"Most of" leads to undecidability
Failure of adding frequencies to LTL
FoSSaCS 2021

Bartosz Bednarczyk, Jakub Michaliszyn

TU Dresden & University of Wrocław
What’s the formal verification about?

system:

property:

model-checking algorithm

Always(safe)

yes/no

"Most of" leads to undecidability
Linear-time Temporal Logic (LTL)

atomic propositions: , , ...

boolean combinators: ¬ ', ' _  , ' ^  , ...

temporal modalities: X ' ' ' ' ' "next " ' U ' ' ' ' ' "until " ' ' ' ' ' "eventually " ' ' ' ' ' "always "

A Kripke structure satisfies ' 2 LTL if all its infinite executions do:

M|= () 8 ⇡ 2 TM. ⇡ |= "true " U ' ' ' ' ' "always " "eventually ".
Linear-time Temporal Logic (LTL)

- atomic propositions: \( \odot, \odot, \ldots \)
Linear-time Temporal Logic (LTL)

- atomic propositions: \( \bigcirc, \bigcirc, \ldots \)

- boolean combinators: \( \neg \phi, \phi \lor \psi, \phi \land \psi, \ldots \)
Linear-time Temporal Logic (LTL)

- atomic propositions: \( \bigcirc, \bigcirc, \ldots \)

- boolean combinators: \( \neg \varphi, \varphi \lor \psi, \varphi \land \psi, \ldots \)

- temporal modalities:

  \[
  \begin{align*}
  X \varphi & \quad \begin{array}{c}
  \text{next } \varphi
  \end{array} \\
  \varphi U \psi & \quad \begin{array}{c}
  \varphi \text{ until } \psi
  \end{array}
  \end{align*}
  \]
Linear-time Temporal Logic (LTL)

- atomic propositions: $\circ, \circ, \ldots$

- boolean combinators: $\neg \varphi$, $\varphi \lor \psi$, $\varphi \land \psi$, ...

- temporal modalities:
  - $X \varphi$  
  - $\varphi U \psi$  
  - $\text{true } U \varphi \equiv F \varphi$  
  - $\neg F \neg \varphi \equiv G \varphi$  

A Kripke structure satisfies $\mathcal{L}_\text{TL}$ if all its infinite executions do:

$M \models \varphi$ if $\forall \pi \in \mathcal{T}_M. \pi \models \varphi$.
Satisfiability and model checking

Two main algorithmic problems

Satisfiability:
Input: a formula \( \phi \) in LTL;
Output: yes if there exists a Kripke structure \( M \) such that \( M \models \phi \); no otherwise.

Model checking:
Input: a formula \( \phi \) in LTL, and a Kripke structure \( M \);
Output: yes if \( M \models \phi \); no otherwise.
Satisfiability and model checking

Two main algorithmic problems

- **Satisfiability:**
  - Input: a formula $\varphi$ in LTL;
  - Output:
    - **yes** if there exists a Kripke structure $M$ s.t. $M \models \varphi$;
    - **no** otherwise.

- Model checking:
  - Input: a formula $\varphi$ in LTL, and a Kripke structure $M$;
  - Output: **yes** if $M \models \varphi$;
  - **no** otherwise.
Satisfiability and model checking

Two main algorithmic problems

- **Satisfiability:**
  - Input: a formula $\varphi$ in LTL;
  - Output:
    - *yes* if there exists a Kripke structure $M$ s.t. $M \models \varphi$;
    - *no* otherwise.

- **Model checking:**
  - Input: a formula $\varphi$ in LTL, and a Kripke structure $M$;
  - Output:
    - *yes* if $M \models \varphi$;
    - *no* otherwise.
LTL: Ups and downs

- Theorem (LTL is PSpace-complete.)
  - Model checking and satisfiability are logspace interreducible.
  - PSpace upper bound = on-the-fly construction of Buchi automata

LTL is useful in verification and has good algorithmic properties but it can't express quantitative properties!
LTL: Ups and downs

**Theorem** (LTL is \textit{PSPACE}-complete.)
**Theorem** *(LTL is PSPACE-complete.)*

- Model checking and satisfiability are logspace interreducible.
LTL: Ups and downs

**Theorem** (LTL is $\text{PSPACE}$-complete.)

- Model checking and satisfiability are logspace interreducible.
- PSpace upper bound = on-the-fly construction of Buchi automata
LTL: Ups and downs

**Theorem** (LTL is \textsf{PSPACE}-complete.)

- Model checking and satisfiability are logspace interreducible.
- PSpace upper bound = on-the-fly construction of Buchi automata

LTL is useful in verification and has good algorithmic properties
Theorem (LTL is $\text{PSpace}$-complete.)

