c) \cdots$$



Channel machines and timed words

We encode an execution of a channel machine as a timed word:

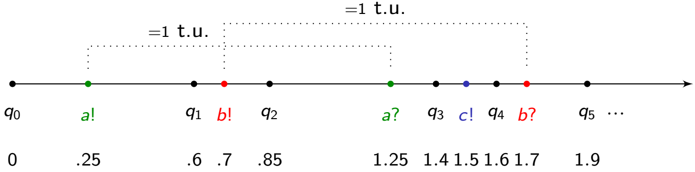
$$(q_0, \varepsilon) \xrightarrow{a!} (q_1, a) \xrightarrow{b!} (q_2, ab) \xrightarrow{a?} (q_3, b) \xrightarrow{c!} (q_4, bc) \xrightarrow{b?} (q_5, c) \cdots$$



Channel machines and timed words

We encode an execution of a channel machine as a timed word:

$$(q_0, \varepsilon) \xrightarrow{a!} (q_1, a) \xrightarrow{b!} (q_2, ab) \xrightarrow{a?} (q_3, b) \xrightarrow{c!} (q_4, bc) \xrightarrow{b?} (q_5, c) \cdots$$



We will give a formula φ such that

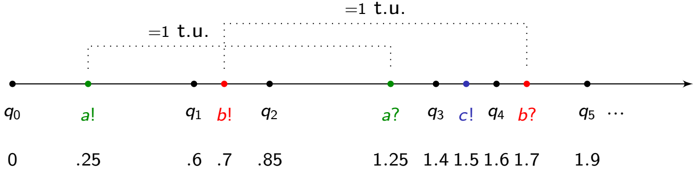
the channel machine* halts iff the formula φ is satisfiable

* possibly with insertion errors

Channel machines and timed words

We encode an execution of a channel machine as a timed word:

$$(q_0, \varepsilon) \xrightarrow{a!} (q_1, a) \xrightarrow{b!} (q_2, ab) \xrightarrow{a?} (q_3, b) \xrightarrow{c!} (q_4, bc) \xrightarrow{b?} (q_5, c) \cdots$$



We will give a formula φ such that

the channel machine* halts iff the formula φ is satisfiable
iff $\mathcal{A}_{\text{univ}} \not\models \neg\varphi$

* possibly with insertion errors

Constraints satisfied by the timed word

- states and actions alternate, and the sequence satisfies the rules of the channel machine: LTL formula

Constraints satisfied by the timed word

- states and actions alternate, and the sequence satisfies the rules of the channel machine: LTL formula
- the channel is FIFO: for every letter a,

$$\mathbf{G}\,(a! \rightarrow \mathbf{F}_{=1}\, a?)$$

Constraints satisfied by the timed word

- states and actions alternate, and the sequence satisfies the rules of the channel machine: LTL formula
- the channel is FIFO: for every letter a,

$$\mathbf{G}\,(a! \rightarrow \mathbf{F}_{=1}\, a?)$$

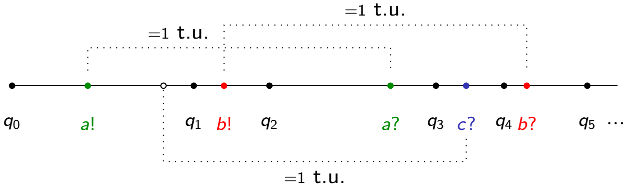
This formula is not sufficient!

Constraints satisfied by the timed word

- states and actions alternate, and the sequence satisfies the rules of the channel machine: LTL formula
- the channel is FIFO: for every letter a,

$$\mathbf{G}\,(a! \rightarrow \mathbf{F}_{=1}\, a?)$$

This formula is not sufficient!

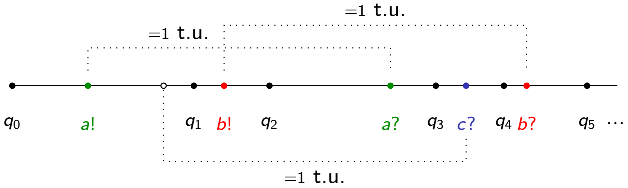


Constraints satisfied by the timed word

- states and actions alternate, and the sequence satisfies the rules of the channel machine: LTL formula
- the channel is FIFO: for every letter a,

$$\mathbf{G}\,(a! \rightarrow \mathbf{F}_{=1}\, a?)$$

This formula is not sufficient!



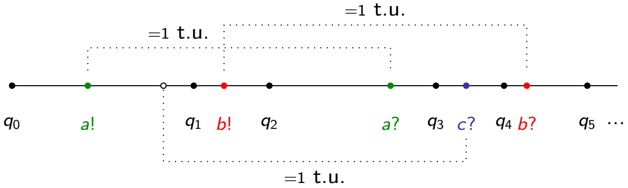
☞ only encodes a channel machine <u>with</u> insertion errors!

Constraints satisfied by the timed word

- states and actions alternate, and the sequence satisfies the rules of the channel machine: LTL formula
- the channel is FIFO: for every letter a,

$$\mathbf{G}\,(a! \rightarrow \mathbf{F}_{=1}\, a?)$$

This formula is not sufficient!



☞ only encodes a channel machine with insertion errors!

☞ model-checking MTL is NPR

We need to express the property:

"Every $a?$-event is preceded one time unit earlier by an $a!$-event"

We need to express the property:

"Every $a?$-event is preceded one time unit earlier by an $a!$-event"

- Why not reverse the previous implication?

$$\mathbf{G}\left((\mathbf{F}_{=1}\, a?) \rightarrow a!\right)$$

We need to express the property:

"Every $a?$-event is preceded one time unit earlier by an $a!$-event"

- Why not reverse the previous implication?

$$\mathbf{G}\left((\mathbf{F}_{=1}\, a?) \longrightarrow a!\right)$$

  - correct in the continuous semantics

We need to express the property:

"Every $a?$-event is preceded one time unit earlier by an $a!$-event"

- Why not reverse the previous implication?

$$\mathbf{G}\left((\mathbf{F}_{=1}\, a?) \rightarrow a!\right)$$

  - correct in the continuous semantics
  - not correct in the pointwise semantics

We need to express the property:

"Every $a?$-event is preceded one time unit earlier by an $a!$-event"

- Why not reverse the previous implication?

$$\mathbf{G}\left((\mathbf{F}_{=1}\, a?) \longrightarrow a!\right)$$

  - correct in the continuous semantics
  - not correct in the pointwise semantics

- Why not look back in the past?

$$\mathbf{G}\left(a? \rightarrow \mathbf{F}^{-1}_{=1}\, a!\right)$$

We need to express the property:

"Every $a?$-event is preceded one time unit earlier by an $a!$-event"

- Why not reverse the previous implication?

$$\mathbf{G}\,((\mathbf{F}_{=1}\,a?) \longrightarrow a!)$$

  - correct in the continuous semantics
  - not correct in the pointwise semantics
- Why not look back in the past?

$$\mathbf{G}\,(a? \rightarrow \mathbf{F}^{-1}_{=1}\,a!)$$

  - correct for MTL+Past (in the continuous and in the pointwise sem.)

We need to express the property:

"Every $a?$-event is preceded one time unit earlier by an $a!$-event"

- Why not reverse the previous implication?

$$\mathbf{G}\,((\mathbf{F}_{=1}\,a?) \longrightarrow a!)$$

  - correct in the continuous semantics
  - not correct in the pointwise semantics
- Why not look back in the past?

$$\mathbf{G}\,(a? \rightarrow \mathbf{F}^{-1}_{=1}\,a!)$$

  - correct for MTL+Past (in the continuous and in the pointwise sem.)
  - no direct translation into MTL

We need to express the property:

"Every $a?$-event is preceded one time unit earlier by an $a!$-event"

- Why not reverse the previous implication?
$$\mathbf{G}\left((\mathbf{F}_{=1}\, a?) \rightarrow a!\right)$$
  - correct in the continuous semantics
  - not correct in the pointwise semantics
- Why not look back in the past?
$$\mathbf{G}\left(a? \rightarrow \mathbf{F}^{-1}_{=1}\, a!\right)$$
  - correct for MTL+Past (in the continuous and in the pointwise sem.)
  - no direct translation into MTL
- A more tricky way:
$$\neg\Big(\mathbf{F}\, x.\mathbf{X}\, y.\mathbf{F}\,(x > 1 \wedge y < 1 \wedge c?)\Big)$$

We need to express the property:

"Every $a?$-event is preceded one time unit earlier by an $a!$-event"

- Why not reverse the previous implication?

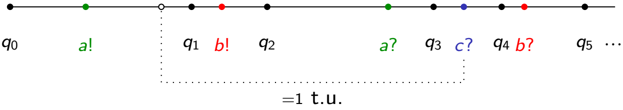
$$\mathbf{G}\,((\mathbf{F}_{=1}\, a?) \rightarrow a!)$$

  - correct in the continuous semantics
  - not correct in the pointwise semantics
- Why not look back in the past?

$$\mathbf{G}\,(a? \rightarrow \mathbf{F}^{-1}_{=1}\, a!)$$

  - correct for MTL+Past (in the continuous and in the pointwise sem.)
  - no direct translation into MTL
- A more tricky way:

$$\neg\Big(\mathbf{F}\, x.\mathbf{X}\, y.\mathbf{F}\,(x > 1 \wedge y < 1 \wedge c?)\Big)$$



We need to express the property:

"Every $a?$-event is preceded one time unit earlier by an $a!$-event"

- Why not reverse the previous implication?