- Model checking and satisfiability are logspace interreducible.
- PSpace upper bound = on-the-fly construction of Buchi automata

LTL is useful in verification and has good algorithmic properties

So what's wrong with it?
LTL: Ups and downs

Theorem (LTL is $\text{PSPACE}$-complete.)

- Model checking and satisfiability are logspace interreducible.
- PSpace upper bound = on-the-fly construction of Buchi automata

LTL is useful in verification and has good algorithmic properties

So what’s wrong with it?

but it can’t express quantitative properties!
Our goal: Extend LTL with frequency constraints

- Frequency LTL [Bollig et al. 12]: Extension of LTL with frequency until.
- Averaging in LTL [Bouyer et al. 14]: weighted alphabet, until calculating avg.
- Discounted-LTL [Almagor et al. 14]: long paths makes formulae less true.
- Metric LTL [Koymans’90]: time modelled as a real line
- ULTL [$F,P,X,Y$] with Presburger Arithmetics [Lodaya and Sreejith 17].
- Availability expressions [Hoenicke et al. 2010]: “Kleene, Rabin, and Scott are available.”

All of them are undecidable!

And the problem seems to be the until operator.
Our goal: Extend LTL with frequency constraints

Related works:

- Frequency LTL [Bollig et al. 12]: Extension of LTL with frequency until.
- Averaging in LTL [Bouyer et al. 14]: weighted alphabet, until calculating avg.
- Discounted-LTL [Almagor et al. 14]: long paths makes formulae less true.
- Metric LTL [Koymans'90]: time modelled as a real line
- ULTL $[F,P, X, Y]$ with Presburger Arithmetics [Lodaya and Sreejith 17].
- Availability expressions [Hoenicke et al. 2010]: “Kleene, Rabin, and Scott are available”

All of them are undecidable! And the problem seems to be the until operator.
Our goal: Extend LTL with frequency constraints

Related works:

- Frequency LTL [Bollig et al. 12]: Extension of LTL with frequency until.
- Averaging in LTL [Bouyer et al. 14]: weighted alphabet, until calculating avg.
- Discounted-LTL [Almagor et al. 14]: long paths makes formulae less true.
- Metric LTL [Koymans'90]: time modelled as a real line.
- ULTL [F, P, X, Y] with Presburger Arithmetics [Lodaya and Sreejith 17].
- Availability expressions [Hoenicke et al. 2010]: "Kleene, Rabin, and Scott are available".

All of them are undecidable! And the problem seems to be the until operator.
Our goal: Extend LTL with frequency constraints

Related works:

- Frequency LTL [Bollig et al. 12]: Extension of LTL with frequency until.
- Averaging in LTL [Bouyer et al. 14]: weighted alphabet, until calculating avg.
Our goal: Extend LTL with frequency constraints

Related works:
- Frequency LTL [Bollig et al. 12]: Extension of LTL with frequency until.
- Averaging in LTL [Bouyer et al. 14]: weighted alphabet, until calculating avg.
- Discounted-LTL [Almagor et al. 14] = long paths makes formulae less true.
Our goal: Extend LTL with frequency constraints

Related works:

• Frequency LTL [Bollig et al. 12]: Extension of LTL with frequency until.
• Averaging in LTL [Bouyer et al. 14]: weighted alphabet, until calculating avg.
• Discounted-LTL [Almagor et al. 14] = long paths makes formulae less true.
• Metric LTL [Koymans’90] - time modelled as a real line
Our goal: Extend LTL with frequency constraints

Related works:

- Frequency LTL [Bollig et al. 12]: Extension of LTL with frequency until.
- Averaging in LTL [Bouyer et al. 14]: weighted alphabet, until calculating avg.
- Discounted-LTL [Almagor et al. 14] = long paths makes formulae less true.
- Metric LTL [Koymans’90] - time modelled as a real line
- ULTL[F,P, X, Y] with Presburger Arithmetics [Lodaya and Sreejith 17].
Our goal: Extend LTL with frequency constraints

Related works:

- Frequency LTL [Bollig et al. 12]: Extension of LTL with frequency until.
- Averaging in LTL [Bouyer et al. 14]: weighted alphabet, until calculating avg.
- Discounted-LTL [Almagor et al. 14] = long paths makes formulae less true.
- Metric LTL [Koymans’90] - time modelled as a real line
- ULTL[F,P, X, Y] with Presburger Arithmetics [Lodaya and Sreejith 17].
- Availability expressions [Hoenicke et al. 2010]:
Our goal: Extend LTL with frequency constraints

Related works:

- Frequency LTL [Bollig et al. 12]: Extension of LTL with frequency until.
- Averaging in LTL [Bouyer et al. 14]: weighted alphabet, until calculating avg.
- Discounted-LTL [Almagor et al. 14] = long paths makes formulae less true.
- Metric LTL [Koymans’90] - time modelled as a real line
- ULTL[F,P, X, Y] with Presburger Arithmetics [Lodaya and Sreejith 17].
- Availability expressions [Hoenicke et al. 2010]:
  "Kleene, Rabin, and Scott are available"
Our goal: Extend LTL with frequency constraints