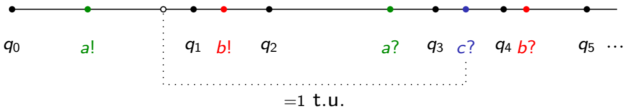
$$\mathbf{G}\left((\mathbf{F}_{=1}\, a?) \rightarrow a!\right)$$

  - correct in the continuous semantics
  - not correct in the pointwise semantics
- Why not look back in the past?

$$\mathbf{G}\left(a? \rightarrow \mathbf{F}^{-1}_{=1}\, a!\right)$$

  - correct for MTL+Past (in the continuous and in the pointwise sem.)
  - no direct translation into MTL
- A more tricky way:

$$\neg\Big(\mathbf{F}\, x.\mathbf{X}\, y.\mathbf{F}\,(x > 1 \wedge y < 1 \wedge c?)\Big)$$



  - this formula is in TPTL (pointwise sem.), not in MTL

What we have proved so far

Theorem

Over finite runs, the model-checking problem is:

	pointwise sem.	continuous sem.
MTL	NPR [OW07]	undecidable [AFH96]
MTL+Past	undecidable	undecidable
TPTL	undecidable [AH94]	undecidable [AH94]

What remains to be proved

Theorem

Over finite runs, the model-checking problem is:

	pointwise sem.	continuous sem.
MTL	decidable, NPR [OW07]	undecidable [AFH96]
MTL+Past	undecidable	undecidable
TPTL	undecidable [AH94]	undecidable [AH94]

From LTL to alternating automata

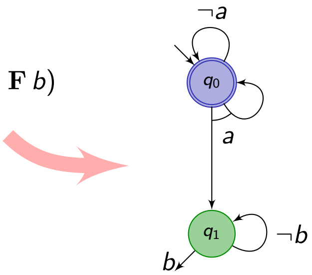
LTL formulas can be turned into alternating (Büchi) automata

$\mathbf{G}\,(a \rightarrow \mathbf{F}\,b)$

From LTL to alternating automata

LTL formulas can be turned into alternating (Büchi) automata

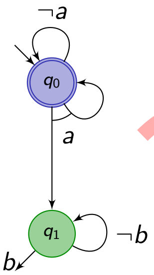
$\mathbf{G}\,(a \rightarrow \mathbf{F}\,b)$

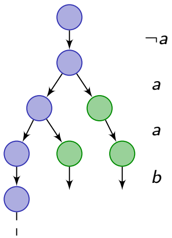


From LTL to alternating automata

LTL formulas can be turned into alternating (Büchi) automata

$\mathbf{G}\,(a \rightarrow \mathbf{F}\,b)$

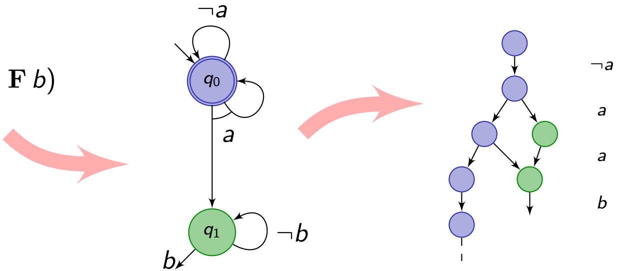




From LTL to alternating automata

LTL formulas can be turned into alternating (Büchi) automata

$$\mathbf{G}\,(a \rightarrow \mathbf{F}\, b)$$



From MTL to alternating timed automata

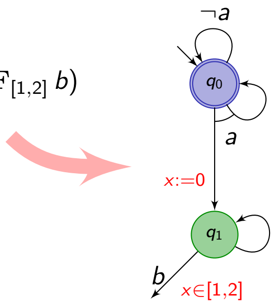
MTL formulas can be turned into linear alternating timed automata

$\mathbf{G}\,(a \rightarrow \mathbf{F}_{[1,2]}\, b)$

From MTL to alternating timed automata

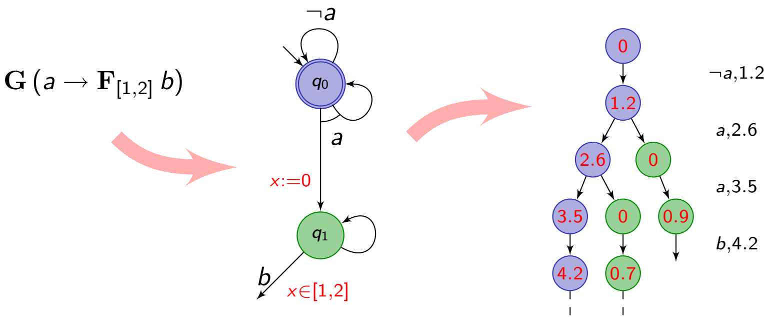
MTL formulas can be turned into linear alternating timed automata

$\mathbf{G}\,(a \rightarrow \mathbf{F}_{[1,2]}\, b)$

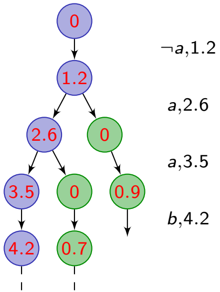


From MTL to alternating timed automata

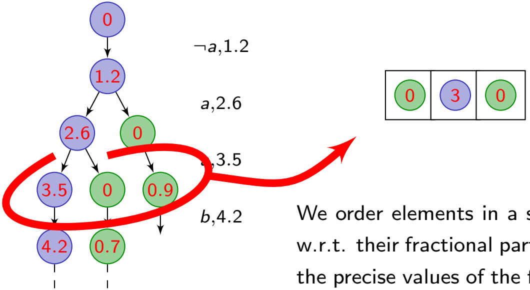
MTL formulas can be turned into linear alternating timed automata



An abstract transition system

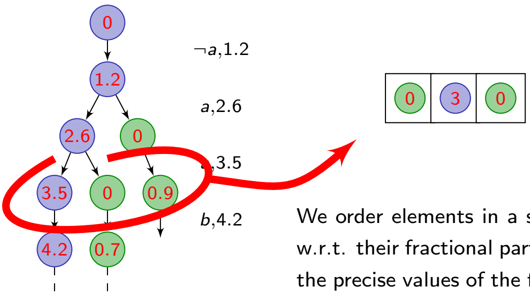


An abstract transition system



We order elements in a slice of the tree w.r.t. their fractional part, and we forget the precise values of the fractional parts.

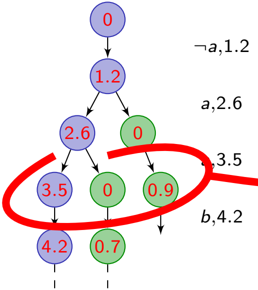
An abstract transition system

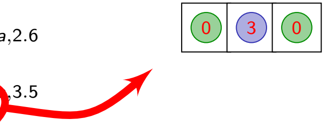


We order elements in a slice of the tree w.r.t. their fractional part, and we forget the precise values of the fractional parts.

☞ this defines an abstract (infinite) transition system

An abstract transition system

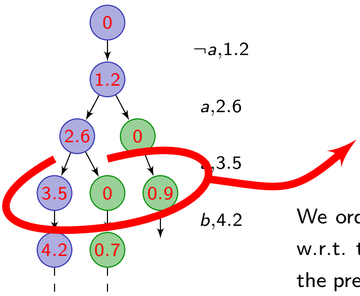


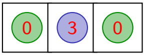


We order elements in a slice of the tree w.r.t. their fractional part, and we forget the precise values of the fractional parts.

- ☞ this defines an abstract (infinite) transition system
- ☞ it is (time-abstract) bisimilar to the transition system of the alternating timed automata

An abstract transition system





We order elements in a slice of the tree w.r.t. their fractional part, and we forget the precise values of the fractional parts.

- ☞ this defines an abstract (infinite) transition system
- ☞ it is (time-abstract) bisimilar to the transition system of the alternating timed automata
- ☞ there is a well quasi-order on the set of abstract configurations (subword relation):

$$\text{higman} \sqsubseteq \text{highmountain}$$

Summary

Theorem

Over finite runs, the model-checking problem is:

	pointwise sem.	continuous sem.
MTL	decidable, NPR [OW05]	undecidable [AFH96]
MTL+Past	undecidable	undecidable
TPTL	undecidable [AH94]	undecidable [AH94]

What about infinite behaviours?

- the previous algorithm cannot be lifted to the infinite behaviours framework

[MH84] Miyano, Hayashi. Alternating finite automata on ω-words (TCS, 1984).
[OW06] Ouaknine, Worrell. On metric temporal logic and faulty Turing machines (FoSSaCS'06).

What about infinite behaviours?

- the previous algorithm cannot be lifted to the infinite behaviours framework
- there is a problem with the accepting condition
  (in the untimed case, we use the Miyano-Hayashi construction [MH84])

[MH84] Miyano, Hayashi. Alternating finite automata on ω-words (TCS, 1984).
[OW06] Ouaknine, Worrell. On metric temporal logic and faulty Turing machines (FoSSaCS'06).

What about infinite behaviours?

- the previous algorithm cannot be lifted to the infinite behaviours framework
- there is a problem with the accepting condition
  (in the untimed case, we use the Miyano-Hayashi construction [MH84])

Theorem

Over infinite runs, the model-checking problem is:

	pointwise sem.	continuous sem.
MTL	undecidable [OW06]*	undecidable [AFH96]
MTL+Past	undecidable	undecidable
TPTL	undecidable [AH94]	undecidable [AH94]

* by reduction of the recurrence problem for channel machines

[MH84] Miyano, Hayashi. Alternating finite automata on ω-words (TCS, 1984).
[OW06] Ouaknine, Worrell. On metric temporal logic and faulty Turing machines (FoSSaCS'06).

Outline

The fragment without punctuality

- The undecidability/NPR proofs heavily rely on punctual constraints.

[AFH96] Alur, Feder, Henzinger. The benefits of relaxing punctuality (Journal of the ACM, 1996).

The fragment without punctuality

- The undecidability/NPR proofs heavily rely on punctual constraints.

  Claim: "Any logic strong enough to express the property $\mathbf{G}\,(\bullet \rightarrow \mathbf{F}_{=1}\,\bullet)$ is undecidable"

[AFH96] Alur, Feder, Henzinger. The benefits of relaxing punctuality (Journal of the ACM, 1996).

The fragment without punctuality

- The undecidability/NPR proofs heavily rely on punctual constraints.

  Claim: "Any logic strong enough to express the property $\mathbf{G}\,(\bullet \rightarrow \mathbf{F}_{=1}\,\bullet)$ is undecidable"

- What if we forbid punctual constraints in MTL?

[AFH96] Alur, Feder, Henzinger. The benefits of relaxing punctuality (Journal of the ACM, 1996).

The fragment without punctuality

- The undecidability/NPR proofs heavily rely on punctual constraints.