Related works:

- Frequency LTL [Bollig et al. 12]: Extension of LTL with frequency until.
- Averaging in LTL [Bouyer et al. 14]: weighted alphabet, until calculating avg.
- Discounted-LTL [Almagor et al. 14] = long paths makes formulae less true.
- Metric LTL [Koymans’90] - time modelled as a real line
- ULTL[F,P, X, Y] with Presburger Arithmetics [Lodaya and Sreejith 17].
- Availability expressions [Hoenicke et al. 2010]:

11 / 18
Our goal: Extend LTL with frequency constraints

Related works:

- Frequency LTL [Bollig et al. 12]: Extension of LTL with frequency until.
- Averaging in LTL [Bouyer et al. 14]: weighted alphabet, until calculating avg.
- Discounted-LTL [Almagor et al. 14] = long paths makes formulae less true.
- Metric LTL [Koymans’90] - time modelled as a real line
- ULTL[F,P, X, Y] with Presburger Arithmetics [Lodaya and Sreejith 17].
- Availability expressions [Hoenicke et al. 2010]:

All of them are undecidable!
Our goal: Extend LTL with frequency constraints

Related works:

- Frequency LTL [Bollig et al. 12]: Extension of LTL with frequency until.
- Averaging in LTL [Bouyer et al. 14]: weighted alphabet, until calculating avg.
- Discounted-LTL [Almagor et al. 14] = long paths makes formulae less true.
- Metric LTL [Koymans’90] - time modelled as a real line
- ULTL[F,P, X, Y] with Presburger Arithmetics [Lodaya and Sreejith 17].
- Availability expressions [Hoenicke et al. 2010]:

All of them are undecidable!

And the problem seems to be the until operator.
Our setting

We allow only for "finally" operator + "most of the previous positions satisfies $\phi$" $PM(\phi)$ or "a is the most-frequent-letter in the past" $MFL(a)$.

$w_i = F(\phi) \text{ if } \exists j \geq i \text{ and } w_j = \phi \quad w_i = PM(\phi) \text{ if } \{j < i : w_j = \phi\} \geq i^2 \quad w_i = MFL(\sigma) \text{ if } \forall \tau \in AP, \{j < i : w_j = \sigma\} \geq \{j < i : w_j = \tau\}$
Our setting

We allow only for “finally” $F$ operator.
Our setting

We allow only for “finally” $F$ operator $+$

“most of the previous positions satisfies $\varphi$” $\text{PM}(\varphi)$
Our setting

We allow only for “finally” $F$ operator $+$

“most of the previous positions satisfies $\varphi$” $PM(\varphi)$

or “a is the most-frequent-letter in the past” $MFL(a)$
Our setting

We allow only for “finally” $F$ operator $+$

“most of the previous positions satisfies $\varphi$” $PM(\varphi)$

or “$a$ is the most-frequent-letter in the past” $MFL(a)$

$\forall \tau \in AP, |\{ j < i : w, j | = \tau \}| \geq |\{ j < i : w, j | = \sigma \}|$

$w, i | = F \varphi$ if $\exists j$ such that $|w| > j \geq i$ and $w, j | = \varphi$

$w, i | = PM(\varphi)$ if $|\{ j < i : w, j | = \varphi \}| \geq i$

$w, i | = MFL(\sigma)$ if $\forall \tau \in AP, |\{ j < i : w, j | = \tau \}| \geq |\{ j < i : w, j | = \sigma \}|$

"eventually $\varphi$"

"always $\varphi$"
Our setting

We allow only for “finally” \( F \) operator +

“most of the previous positions satisfies \( \varphi \)” \( \text{PM}(\varphi) \)

or “\( a \) is the most-frequent-letter in the past” \( \text{MFL}(a) \)

\[ \begin{align*}
\varphi & \rightarrow \text{MP} \varphi \rightarrow \text{MFL} \rightarrow \text{MFL} \rightarrow \ldots \\
\varphi & \rightarrow \varphi \rightarrow \varphi \rightarrow \varphi \rightarrow \varphi \rightarrow \ldots
\end{align*} \]

\( F \varphi \) “eventually \( \varphi \)”

\( \neg F \neg \varphi \equiv \text{G} \varphi \) “always \( \varphi \)”

\( \mathfrak{w}, i \models F \varphi \) if \( \exists j \) such that \( |\mathfrak{w}| > j \geq i \) and \( \mathfrak{w}, j \models \varphi \)
Our setting