  Claim: "Any logic strong enough to express the property $\mathbf{G}\,(\bullet \rightarrow \mathbf{F}_{=1}\,\bullet)$ is undecidable"
- What if we forbid punctual constraints in MTL?

Metric Interval Temporal Logic (MITL): [AFH96]

$$\text{MITL} \ni \varphi \ ::= \ a \mid \neg\varphi \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \varphi\,\mathbf{U}_I\,\varphi$$

with I a non-punctual interval

[AFH96] Alur, Feder, Henzinger. The benefits of relaxing punctuality (Journal of the ACM, 1996).

The fragment without punctuality

- The undecidability/NPR proofs heavily rely on punctual constraints.

  Claim: "Any logic strong enough to express the property $\mathbf{G}\,(\bullet \rightarrow \mathbf{F}_{=1}\,\bullet)$ is undecidable"
- What if we forbid punctual constraints in MTL?

Metric Interval Temporal Logic (MITL): [AFH96]

$$\mathsf{MITL} \ni \varphi \ ::= \ a \mid \neg\varphi \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \varphi\,\mathbf{U}_I\,\varphi$$

with I a non-punctual interval

- Examples:
  - $\mathbf{G}\,(\bullet \rightarrow \mathbf{F}_{=1}\,\bullet)$ is not in MITL

[AFH96] Alur, Feder, Henzinger. The benefits of relaxing punctuality (Journal of the ACM, 1996).

The fragment without punctuality

- The undecidability/NPR proofs heavily rely on punctual constraints.

  Claim: "Any logic strong enough to express the property $\mathbf{G}\,(\color{red}\bullet \color{black}\rightarrow \mathbf{F}_{=1}\,\color{green}\bullet\color{black})$ is undecidable"

- What if we forbid punctual constraints in MTL?

Metric Interval Temporal Logic (MITL): [AFH96]

$$\mathsf{MITL} \ni \varphi \ ::= \ a \ | \ \neg\varphi \ | \ \varphi \vee \varphi \ | \ \varphi \wedge \varphi \ | \ \varphi\,\mathbf{U}_{I}\,\varphi$$

with I a non-punctual interval

- Examples:
  - $\mathbf{G}\,(\color{red}\bullet \color{black}\rightarrow \mathbf{F}_{=1}\,\color{green}\bullet\color{black})$ is not in MITL
  - $\mathbf{G}\,(\color{red}\bullet \color{black}\rightarrow \mathbf{F}_{[1,2]}\,\color{green}\bullet\color{black})$ is in MITL

[AFH96] Alur, Feder, Henzinger. The benefits of relaxing punctuality (Journal of the ACM, 1996).

Model-checking MITL is “easy”

Theorem

The model-checking problem for MITL is EXPSPACE-complete [AFH96].
If constants are encoded in unary, it is even PSPACE-complete [HR04].

[HR04] Hirshfeld, Rabinovich. Logics for real time: decidability and complexity (Fundamenta Informaticae, 2004).

Model-checking MITL is "easy"

Theorem

The model-checking problem for MITL is EXPSPACE-complete [AFH96].
If constants are encoded in unary, it is even PSPACE-complete [HR04].

☞ we can bound the variability of the signals

[HR04] Hirshfeld, Rabinovich. Logics for real time: decidability and complexity (Fundamenta Informaticae, 2004).

Model-checking MITL is “easy”

Theorem

The model-checking problem for MITL is EXPSPACE-complete [AFH96].
If constants are encoded in unary, it is even PSPACE-complete [HR04].

- ☞ we can bound the variability of the signals
- ☞ an MITL formula defines a timed regular language

Example: consider the formula $\varphi = \mathbf{G}_{(0,1)} (\bullet \rightarrow \mathbf{F}_{[1,2]} \bullet)$

[HR04] Hirshfeld, Rabinovich. Logics for real time: decidability and complexity (Fundamenta Informaticae, 2004).

Model-checking MITL is "easy"

Theorem

The model-checking problem for MITL is EXPSPACE-complete [AFH96].
If constants are encoded in unary, it is even PSPACE-complete [HR04].

- ☞ we can bound the variability of the signals
- ☞ an MITL formula defines a timed regular language

Example: consider the formula $\varphi = \mathbf{G}_{(0,1)}\,(\bullet \to \mathbf{F}_{[1,2]}\,\bullet)$

- each time an • occurs within the first time unit, start a new clock, and check that a • occurs between 1 and 2 time units afterwards

[HR04] Hirshfeld, Rabinovich. Logics for real time: decidability and complexity (Fundamenta Informaticae, 2004).

Model-checking MITL is "easy"

Theorem

The model-checking problem for MITL is EXPSPACE-complete [AFH96].
If constants are encoded in unary, it is even PSPACE-complete [HR04].

- ☞ we can bound the variability of the signals
- ☞ an MITL formula defines a timed regular language

Example: consider the formula $\varphi = \mathbf{G}_{(0,1)} (\bullet \rightarrow \mathbf{F}_{[1,2]} \bullet)$

- each time an • occurs within the first time unit, start a new clock, and check that a • occurs between 1 and 2 time units afterwards
- this requires an unbounded number of clocks

[HR04] Hirshfeld, Rabinovich. Logics for real time: decidability and complexity (Fundamenta Informaticae, 2004).

Model-checking MITL is "easy"

Theorem

The model-checking problem for MITL is EXPSPACE-complete [AFH96]. If constants are encoded in unary, it is even PSPACE-complete [HR04].

☞ we can bound the variability of the signals

☞ an MITL formula defines a timed regular language

Example: consider the formula $\varphi = \mathbf{G}_{(0,1)}\,(\bullet \rightarrow \mathbf{F}_{[1,2]}\,\bullet)$

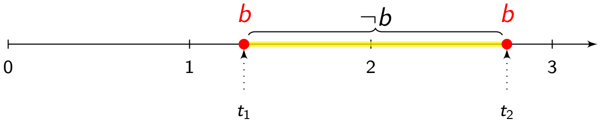
- each time an • occurs within the first time unit, start a new clock, and check that a • occurs between 1 and 2 time units afterwards
- this requires an unbounded number of clocks

☞ something more clever needs to be done

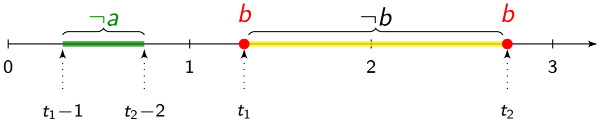
[HR04] Hirshfeld, Rabinovich. Logics for real time: decidability and complexity (Fundamenta Informaticae, 2004).

$$\varphi \;=\; \mathbf{G}_{(0,1)}\,(a \rightarrow \mathbf{F}_{[1,2]}\, b)$$

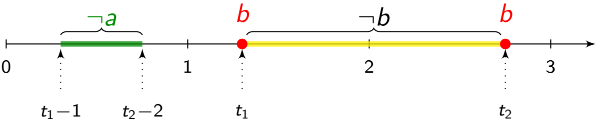
$$\varphi = \mathbf{G}_{(0,1)} (a \rightarrow \mathbf{F}_{[1,2]}\, b)$$

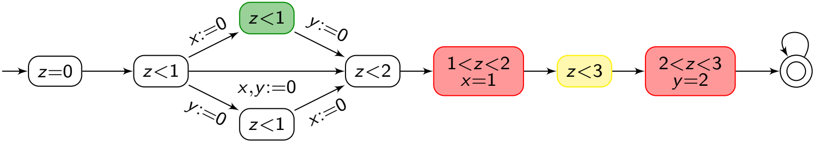


$$\varphi \;=\; \mathbf{G}_{(0,1)}\,(a \rightarrow \mathbf{F}_{[1,2]}\, b)$$

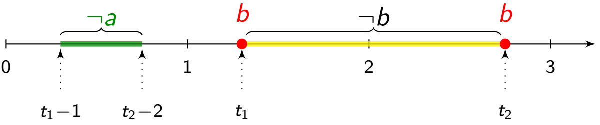


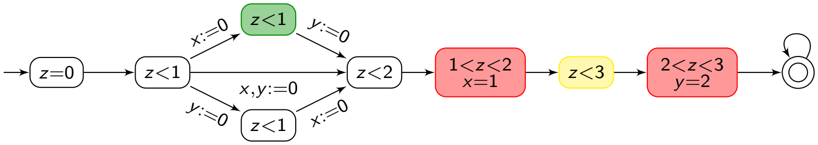
$$\varphi = \mathbf{G}_{(0,1)}\,(a \rightarrow \mathbf{F}_{[1,2]}\,b)$$





$$\varphi \;=\; \mathbf{G}_{(0,1)}\,(a \rightarrow \mathbf{F}_{[1,2]}\, b)$$





☞ This idea can be extended to any formula in MITL

A co-flat fragment of MTL

- Do punctual constraints really need to be banned?

A co-flat fragment of MTL

- Do punctual constraints really need to be banned?
- Does punctuality always lead to undecidability?

A co-flat fragment of MTL

- Do punctual constraints really need to be banned?
- Does punctuality always lead to undecidability?

We define coFlat-$\mathsf{MTL}_{\mathsf{LTL}}$: [BMOW07]

$$\text{coFlat-MTL}_{\text{LTL}} \ni \varphi \ ::= \ a \mid \neg a \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \varphi \, \mathbf{U}_I \, \psi \mid \psi \, \widetilde{\mathbf{U}}_I \, \varphi$$

where I unbounded $\Rightarrow \psi \in$ LTL

[BMOW07] Bouyer, Markey, Ouaknine, Worrell. The cost of punctuality (LICS'07).

A co-flat fragment of MTL

- Do punctual constraints really need to be banned?
- Does punctuality always lead to undecidability?

We define coFlat-MTL$_{\mathsf{LTL}}$: [BMOW07]

$$\text{coFlat-MTL}_{\mathsf{LTL}} \ni \varphi ::= a \mid \neg a \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \varphi\, \mathbf{U}_I\, \psi \mid \psi\, \widetilde{\mathbf{U}}_I\, \varphi$$

where I unbounded $\Rightarrow \psi \in$ LTL

- Examples:
  - $\mathbf{G}\,(\bullet \rightarrow \mathbf{F}_{=1}\,\bullet)$ is in coFlat-MTL$_{\mathsf{LTL}}$

[BMOW07] Bouyer, Markey, Ouaknine, Worrell. The cost of punctuality (LICS'07).