We allow only for “finally” $F$ operator +

“most of the previous positions satisfies $\varphi$” $PM(\varphi)$
or “a is the most-frequent-letter in the past” $MFL(a)$

\[
\begin{align*}
F\varphi & \quad \text{“eventually } \varphi\text{”} \\
\neg F \neg \varphi & \equiv G\varphi \quad \text{“always } \varphi\text{”}
\end{align*}
\]

$w, i \models F\varphi$ if $\exists j$ such that $|w| > j \geq i$ and $w, j \models \varphi$

$w, i \models PM\varphi$ if $|\{j < i : w, j \models \varphi\}| \geq \frac{i}{2}$
Our setting

We allow only for “finally” $F$ operator +

“most of the previous positions satisfies $\varphi$” $PM(\varphi)$
or “$a$ is the most-frequent-letter in the past” $MFL(a)$

$w, i \models F \varphi$ if $\exists j$ such that $|w| > j \geq i$ and $w, j \models \varphi$

$w, i \models PM \varphi$ if $|\{j < i: w, j \models \varphi\}| \geq \frac{i}{2}$

$w, i \models MFL \sigma$ if $\forall \tau \in AP. |\{j < i: w, j \models \sigma\}| \geq |\{j < i: w, j \models \tau\}|$
Our results

• LTL with F and PM is undecidable.
• LTL with F and MFL is undecidable.
• Some rather uninteresting fragments of LTL + PM are decidable.
• FO[<] + Majority quantifier is undecidable.

Our proof technique

We focus on a single modality Half:\[\text{Half } \varphi := \text{PM}(\varphi) \land \text{PM}(\neg \varphi)\]

The proof goes via encoding of Minsky’s two counter machines

In the last few minutes we present the main ideas of the encoding.
Our results

- LTL with $F$ and $PM$ is undecidable.
Our results

- LTL with $F$ and $PM$ is undecidable.
- LTL with $F$ and $MFL$ is undecidable.
Our results

- LTL with $F$ and $PM$ is undecidable.
- LTL with $F$ and $MFL$ is undecidable.
- Some rather uninteresting fragments of LTL+$PM$ are decidable.
Our results

• LTL with F and PM is undecidable.
• LTL with F and MFL is undecidable.
• Some rather uninteresting fragments of LTL+PM are decidable.
• FO²[<] + Majority quantifier is undecidable.
Our results

- LTL with \( F \) and \( \text{PM} \) is undecidable.
- LTL with \( F \) and \( \text{MFL} \) is undecidable.
- Some rather uninteresting fragments of \( \text{LTL} + \text{PM} \) are decidable.
- \( \text{FO}^2[<] \) + Majority quantifier is undecidable.

Our proof technique
Our results

- LTL with $F$ and $PM$ is undecidable.
- LTL with $F$ and $MFL$ is undecidable.
- Some rather uninteresting fragments of LTL+PM are decidable.
- $FO^2[\prec] +$ Majority quantifier is undecidable.

Our proof technique

- We focus on a single modality $\text{Half}$:
Our results

- LTL with F and PM is undecidable.
- LTL with F and MFL is undecidable.
- Some rather uninteresting fragments of LTL+PM are decidable.
- FO²[<] + Majority quantifier is undecidable.

Our proof technique

- We focus on a single modality Half:

  \[ \omega, i \models \text{Half} \varphi \text{ if } |\{j < i : \omega, j \models \varphi\}| = \frac{i}{2} \]
Our results

- LTL with F and PM is undecidable.
- LTL with F and MFL is undecidable.
- Some rather uninteresting fragments of LTL+PM are decidable.
- FO\(^2[<]\) + Majority quantifier is undecidable.

Our proof technique

- We focus on a single modality Half:

\[ \omega, i \models \text{Half} \varphi \text{ if } |\{ j < i : \omega, j \models \varphi \}| = \frac{i}{2} \]

Half \( \varphi \) := PM(\( \varphi \)) \& PM(\( \neg \varphi \))
Our results

- LTL with $F$ and $PM$ is undecidable.
- LTL with $F$ and $MFL$ is undecidable.
- Some rather uninteresting fragments of $LTL+PM$ are decidable.
- $FO^2[<] +$ Majority quantifier is undecidable.

Our proof technique

- We focus on a single modality $Half$:

  $$\sigma, i \models \text{Half } \varphi \text{ if } |\{ j < i : \sigma, j \models \varphi \}| = \frac{i}{2}$$

  $Half \varphi := PM(\varphi) \land PM(\neg \varphi)$

- The proof goes via encoding of Minsky’s two counter machines
Our results

- LTL with F and PM is undecidable.
- LTL with F and MFL is undecidable.
- Some rather uninteresting fragments of LTL+PM are decidable.
- FO$^2[<]$ + Majority quantifier is undecidable.