A co-flat fragment of MTL

- Do punctual constraints really need to be banned?
- Does punctuality always lead to undecidability?

We define coFlat-MTL_{LTL}: [BMOW07]

$$\text{coFlat-MTL}_{\text{LTL}} \ni \varphi \ ::= \ a \mid \neg a \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \varphi \, \mathbf{U}_I \, \psi \mid \psi \, \widetilde{\mathbf{U}}_I \, \varphi$$

where I unbounded $\Rightarrow \psi \in$ LTL

- Examples:
  - $\mathbf{G}\,(\bullet \rightarrow \mathbf{F}_{=1}\,\bullet)$ is in coFlat-MTL_{LTL}
  - $\mathbf{F}\,\mathbf{G}_{\leqslant 1}\,\bullet$ is not in coFlat-MTL_{LTL}

[BMOW07] Bouyer, Markey, Ouaknine, Worrell. The cost of punctuality (LICS'07).

A co-flat fragment of MTL

- Do punctual constraints really need to be banned?
- Does punctuality always lead to undecidability?

We define coFlat-$\mathsf{MTL}_{\mathsf{LTL}}$: [BMOW07]

$$\text{coFlat-MTL}_{\text{LTL}} \ni \varphi ::= a \mid \neg a \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \varphi \,\mathbf{U}_I\, \psi \mid \psi \,\widetilde{\mathbf{U}}_I\, \varphi$$

where I unbounded $\Rightarrow \psi \in$ LTL

- Examples:
  - $\mathbf{G}\,(\bullet \rightarrow \mathbf{F}_{=1}\,\bullet)$ is in coFlat-$\mathsf{MTL}_{\mathsf{LTL}}$
  - $\mathbf{F}\,\mathbf{G}_{\leqslant 1}\,\bullet$ is not in coFlat-$\mathsf{MTL}_{\mathsf{LTL}}$
  - coFlat-$\mathsf{MTL}_{\mathsf{LTL}}$ contains Bounded-MTL (all modalities are time-bounded)

[BMOW07] Bouyer, Markey, Ouaknine, Worrell. The cost of punctuality (LICS'07).

Model-checking coFlat-$\mathsf{MTL}_{\mathsf{LTL}}$ is "easy"

Theorem

The model-checking problem for coFlat-$\mathsf{MTL}_{\mathsf{LTL}}$ or Bounded-MTL is EXPSPACE-complete [BMOW07]. If constants are encoded in unary, the model-checking of Bounded-MTL is PSPACE-complete.

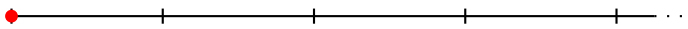
Model-checking coFlat-$\mathsf{MTL}_{\mathsf{LTL}}$ is "easy"

Theorem

The model-checking problem for coFlat-$\mathsf{MTL}_{\mathsf{LTL}}$ or Bounded-MTL is EXPSPACE-complete [BMOW07]. If constants are encoded in unary, the model-checking of Bounded-MTL is PSPACE-complete.

- The variability of a Bounded-MTL formula can be high (doubly-exp.):

$$\varphi_n \equiv {\color{red}\bullet} \wedge \mathbf{G}_{[0,2^n]}\, \varphi_D \qquad \text{with} \qquad \varphi_D = \begin{array}{ll} & ({\color{red}\bullet} \rightarrow \mathbf{F}_{=1}\, ({\color{red}\bullet} \wedge \mathbf{F}_{\leqslant 1}\, {\color{green}\bullet})) \\ \wedge & ({\color{green}\bullet} \rightarrow \mathbf{F}_{=1}\, ({\color{red}\bullet} \wedge \mathbf{F}_{\leqslant 1}\, {\color{green}\bullet})) \end{array}$$



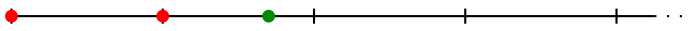
Model-checking coFlat-MTL$_{\text{LTL}}$ is "easy"

Theorem

The model-checking problem for coFlat-MTL$_{\text{LTL}}$ or Bounded-MTL is EXPSPACE-complete [BMOW07]. If constants are encoded in unary, the model-checking of Bounded-MTL is PSPACE-complete.

- The variability of a Bounded-MTL formula can be high (doubly-exp.):

$$\varphi_n \equiv \textcolor{red}{\bullet} \wedge \mathbf{G}_{[0,2^n]}\, \varphi_D \qquad \text{with} \qquad \varphi_D = \begin{array}{ll} & (\textcolor{red}{\bullet} \rightarrow \mathbf{F}_{=1}\,(\textcolor{red}{\bullet} \wedge \mathbf{F}_{\leqslant 1}\, \textcolor{green}{\bullet})) \\ \wedge & (\textcolor{green}{\bullet} \rightarrow \mathbf{F}_{=1}\,(\textcolor{red}{\bullet} \wedge \mathbf{F}_{\leqslant 1}\, \textcolor{green}{\bullet})) \end{array}$$



Model-checking coFlat-$\mathsf{MTL}_{\mathsf{LTL}}$ is “easy”

Theorem

The model-checking problem for coFlat-$\mathsf{MTL}_{\mathsf{LTL}}$ or Bounded-MTL is EXPSPACE-complete [BMOW07]. If constants are encoded in unary, the model-checking of Bounded-MTL is PSPACE-complete.

- The variability of a Bounded-MTL formula can be high (doubly-exp.):

$$\varphi_n \equiv {\color{red}\bullet} \wedge \mathbf{G}_{[0,2^n]}\, \varphi_D \qquad \text{with} \qquad \varphi_D = \begin{array}{ll} & ({\color{red}\bullet} \rightarrow \mathbf{F}_{=1}\,({\color{red}\bullet} \wedge \mathbf{F}_{\leqslant 1}\,{\color{green}\bullet})) \\ \wedge & ({\color{green}\bullet} \rightarrow \mathbf{F}_{=1}\,({\color{red}\bullet} \wedge \mathbf{F}_{\leqslant 1}\,{\color{green}\bullet})) \end{array}$$

Model-checking coFlat-MTL$_{\mathsf{LTL}}$ is "easy"

Theorem

The model-checking problem for coFlat-MTL$_{\mathsf{LTL}}$ or Bounded-MTL is EXPSPACE-complete [BMOW07]. If constants are encoded in unary, the model-checking of Bounded-MTL is PSPACE-complete.

- The variability of a Bounded-MTL formula can be high (doubly-exp.):

$$\varphi_n \equiv \bullet \wedge \mathbf{G}_{[0,2^n]}\, \varphi_D \qquad \text{with} \qquad \varphi_D = \begin{array}{cl} & (\bullet \rightarrow \mathbf{F}_{=1}\,(\bullet \wedge \mathbf{F}_{\leqslant 1}\,\bullet)) \\ \wedge & (\bullet \rightarrow \mathbf{F}_{=1}\,(\bullet \wedge \mathbf{F}_{\leqslant 1}\,\bullet)) \end{array}$$

Model-checking coFlat-MTL$_{\text{LTL}}$ is "easy"

Theorem

The model-checking problem for coFlat-MTL$_{\text{LTL}}$ or Bounded-MTL is EXPSPACE-complete [BMOW07]. If constants are encoded in unary, the model-checking of Bounded-MTL is PSPACE-complete.

- The variability of a Bounded-MTL formula can be high (doubly-exp.):

$$\varphi_n \equiv {\color{red}\bullet} \wedge \mathbf{G}_{[0,2^n]}\, \varphi_D \qquad \text{with} \qquad \varphi_D = \begin{array}{rl} & ({\color{red}\bullet} \rightarrow \mathbf{F}_{=1}\,({\color{red}\bullet} \wedge \mathbf{F}_{\leqslant 1}\,{\color{green}\bullet})) \\ \wedge & ({\color{green}\bullet} \rightarrow \mathbf{F}_{=1}\,({\color{red}\bullet} \wedge \mathbf{F}_{\leqslant 1}\,{\color{green}\bullet})) \end{array}$$

Model-checking coFlat-MTL$_{\text{LTL}}$ is "easy"

Theorem

The model-checking problem for coFlat-MTL$_{\text{LTL}}$ or Bounded-MTL is EXPSPACE-complete [BMOW07]. If constants are encoded in unary, the model-checking of Bounded-MTL is PSPACE-complete.

- The variability of a Bounded-MTL formula can be high (doubly-exp.):

$$\varphi_n \equiv {\color{red}\bullet} \wedge \mathbf{G}_{[0,2^n]}\, \varphi_D \qquad \text{with} \qquad \varphi_D = \begin{array}{rl} & ({\color{red}\bullet} \rightarrow \mathbf{F}_{=1}\,({\color{red}\bullet} \wedge \mathbf{F}_{\leqslant 1}\,{\color{green}\bullet})) \\ \wedge & ({\color{green}\bullet} \rightarrow \mathbf{F}_{=1}\,({\color{red}\bullet} \wedge \mathbf{F}_{\leqslant 1}\,{\color{green}\bullet})) \end{array}$$

- A Bounded-MTL formula may define a non timed-regular language:

$$\mathbf{G}_{\leqslant 1}\,({\color{red}\bullet} \rightarrow \mathbf{F}_{=1}\,{\color{green}\bullet}) \wedge \mathbf{G}_{\leqslant 1}\,{\color{red}\bullet} \wedge \mathbf{G}_{(1,2]}\,{\color{green}\bullet}$$

defines the context-free language $\{{\color{red}\bullet}^n {\color{green}\bullet}^m \mid n \leqslant m\}$.