Our proof technique

- We focus on a single modality \textbf{Half}:
  \[ \mathfrak{w}, i \models \text{Half} \varphi \text{ if } |\{j < i : \mathfrak{w}, j \models \varphi\}| = \frac{i}{2} \]
  \[ \text{Half } \varphi := \text{PM}(\varphi) \land \text{PM}(\neg \varphi) \]
- The proof goes via encoding of Minsky's two counter machines
  In the last few minutes we present the main ideas of the encoding.
Shadowy words

Consider an alphabet \{wht, shdw\}.

A word \(w\) is shadowy if it belongs to \((wht \cdot shdw) +\).

Lemma

Shadowy words are LTL-F, Half-definable.

Proof

It suffices to employ the following formulae:

• \(wht \cdot G (wht \leftrightarrow \neg shdw)\)
• \(G (wht \rightarrow F (shdw))\)
• \(G (\varphi_{even} \leftrightarrow wht)\), where \(\varphi_{even} := \text{Half} wht\)
**Shadowy words**

Consider an alphabet \( \{wht, shdw\} \).
Shadowy words

Consider an alphabet \( \{wht, shdw\} \).

A word \( w \) is shadowy if it belongs to \( (wht \cdot shdw)^+ \).
Shadowy words

Consider an alphabet \( \{ \text{wht}, \text{shdw} \} \).

A word \( \omega \) is shadowy if it belongs to \((\text{wht} \cdot \text{shdw})^+\).
Shadowy words

Consider an alphabet \( \{ \text{wht}, \text{shdw} \} \).

A word \( \omega \) is shadowy if it belongs to \( (\text{wht} \cdot \text{shdw})^+ \).

**Lemma**

Shadowy words are \( \text{LTL}_{F, \text{Half}} \)-definable.
Shadowy words

Consider an alphabet \( \{wht, shdw\} \).

A word \( w \) is shadowy if it belongs to \((wht \cdot shdw)^+\).

Lemma

Shadowy words are \( \text{LTL}_{F, \text{Half}} \)-definable.

Proof

It suffices to employ the following formulae:
Shadowy words

Consider an alphabet \( \{wht, shdw\} \).

A word \( w \) is shadowy if it belongs to \( (wht \cdot shdw)^+ \).

**Lemma**

Shadowy words are \( \text{LTL}_{F, \text{Half}} \)-definable.

**Proof**

It suffices to employ the following formulae:

- \( wht \)
Shadowy words

Consider an alphabet \{wht, shdw\}.

A word \(w\) is shadowy if it belongs to \((wht \cdot shdw)^+\).

**Lemma**

Shadowy words are \(\mathbb{LTL}_{F,\text{Half}}\)-definable.

**Proof**

It suffices to employ the following formulae:

- \(wht\)
- \(G (wht \leftrightarrow \neg shdw)\)
Shadowy words

Consider an alphabet \( \{ \text{wht}, \text{shdw} \} \).

A word \( \omega \) is shadowy if it belongs to \( (\text{wht} \cdot \text{shdw})^+ \).

**Lemma**

Shadowy words are \( \text{LTL}_{F,\text{Half}} \)-definable.

**Proof**

It suffices to employ the following formulae:

- \( \text{wht} \)
- \( \text{G} (\text{wht} \leftrightarrow \neg \text{shdw}) \)
- \( \text{G} (\text{wht} \rightarrow \text{F} (\text{shdw})) \)
Shadowy words

Consider an alphabet \( \{ wht, shdw \} \).

A word \( w \) is shadowy it belongs to \((wht \cdot shdw)^+\)

Lemma

Shadowy words are \( LTL_{F, \text{Half}} \)-definable.

Proof

It suffices to employ the following formulae:

- \( wht \)
- \( G(wht \leftrightarrow \neg shdw) \)
- \( G(wht \rightarrow F(\text{shdw})) \)
- \( G(\varphi_{\text{even}} \leftrightarrow wht) \)
**Shadowy words**

Consider an alphabet \( \{ \text{wht}, \text{shdw} \} \).

A word \( \omega \) is shadowy if it belongs to \( (\text{wht} \cdot \text{shdw})^+ \).

**Lemma**

Shadowy words are \( \text{LTL}_{F,\text{Half}} \)-definable.

**Proof**

It suffices to employ the following formulae:

- \( \text{wht} \)
- \( \text{G} (\text{wht} \leftrightarrow \neg \text{shdw}) \)
- \( \text{G} (\text{wht} \rightarrow \text{F} (\text{shdw})) \)
- \( \text{G} (\phi_{\text{even}} \leftrightarrow \text{wht}) \), where \( \phi_{\text{even}} := \text{Half} \text{ wht} \)
Transferring truth predicates

Proof
It suffices to express:
• $(\{ w : 0 \})$ : for the last white position $p$ we have:
  $w, p \models \sigma \iff w, p + 1 \models \tilde{\sigma}$.
• all white $p$s satisfy $(\spadesuit) : \# \text{wht} \land \sigma(w, p) = \# \text{shdw} \land \tilde{\sigma}(w, p)$.
Exercise 3.3. Let $\sigma$ and $\bar{\sigma}$ be distinct letters from $\text{AP} \setminus \{\text{wht, shdw}\}$. There is an $\text{LTL}_{F,\text{Half}}$ formula $\varphi^{\text{trans}}_{\sigma \sim \bar{\sigma}}$, such that $w \models \varphi^{\text{trans}}_{\sigma \sim \bar{\sigma}}$ iff:

1. $w$ is shadowy,
2. only white (resp., shadow) positions of $w$ can be labelled $\sigma$ (resp., $\bar{\sigma}$) and
3. for any even position $p$ we have: $w, p \models \sigma \iff w, p+1 \models \bar{\sigma}$.

\begin{center}
\begin{tikzpicture}[node distance = 1cm,>=latex]
  \tikzstyle{every node}=[circle,draw,fill=black!20]
  \node [label=below:$\text{wht} \sigma$] (1) ;
  \node [label=below:$\text{shdw} \bar{\sigma}$, right of=1] (2) ;
  \node [label=below:$\text{wht} \bar{\sigma}$, right of=2] (3) ;
  \node [label=below:$\text{shdw} \bar{\sigma}$, right of=3] (4) ;
  \node [label=below:$\text{wht} \bar{\sigma}$, right of=4] (5) ;
  \node [label=below:$\text{shdw} \bar{\sigma}$, right of=5] (6) ;
  \draw (1) edge (2)
        (2) edge (3)
        (3) edge (4)
        (4) edge (5)
        (5) edge (6);
\end{tikzpicture}
\end{center}
Exercise 3.3. Let $\sigma$ and $\bar{\sigma}$ be distinct letters from $\text{AP} \setminus \{\text{wht}, \text{shdw}\}$. There is an $\text{LTL}_{F,\text{Half}}$ formula $\varphi^{\text{trans}}_{\sigma \sim \bar{\sigma}}$, such that $w \models \varphi^{\text{trans}}_{\sigma \sim \bar{\sigma}}$ iff:

1. $w$ is shadowy,

2. only white (resp., shadow) positions of $w$ can be labelled $\sigma$ (resp., $\bar{\sigma}$) and

3. for any even position $p$ we have: $w, p \models \sigma \iff w, p+1 \models \bar{\sigma}$.

![Diagram showing the sequence of white and shadow positions]

Lemma

Transfer formulae $\text{LTL}_{F,\text{Half}}$-definable.
Exercise 3.3. Let $\sigma$ and $\bar{\sigma}$ be distinct letters from $\text{AP} \setminus \{\text{wht}, \text{shdw}\}$. There is an $\text{LTL}_{F,\text{Half}}$ formula $\varphi_{\sigma \sim \bar{\sigma}}^{\text{trans}}$, such that $w \models \varphi_{\sigma \sim \bar{\sigma}}^{\text{trans}}$ iff:

1. $w$ is shadowy,

2. only white (resp., shadow) positions of $w$ can be labelled $\sigma$ (resp., $\bar{\sigma}$) and

3. for any even position $p$ we have: $w, p \models \sigma \iff w, p+1 \models \bar{\sigma}$.

Lemma

Transfer formulae $\text{LTL}_{F,\text{Half}}$-definable.

Proof

It suffices to express:
Exercise 3.3. Let \( \sigma \) and \( \tilde{\sigma} \) be distinct letters from \( \text{AP} \setminus \{\text{wht}, \text{shdw}\} \). There is an \( \text{LTL}_{F,\text{Half}} \) formula \( \varphi^{\text{trans}}_{\sigma \sim \tilde{\sigma}} \), such that \( \mathfrak{w} \models \varphi^{\text{trans}}_{\sigma \sim \tilde{\sigma}} \) iff:

1. \( \mathfrak{w} \) is shadowy,
2. only white (resp., shadow) positions of \( \mathfrak{w} \) can be labelled \( \sigma \) (resp., \( \tilde{\sigma} \)) and
3. for any even position \( p \) we have: \( \mathfrak{w}, p \models \sigma \Leftrightarrow \mathfrak{w}, p+1 \models \tilde{\sigma} \).

\[
\begin{array}{cccccc}
\text{wht} & \rightarrow & \text{shdw} & \rightarrow & \text{wht} & \rightarrow & \text{shdw} \\
\sigma & \rightarrow & \tilde{\sigma} & \rightarrow & \sigma & \rightarrow & \tilde{\sigma}
\end{array}
\]

Lemma

Transfer formulae \( \text{LTL}_{F,\text{Half}} \)-definable.