Algorithm for Bounded-MTL

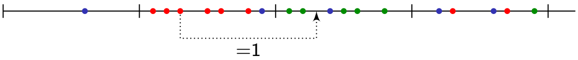
Assume one wants to verify the formula

$$\mathbf{G}_{<2}\Big(\textcolor{red}{\bullet} \longrightarrow \mathbf{F}_{=1}\,\textcolor{green}{\bullet}\Big)$$

Algorithm for Bounded-MTL

Assume one wants to verify the formula

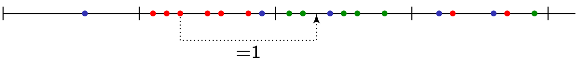
$$\mathbf{G}_{<2}\Big(\textcolor{red}{\bullet} \rightarrow \mathbf{F}_{=1}\,\textcolor{green}{\bullet}\Big)$$



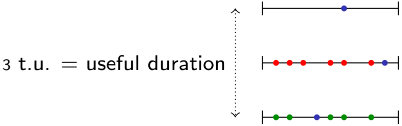
Algorithm for Bounded-MTL

Assume one wants to verify the formula

$$\mathbf{G}_{<2}\left(\bullet \rightarrow \mathbf{F}_{=1}\,\bullet\right)$$



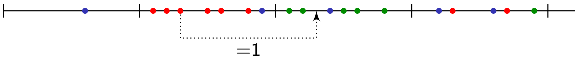
Offline, we stack all ‘relevant’ time units and use a sliding window:



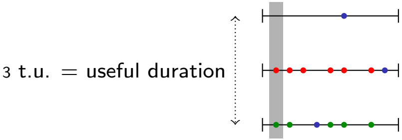
Algorithm for Bounded-MTL

Assume one wants to verify the formula

$$\mathbf{G}_{<2}\left(\color{red}{\bullet} \rightarrow \mathbf{F}_{=1}\,\color{green}{\bullet}\right)$$



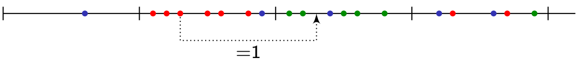
Offline, we stack all ‘relevant’ time units and use a sliding window:



Algorithm for Bounded-MTL

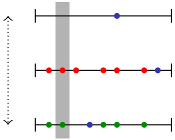
Assume one wants to verify the formula

$$\mathbf{G}_{<2}\left({\color{red}\bullet} \rightarrow \mathbf{F}_{=1} {\color{green}\bullet} \right)$$



Offline, we stack all ‘relevant’ time units and use a sliding window:

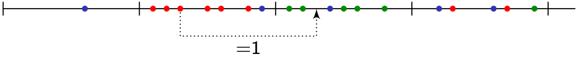
3 t.u. = useful duration



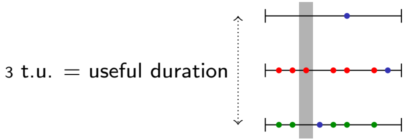
Algorithm for Bounded-MTL

Assume one wants to verify the formula

$$\mathbf{G}_{<2}\left(\bullet \rightarrow \mathbf{F}_{=1} \bullet \right)$$



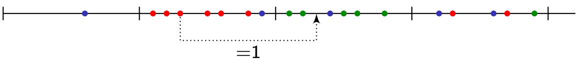
Offline, we stack all ‘relevant’ time units and use a sliding window:



Algorithm for Bounded-MTL

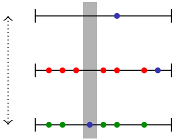
Assume one wants to verify the formula

$$\mathbf{G}_{<2}\Big({\color{red}\bullet} \rightarrow \mathbf{F}_{=1}\,{\color{green}\bullet}\Big)$$



Offline, we stack all ‘relevant’ time units and use a sliding window:

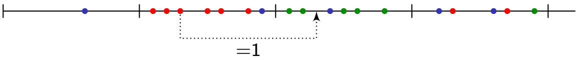
3 t.u. = useful duration



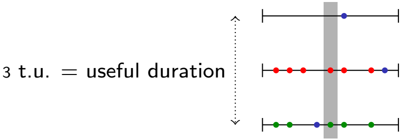
Algorithm for Bounded-MTL

Assume one wants to verify the formula

$$\mathbf{G}_{<2}\left({\color{red}\bullet} \rightarrow \mathbf{F}_{=1}\, {\color{green}\bullet}\right)$$



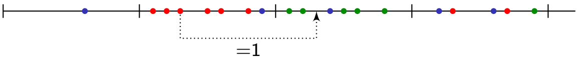
Offline, we stack all ‘relevant’ time units and use a sliding window:



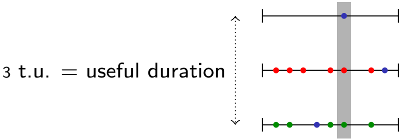
Algorithm for Bounded-MTL

Assume one wants to verify the formula

$$\mathbf{G}_{<2}\Big(\bullet \longrightarrow \mathbf{F}_{=1}\,\bullet\Big)$$



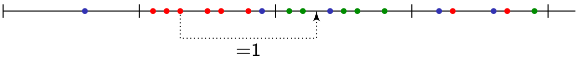
Offline, we stack all ‘relevant’ time units and use a sliding window:



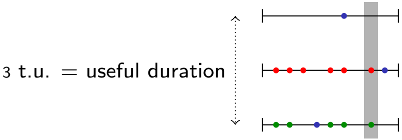
Algorithm for Bounded-MTL

Assume one wants to verify the formula

$$\mathbf{G}_{<2}\left(\textcolor{red}{\bullet} \rightarrow \mathbf{F}_{=1}\,\textcolor{green}{\bullet}\right)$$



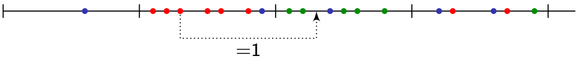
Offline, we stack all ‘relevant’ time units and use a sliding window:



Algorithm for Bounded-MTL

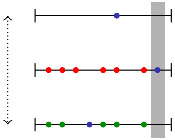
Assume one wants to verify the formula

$$\mathbf{G}_{<2}\Big(\textcolor{red}{\bullet} \rightarrow \mathbf{F}_{=1}\,\textcolor{green}{\bullet}\Big)$$



Offline, we stack all 'relevant' time units and use a sliding window:

3 t.u. = useful duration



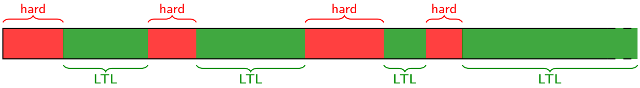
Algorithm for coFlat-$\mathsf{MTL}_{\mathsf{LTL}}$

$\varphi \rightsquigarrow$ alternating timed automata $\mathcal{B}_{\neg\varphi}$ for $\neg\varphi$ with a 'flatness' property

Algorithm for coFlat-$\mathsf{MTL}_{\mathsf{LTL}}$

$\varphi \rightsquigarrow$ alternating timed automata $\mathcal{B}_{\neg\varphi}$ for $\neg\varphi$ with a 'flatness' property

Only counter-examples of the following form need to be looked for:

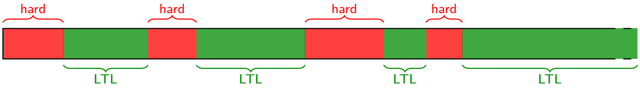


where - the number of hard fragments is at most exponential
- the total duration of hard fragments is at most exponential

Algorithm for coFlat-$\mathsf{MTL}_{\mathsf{LTL}}$

$\varphi \rightsquigarrow$ alternating timed automata $\mathcal{B}_{\neg\varphi}$ for $\neg\varphi$ with a 'flatness' property

Only counter-examples of the following form need to be looked for:



where - the number of hard fragments is at most exponential
- the total duration of hard fragments is at most exponential

- hard fragment = sliding window algorithm
- LTL fragment = finite automaton computation

The ultimate fragment?

- MITL and coFlat-$\mathsf{MTL}_{\mathsf{LTL}}$ have ‘low’ complexities for pretty different reasons...

The ultimate fragment?

- MITL and coFlat-MTL_{LTL} have 'low' complexities for pretty different reasons...
- ... why not mix the two logics?

The ultimate fragment?

- MITL and coFlat-$\mathrm{MTL}_{\mathrm{LTL}}$ have 'low' complexities for pretty different reasons...
- ... why not mix the two logics?

We define coFlat-$\mathrm{MTL}_{\mathrm{MITL}}$: [BMOW08]

$$\text{coFlat-MTL}_{\mathrm{MITL}} \ni \varphi \ ::= \ a \mid \neg a \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \varphi\, \mathbf{U}_I\, \psi \mid \psi\, \widetilde{\mathbf{U}}_I\, \varphi$$

where I unbounded $\Rightarrow \psi \in$ MITL.

[BMOW08] Bouyer, Markey, Ouaknine, Worrell. On Expressiveness and Complexity in Real-time Model Checking (ICALP'08).

The ultimate fragment?

- MITL and coFlat-MTL_{LTL} have 'low' complexities for pretty different reasons...
- ... why not mix the two logics?

We define coFlat-MTL_{MITL}: [BMOW08]

$$\text{coFlat-MTL}_{\text{MITL}} \ni \varphi \ ::= \ a \mid \neg a \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \varphi \, \mathbf{U}_I \, \psi \mid \psi \, \widetilde{\mathbf{U}}_I \, \varphi$$

where I unbounded $\Rightarrow \psi \in$ MITL.

- Examples:
  - $\mathbf{F}\,\mathbf{G}_{\leqslant 1} \bullet$ is in coFlat-MTL_{MITL}
  - $\mathbf{F}\,\mathbf{G}_{=1} \bullet$ is not in coFlat-MTL_{MITL}
  - coFlat-MTL_{MITL} generalizes both logics MITL and coFlat-MTL_{LTL}

[BMOW08] Bouyer, Markey, Ouaknine, Worrell. On Expressiveness and Complexity in Real-time Model Checking (ICALP'08).

Model-checking coFlat-$\mathsf{MTL}_{\mathsf{MITL}}$

Theorem

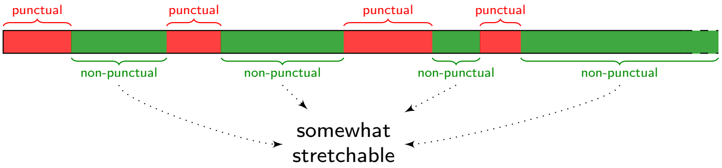
The model-checking problem for coFlat-$\mathsf{MTL}_{\mathsf{MITL}}$ is EXPSPACE-complete [BMOW08], regardless of the encoding of constants in unary or in binary.