Proof

It suffices to express:

- \((\Diamond)\) : for the last white position \( p \) we have: \( \mathfrak{w}, p \models \sigma \Leftrightarrow \mathfrak{w}, p+1 \models \tilde{\sigma} \).
Exercise 3.3. Let $\sigma$ and $\tilde{\sigma}$ be distinct letters from $\text{AP} \setminus \{\text{wht, shdw}\}$. There is an $\text{LTL}_{F,\text{Half}}$ formula $\varphi_{\sigma \sim \tilde{\sigma}}^{\text{trans}}$, such that $w \models \varphi_{\sigma \sim \tilde{\sigma}}^{\text{trans}}$ iff:

1. $w$ is shadowy,

2. only white (resp., shadow) positions of $w$ can be labelled $\sigma$ (resp., $\tilde{\sigma}$) and

3. for any even position $p$ we have: $w, p \models \sigma \iff w, p+1 \models \tilde{\sigma}$.

Lemma

Transfer formulae $\text{LTL}_{F,\text{Half}}$-definable.

Proof

It suffices to express:

- $(\diamondsuit)$: for the last white position $p$ we have: $w, p \models \sigma \iff w, p+1 \models \tilde{\sigma}$.

- all white $p$ satisfy $(\heartsuit)$: $\#_{\text{wht} \land \sigma}^<(w, p) = \#_{\text{shdw} \land \tilde{\sigma}}^<(w, p)$
• ($\Diamond$): for the last white position $p$ we have: $\mathfrak{w}, p \models \sigma \iff \mathfrak{w}, p+1 \models \tilde{\sigma}$. 
(◇) : for the last white position $p$ we have: $w, p \models \sigma \iff w, p + 1 \models \tilde{\sigma}$.

Last position sees only shadows!
• (⋄) : for the last white position \( p \) we have: \( \mathfrak{w}, p \models \sigma \iff \mathfrak{w}, p+1 \models \tilde{\sigma} \).

Last position sees only shadows! \( \varphi_{last} := \mathbf{G}(shdw) \)
• (◇) : for the last white position $p$ we have: $\mathbf{w}, p \models \sigma \iff \mathbf{w}, p+1 \models \tilde{\sigma}$.

Last position sees only shadows! $\varphi_{\text{last}} := \mathbf{G}(\text{shdw})$

Second to last position is white:
(◊) : for the last white position \( p \) we have: \( w, p \models \sigma \iff w, p+1 \models \tilde{\sigma} \).

Last position sees only shadows! \( \varphi_{\text{last}} := G(\text{shdw}) \)

Second to last position is white: \( \text{wht} \ldots \)
• (◊) : for the last white position $p$ we have: $w, p \models \sigma \iff w, p+1 \models \tilde{\sigma}$.

Last position sees only shadows! $\varphi_{\text{last}} := G(\text{shdw})$

Second to last position is white: $wht \ldots$

and sees only the last shadows
(◊) : for the last white position $p$ we have: $w, p \models \sigma \iff w, p+1 \models \tilde{\sigma}$.

Last position sees only shadows! $\varphi_{\text{last}} := G(shdw)$

Second to last position is white: $wht \ldots$

and sees only the last shadows $G(shdw \rightarrow \varphi_{\text{last}})$
• (◊) : for the last white position \( p \) we have: \( \mathbf{w}, p \models \sigma \iff \mathbf{w}, p+1 \models \tilde{\sigma} \).

  Last position sees only shadows! \( \varphi_{\text{last}} := \mathbf{G}(\text{shdw}) \)

  Second to last position is white: \( \text{wht} \ldots \)

  and sees only the last shadows \( \mathbf{G}(\text{shdw} \to \varphi_{\text{last}}) \)

  Hence, take \( \varphi_{\text{sec-to-last}} := \text{wht} \land \mathbf{G}(\text{shdw} \to \varphi_{\text{last}}) \)
• (◇): for the last white position $p$ we have: $w, p \models \sigma \iff w, p+1 \models \tilde{\sigma}$.

Last position sees only shadows!  $\varphi_{\text{last}} := G \left( shdw \right)$

Second to last position is white:  $wht \ldots$

and sees only the last shadows  $G \left( shdw \rightarrow \varphi_{\text{last}} \right)$