Model-checking coFlat-$\mathsf{MTL}_{\mathsf{MITL}}$

Theorem

The model-checking problem for coFlat-$\mathsf{MTL}_{\mathsf{MITL}}$ is EXPSPACE-complete [BMOW08], regardless of the encoding of constants in unary or in binary.

Only counter-examples of the following form need to be looked for:

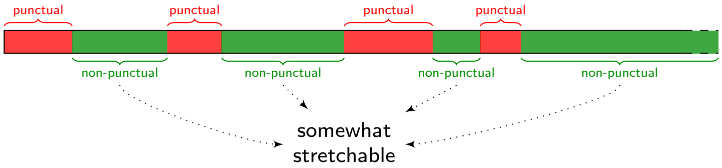


Model-checking coFlat-$\mathsf{MTL}_{\mathsf{MITL}}$

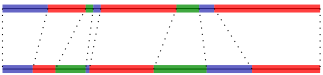
Theorem

The model-checking problem for coFlat-$\mathsf{MTL}_{\mathsf{MITL}}$ is EXPSPACE-complete [BMOW08], regardless of the encoding of constants in unary or in binary.

Only counter-examples of the following form need to be looked for:



Stretchable signal:

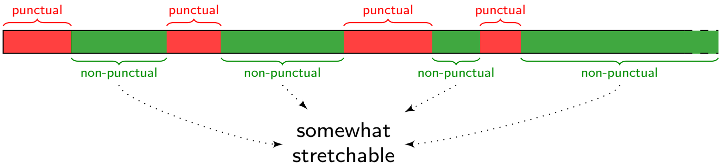


Model-checking coFlat-$\mathsf{MTL}_{\mathsf{MITL}}$

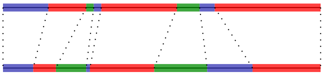
Theorem

The model-checking problem for coFlat-$\mathsf{MTL}_{\mathsf{MITL}}$ is EXPSPACE-complete [BMOW08], regardless of the encoding of constants in unary or in binary.

Only counter-examples of the following form need to be looked for:



Stretchable signal:



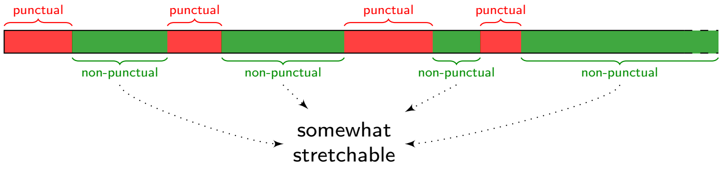
- Any model of an LTL formula is stretchable.

Model-checking coFlat-MTL$_{\text{MITL}}$

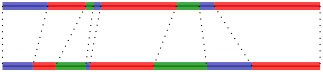
Theorem

The model-checking problem for coFlat-MTL$_{\text{MITL}}$ is EXPSPACE-complete [BMOW08], regardless of the encoding of constants in unary or in binary.

Only counter-examples of the following form need to be looked for:



Stretchable signal:



- Any model of an LTL formula is stretchable.
- Any model of an MITL formula is *somewhat* stretchable.

- non-punctual part: somewhat stretchable
  $\longrightarrow$ transform into LTL constraints

- non-punctual part: somewhat stretchable
  $\longrightarrow$ transform into LTL constraints

- punctual part:

- non-punctual part: somewhat stretchable
  $\longrightarrow$ transform into LTL constraints

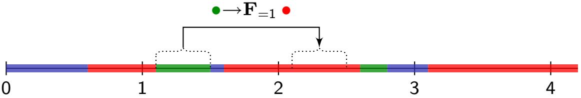
- punctual part: non-stretchable... ☹

- non-punctual part: somewhat stretchable
  $\longrightarrow$ transform into LTL constraints

- punctual part: non-stretchable... ☹
  but not too long... ☺

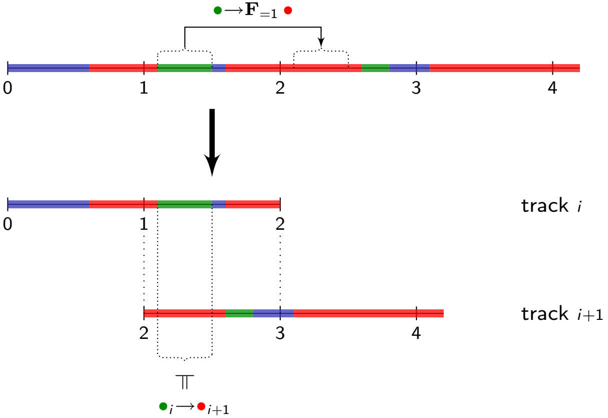
- non-punctual part: somewhat stretchable
  $\longrightarrow$ transform into LTL constraints

- punctual part: non-stretchable... ☹
  but not too long... ☺

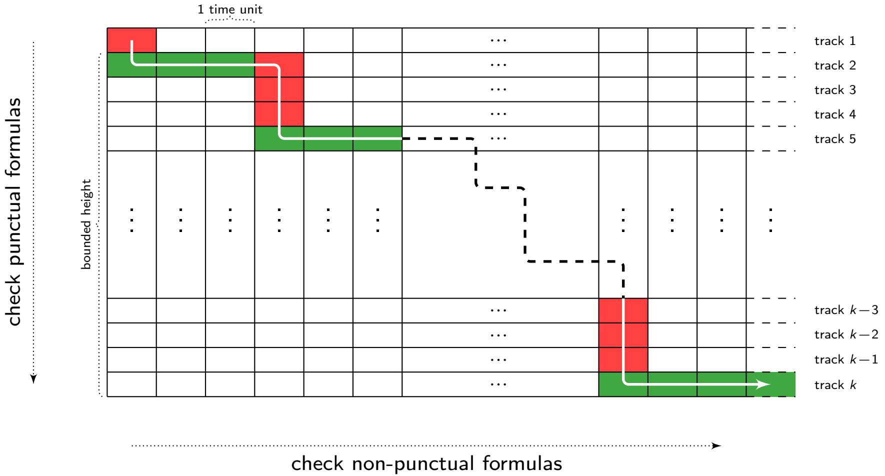


- non-punctual part: somewhat stretchable
  $\longrightarrow$ transform into LTL constraints

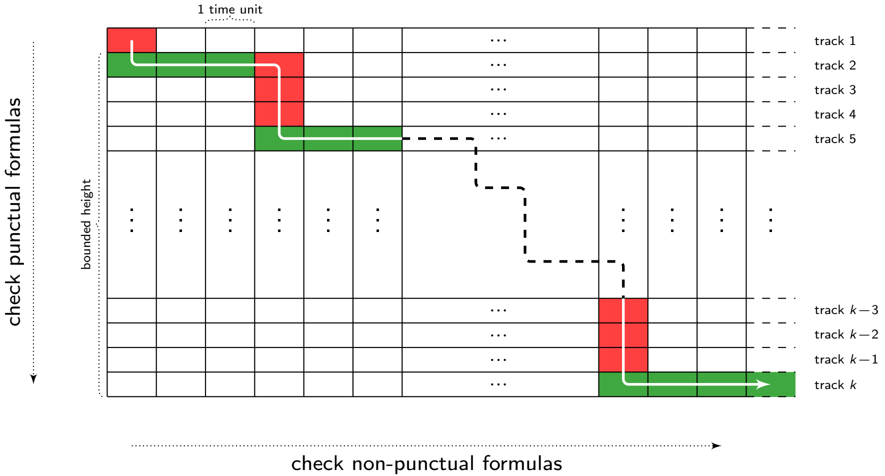
- punctual part: non-stretchable... ☹
  but not too long... ☺



A tableau satisfiability problem



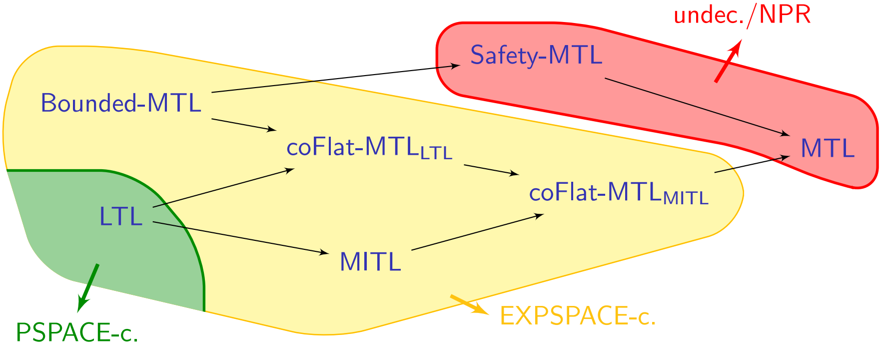
A tableau satisfiability problem



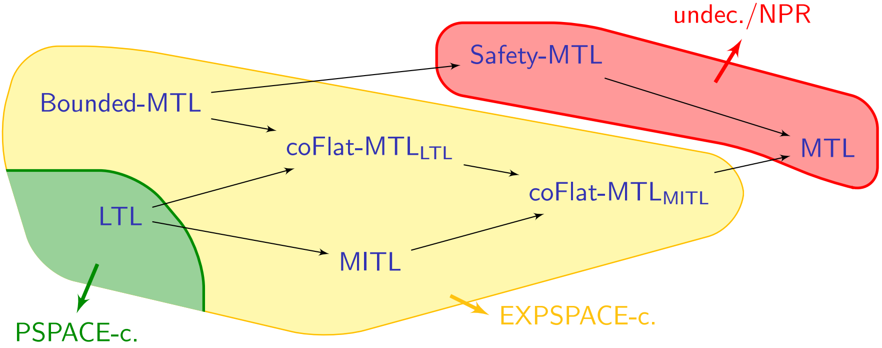
☞ transform into sat. prob. for LTL+Past over $\mathbb{R}_+$ (PSPACE: [Rey04])

[Rey04] Reynolds. The complexity of the temporal logic over the reals (Subm.'04).

The end of the quest for tractable fragments of MTL?



The end of the quest for tractable fragments of MTL?



+ add Positive-MTL [PW09]

[PW09] Parys, Walukiewicz. Weak Alternating Timed Automata *(ICALP'09)*.

Outline

Conclusion

- Recent advances have raised a new interest for linear-time timed temporal logics
    - Not everything is undecidable
    - Some rather ‘efficient’ subclasses
        - non-punctual formulas
        - structurally (co-)flat formulas

Conclusion

- Recent advances have raised a new interest for linear-time timed temporal logics
  - Not everything is undecidable
  - Some rather 'efficient' subclasses
    - non-punctual formulas
    - structurally (co-)flat formulas
- No real data structures do exist for these logics.

Going further with quantities...

- System resources might be relevant and even crucial information
  - energy consumption, memory usage, price to pay, bandwidth, ...

Going further with quantities...

- System resources might be relevant and even crucial information
  - energy consumption, memory usage, price to pay, bandwidth, ...
- We need to integrate those aspects in models and in logics
  - *Models:* hybrid automata, timed automata with observer variables
  - *Logics:* extensions of classical temporal logics with quantitative constraints on the observer variables

Going further with quantities...

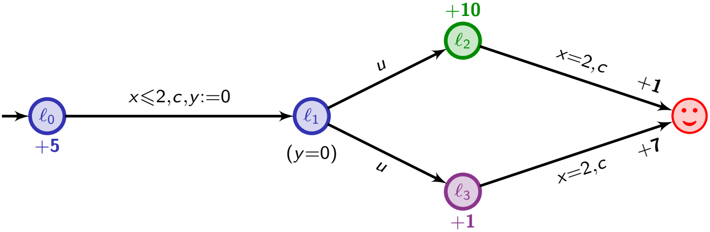
- System resources might be relevant and even crucial information
  - energy consumption, memory usage, price to pay, bandwidth, ...
- We need to integrate those aspects in models and in logics
  - *Models:* hybrid automata, timed automata with observer variables
  - *Logics:* extensions of classical temporal logics with quantitative constraints on the observer variables
- Very few decidability results, unless strong restrictions..

Going further with quantities...

- System resources might be relevant and even crucial information
  - energy consumption, memory usage, price to pay, bandwidth, ...
- We need to integrate those aspects in models and in logics
  - *Models:* hybrid automata, timed automata with observer variables
  - *Logics:* extensions of classical temporal logics with quantitative constraints on the observer variables
- Very few decidability results, unless strong restrictions..

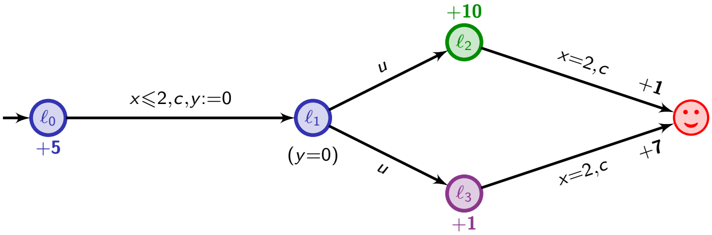
  ... but this is a very active field of research!

Weighted/priced timed automata [ALP01,BFH+01]



[ALP01] Alur, La Torre, Pappas. Optimal paths in weighted timed automata *(HSCC'01)*.
[BFH+01] Behrmann, Fehnker, Hune, Larsen, Pettersson, Romijn, Vaandrager. Minimum-cost reachability in priced timed automata *(HSCC'01)*.

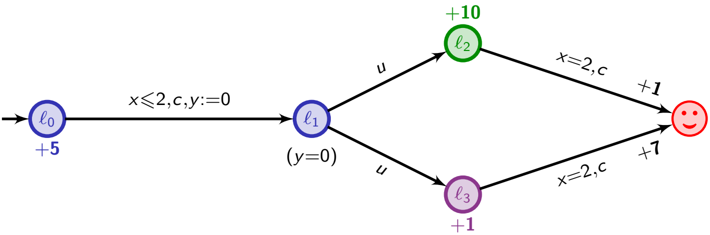
Weighted/priced timed automata [ALP01,BFH+01]



	ℓ_0	$\xrightarrow{1.3}$	ℓ_0	$\xrightarrow{c}$	ℓ_1	$\xrightarrow{u}$	ℓ_3	$\xrightarrow{0.7}$	ℓ_3	$\xrightarrow{c}$	☺
x	0		1.3		1.3		1.3		2		
y	0		1.3		0		0		0.7		

[ALP01] Alur, La Torre, Pappas. Optimal paths in weighted timed automata *(HSCC'01)*.
[BFH+01] Behrmann, Fehnker, Hune, Larsen, Pettersson, Romijn, Vaandrager. Minimum-cost reachability in priced timed automata *(HSCC'01)*.

Weighted/priced timed automata [ALP01,BFH+01]

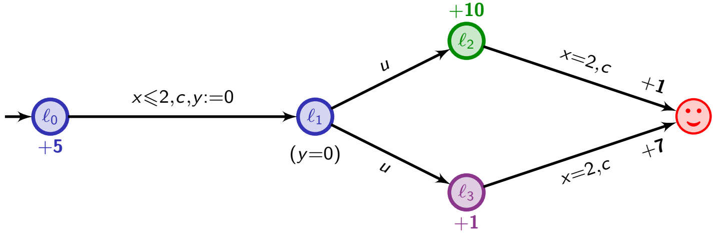


$$\begin{array}{ccccccccccccc} & \ell_0 & \xrightarrow{1.3} & \ell_0 & \xrightarrow{c} & \ell_1 & \xrightarrow{u} & \ell_3 & \xrightarrow{0.7} & \ell_3 & \xrightarrow{c} & ☺ \\ x & 0 & & 1.3 & & 1.3 & & 1.3 & & 2 & & \\ y & 0 & & 1.3 & & 0 & & 0 & & 0.7 & & \end{array}$$

cost :

[ALP01] Alur, La Torre, Pappas. Optimal paths in weighted timed automata *(HSCC'01)*.
[BFH+01] Behrmann, Fehnker, Hune, Larsen, Pettersson, Romijn, Vaandrager. Minimum-cost reachability in priced timed automata *(HSCC'01)*.

Weighted/priced timed automata [ALP01,BFH+01]

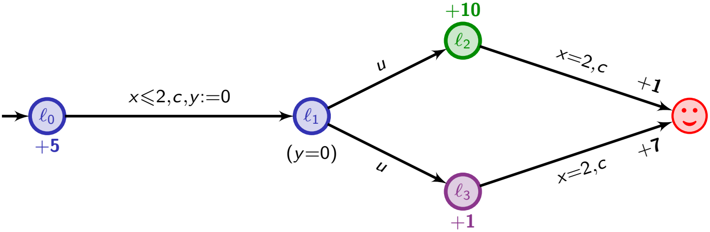


	ℓ_0	$\xrightarrow{1.3}$	ℓ_0	$\xrightarrow{c}$	ℓ_1	$\xrightarrow{u}$	ℓ_3	$\xrightarrow{0.7}$	ℓ_3	$\xrightarrow{c}$	☺
x	0		1.3		1.3		1.3		2		
y	0		1.3		0		0		0.7		

cost : 6.5

[ALP01] Alur, La Torre, Pappas. Optimal paths in weighted timed automata *(HSCC'01)*.
[BFH+01] Behrmann, Fehnker, Hune, Larsen, Pettersson, Romijn, Vaandrager. Minimum-cost reachability in priced timed automata *(HSCC'01)*.

Weighted/priced timed automata [ALP01,BFH+01]

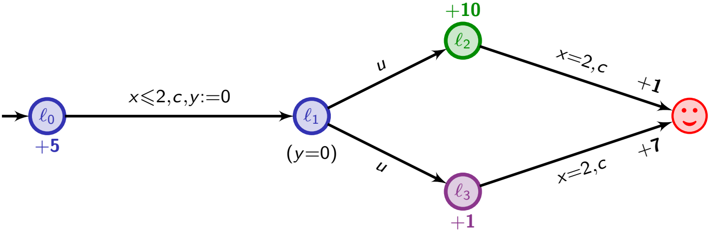


	ℓ_0	$\xrightarrow{1.3}$	ℓ_0	$\xrightarrow{c}$	ℓ_1	$\xrightarrow{u}$	ℓ_3	$\xrightarrow{0.7}$	ℓ_3	$\xrightarrow{c}$	☺
x	0		1.3		1.3		1.3		2		
y	0		1.3		0		0		0.7		

cost : 6.5 + 0

[ALP01] Alur, La Torre, Pappas. Optimal paths in weighted timed automata *(HSCC'01)*.
[BFH+01] Behrmann, Fehnker, Hune, Larsen, Pettersson, Romijn, Vaandrager. Minimum-cost reachability in priced timed automata *(HSCC'01)*.

Weighted/priced timed automata [ALP01,BFH+01]

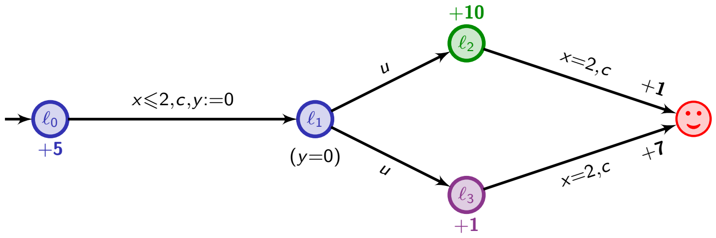


	ℓ_0	$\xrightarrow{1.3}$	ℓ_0	$\xrightarrow{c}$	ℓ_1	$\xrightarrow{u}$	ℓ_3	$\xrightarrow{0.7}$	ℓ_3	$\xrightarrow{c}$	☺
x	0		1.3		1.3		1.3		2		
y	0		1.3		0		0		0.7		

cost : 6.5 + 0 + 0

[ALP01] Alur, La Torre, Pappas. Optimal paths in weighted timed automata *(HSCC'01)*.
[BFH+01] Behrmann, Fehnker, Hune, Larsen, Pettersson, Romijn, Vaandrager. Minimum-cost reachability in priced timed automata *(HSCC'01)*.