Hence, take $\varphi_{\text{sec-to-last}} := wht \land G \left( shdw \rightarrow \varphi_{\text{last}} \right)$

and the formula $F \left( \varphi_{\text{sec-to-last}} \land \pm \sigma \right) \land F \left( \varphi_{\text{last}} \land \pm \tilde{\sigma} \right)$ does the job!
all white $p$ satisfy (♥) : $\#^{\text{wht} \land \sigma}(w, p) = \#^{\text{shd} \land \overline{\sigma}}(w, p)$
• all white $p$ satisfy ($\heartsuit$):  $\#_{\text{wht} \land \sigma}(w, p) = \#_{\text{shdw} \land \bar{\sigma}}(w, p)$

$$\#_{\text{wht} \land \sigma}(w, p) = \#_{\text{shdw} \land \bar{\sigma}}(w, p)$$
• all white \( p \) satisfy (\( \diamond \)) : \( \#_{\text{wht} \land \sigma} (w, p) = \#_{\text{shdw} \land \tilde{\sigma}} (w, p) \)

\[
\#_{\text{wht} \land \sigma} (w, p) = \#_{\text{shdw} \land \tilde{\sigma}} (w, p)
\]

\[
\#_{\text{wht} \land \sigma} (w, p) - \#_{\text{shdw} \land \tilde{\sigma}} (w, p) = 0
\]
• all white $p$ satisfy (♥) : \[ \#_{\text{wht} \land \sigma}(w, p) = \#_{\text{shdw} \land \bar{\sigma}}(w, p) \]

\[ \#_{\text{wht} \land \sigma}(w, p) - \#_{\text{shdw} \land \bar{\sigma}}(w, p) = 0 \]

\[ \#_{\text{wht} \land \sigma}(w, p) + \frac{p}{2} - \#_{\text{shdw} \land \bar{\sigma}}(w, p) = \frac{p}{2} = \text{"Half"} \]
• all white $p$ satisfy $(\heartsuit)$: $\#_{wht \land \sigma}(w, p) = \#_{shdw \land \bar{\sigma}}(w, p)$

$$\#_{wht \land \sigma}(w, p) = \#_{shdw \land \bar{\sigma}}(w, p)$$

$$\#_{wht \land \sigma}(w, p) - \#_{shdw \land \bar{\sigma}}(w, p) = 0$$

$$\#_{wht \land \sigma}(w, p) + \frac{p}{2} - \#_{shdw \land \bar{\sigma}}(w, p) = \frac{p}{2} = "Half"$$

$$\#_{wht \land \sigma}(w, p) + \#_{shdw}(w, p) - \#_{shdw \land \bar{\sigma}}(w, p) = "Half"$$
all white $p$ satisfy (♥):

$$\#_{wht \land \sigma}(w, p) = \#_{shdw \land \bar{\sigma}}(w, p)$$

$$\#_{wht \land \sigma}(w, p) - \#_{shdw \land \bar{\sigma}}(w, p) = 0$$

$$\#_{wht \land \sigma}(w, p) + \frac{p}{2} - \#_{shdw \land \bar{\sigma}}(w, p) = \frac{p}{2} = "Half"$$

$$\#_{wht \land \sigma}(w, p) + \#_{shdw}(w, p) - \#_{shdw \land \bar{\sigma}}(w, p) = "Half"$$

$$\#_{wht \land \sigma}(w, p) + \#_{shdw \land \bar{\sigma}}(w, p) = "Half"$$
• all white $p$ satisfy ($\heartsuit$): $\#^{\text{wht} \land \sigma}(w, p) = \#^{\text{shdw} \land \bar{\sigma}}(w, p)$

$$\#^{\text{wht} \land \sigma}(w, p) = \#^{\text{shdw} \land \bar{\sigma}}(w, p)$$

$$\#^{\text{wht} \land \sigma}(w, p) - \#^{\text{shdw} \land \bar{\sigma}}(w, p) = 0$$

$$\#^{\text{wht} \land \sigma}(w, p) + \frac{p}{2} - \#^{\text{shdw} \land \bar{\sigma}}(w, p) = \frac{p}{2} = \text{"Half"}$$

$$\#^{\text{wht} \land \sigma}(w, p) + \#^{\text{shdw}}(w, p) - \#^{\text{shdw} \land \bar{\sigma}}(w, p) = \text{"Half"}$$

$$\#^{\text{wht} \land \sigma}(w, p) + \#^{\text{shdw} \land \neg \bar{\sigma}}(w, p) = \text{"Half"}$$

and hence we get a formula $\text{Half}([\text{wht} \land \sigma] \lor [\text{shdw} \land \neg \bar{\sigma}])$
Our results

- LTL with $F$ and $PM$ is undecidable.
- LTL with $F$ and $MFL$ is undecidable.
- Some rather uninteresting fragments of $LTL + PM$ are decidable.
- $FO^2[<] +$ Majority quantifier is undecidable.

Our proof technique

- We focus on a single modality $Half$:

  \[ w, i \models Half \varphi \text{ if } |\{j < i : w, j \models \varphi\}| = \frac{i}{2} \]

  \[ Half \varphi := PM(\varphi) \land PM(\neg \varphi) \]

- The proof goes via encoding of Minsky’s two counter machines
- We use shadowy words and tricks with $+\frac{P}{2}$ to express equicardinality

Thanks for attention!

Some initial LTL slides by ©Nicolas Markey.