Weighted/priced timed automata [ALP01,BFH+01]

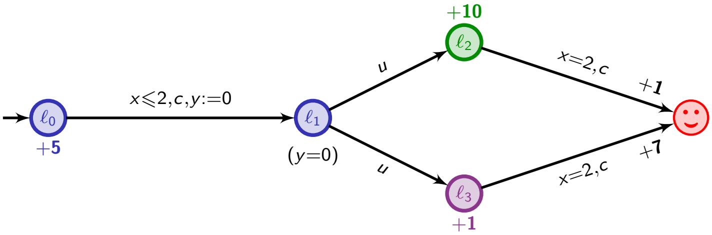


	ℓ_0	$\xrightarrow{1.3}$	ℓ_0	$\xrightarrow{c}$	ℓ_1	$\xrightarrow{u}$	ℓ_3	$\xrightarrow{0.7}$	ℓ_3	$\xrightarrow{c}$	☺
x	0		1.3		1.3		1.3		2		
y	0		1.3		0		0		0.7		

cost : 6.5 + 0 + 0 + 0.7

[ALP01] Alur, La Torre, Pappas. Optimal paths in weighted timed automata *(HSCC'01)*.
[BFH+01] Behrmann, Fehnker, Hune, Larsen, Pettersson, Romijn, Vaandrager. Minimum-cost reachability in priced timed automata *(HSCC'01)*.

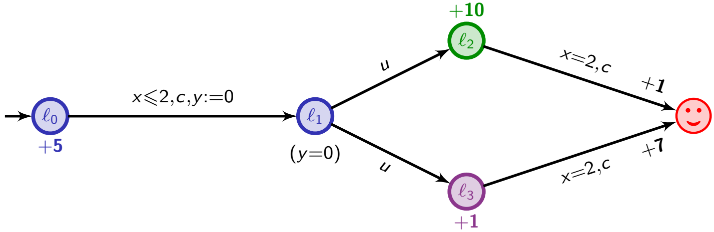
Weighted/priced timed automata [ALP01,BFH+01]



	ℓ_0	$\xrightarrow{1.3}$	ℓ_0	$\xrightarrow{c}$	ℓ_1	$\xrightarrow{u}$	ℓ_3	$\xrightarrow{0.7}$	ℓ_3	$\xrightarrow{c}$	☺
x	0		1.3		1.3		1.3		2		
y	0		1.3		0		0		0.7		
cost :		6.5	+	0	+	0	+	0.7	+	7	

[ALP01] Alur, La Torre, Pappas. Optimal paths in weighted timed automata *(HSCC'01)*.
[BFH+01] Behrmann, Fehnker, Hune, Larsen, Pettersson, Romijn, Vaandrager. Minimum-cost reachability in priced timed automata *(HSCC'01)*.

Weighted/priced timed automata [ALP01,BFH+01]

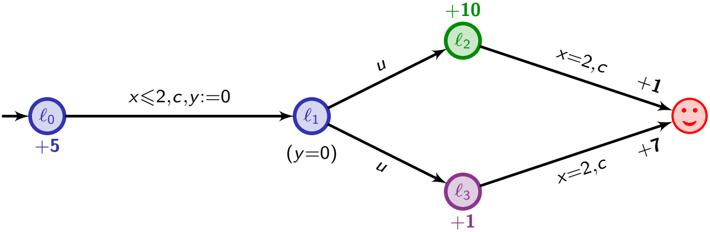


	ℓ_0	$\xrightarrow{1.3}$	ℓ_0	$\xrightarrow{c}$	ℓ_1	$\xrightarrow{u}$	ℓ_3	$\xrightarrow{0.7}$	ℓ_3	$\xrightarrow{c}$	☺
x	0		1.3		1.3		1.3		2		
y	0		1.3		0		0		0.7		

cost : $6.5 + 0 + 0 + 0.7 + 7 = 14.2$

[ALP01] Alur, La Torre, Pappas. Optimal paths in weighted timed automata *(HSCC'01)*.
[BFH+01] Behrmann, Fehnker, Hune, Larsen, Pettersson, Romijn, Vaandrager. Minimum-cost reachability in priced timed automata *(HSCC'01)*.

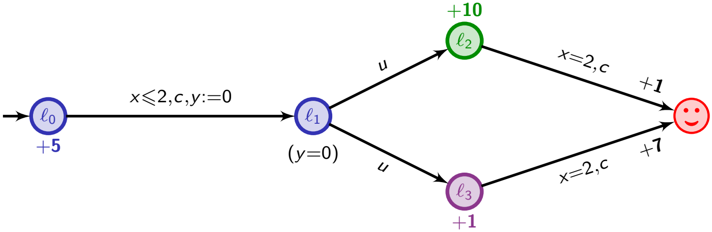
Weighted/priced timed automata [ALP01,BFH+01]



Question: what is the optimal cost for reaching 🙂 ?

[ALP01] Alur, La Torre, Pappas. Optimal paths in weighted timed automata *(HSCC'01)*.
[BFH+01] Behrmann, Fehnker, Hune, Larsen, Pettersson, Romijn, Vaandrager. Minimum-cost reachability in priced timed automata *(HSCC'01)*.

Weighted/priced timed automata [ALP01,BFH+01]

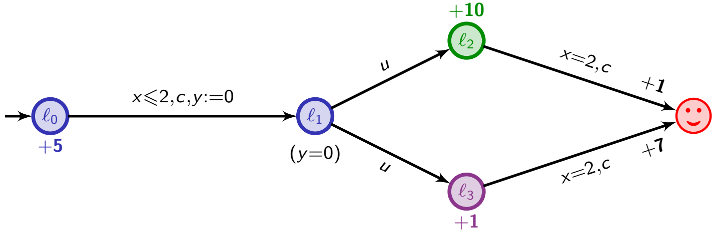


Question: what is the optimal cost for reaching 🙂 ?

$$5t + 10(2 - t) + 1$$

[ALP01] Alur, La Torre, Pappas. Optimal paths in weighted timed automata *(HSCC'01)*.
[BFH+01] Behrmann, Fehnker, Hune, Larsen, Pettersson, Romijn, Vaandrager. Minimum-cost reachability in priced timed automata *(HSCC'01)*.

Weighted/priced timed automata [ALP01,BFH+01]

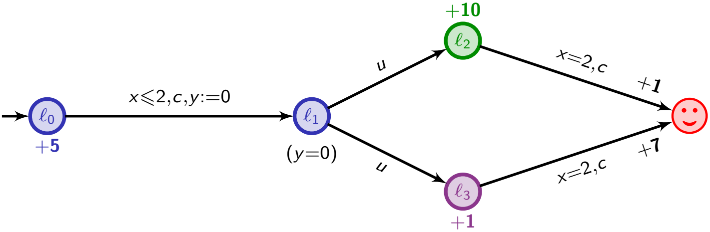


Question: what is the optimal cost for reaching 🙂 ?

$$5t + 10(2 - t) + 1 \;,\; 5t + (2 - t) + 7$$

[ALP01] Alur, La Torre, Pappas. Optimal paths in weighted timed automata *(HSCC'01)*.
[BFH+01] Behrmann, Fehnker, Hune, Larsen, Pettersson, Romijn, Vaandrager. Minimum-cost reachability in priced timed automata *(HSCC'01)*.

Weighted/priced timed automata [ALP01,BFH+01]

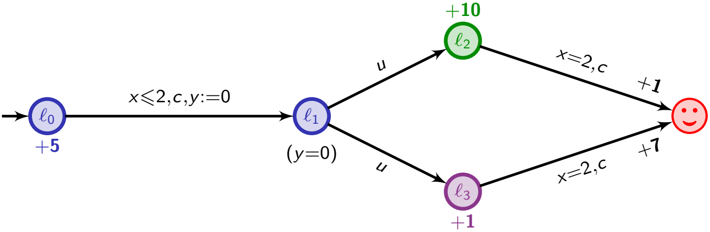


Question: what is the optimal cost for reaching ?

$$\min\left(\ 5t+10(2-t)+1\ ,\ 5t+(2-t)+7\ \right)$$

[ALP01] Alur, La Torre, Pappas. Optimal paths in weighted timed automata *(HSCC'01)*.
[BFH+01] Behrmann, Fehnker, Hune, Larsen, Pettersson, Romijn, Vaandrager. Minimum-cost reachability in priced timed automata *(HSCC'01)*.

Weighted/priced timed automata [ALP01,BFH+01]

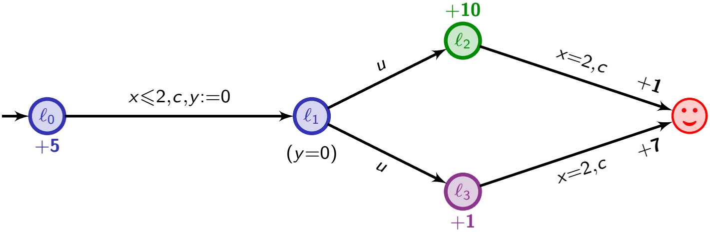


Question: what is the optimal cost for reaching ☺?

$$\inf_{0\leqslant t\leqslant 2} \min\left(5t+10(2-t)+1\ ,\ 5t+(2-t)+7 \right) = 9$$

[ALP01] Alur, La Torre, Pappas. Optimal paths in weighted timed automata *(HSCC'01)*.
[BFH+01] Behrmann, Fehnker, Hune, Larsen, Pettersson, Romijn, Vaandrager. Minimum-cost reachability in priced timed automata *(HSCC'01)*.

Weighted/priced timed automata [ALP01,BFH+01]



Question: what is the optimal cost for reaching 🙂 ?

$$\inf_{0 \leqslant t \leqslant 2} \min \left(5t + 10(2-t) + 1 \, , \, 5t + (2-t) + 7 \right) = 9$$

⇝ *strategy:* leave immediately ℓ_0, go to ℓ_3, and wait there 2 t.u.

[ALP01] Alur, La Torre, Pappas. Optimal paths in weighted timed automata *(HSCC'01)*.
[BFH+01] Behrmann, Fehnker, Hune, Larsen, Pettersson, Romijn, Vaandrager. Minimum-cost reachability in priced timed automata *(HSCC'01)*.