Welcome

LTLf: LTL on finite traces

LTLf is a variant of LTL. Its semantics is restricted to finite traces, as opposed to infinite traces (ω-words), and it adds a "weak next" operator (Xw) that, unlike the "next" operator (X), is satisfied when no next state exists.

The purpose of this survey is to illustrate differences between LTLf and LTL.

This survey was designed by researchers at Brown University, the University of Oxford, and the Free University of Bozen--Bolzano. Be advised that your anonymized responses may appear in a public dataset. For more information, contact benjamin.l.greenman@gmail.com

This survey has 5 parts:

1. Explain mismatched traces (5 free response q's)

2. Match traces and formulas (6 yes/no q's)
3. Describe formulas in English (4 free response q's)
4. Translate English to formulas (5 free response q's)
5. Check LTLf equations (4 yes/no q's)

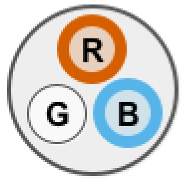
The formulas use five temporal operators:

- `G(a)`: (always) `a` holds for every remaining state in the trace
- `F(a)`: (eventually) `a` holds for some state in the trace
- `a U b`: (strong until) `b` holds for some state in the trace, and `a` holds for every state between that state and the current state
- `X(a)`: (next) there must be a next state and `a` must hold in it
- `Xw(a)`: (weak next) if there is a next state, then `a` must hold in it

The formulas also use four propositional operators:

- `&` (and)
- `||` (or)
- `=>` (implies)
- `!` (not)

As a concrete domain, the questions ask about the state of an instrument panel over time. The panel has three colors: Red, Green, and Blue. For example, the picture below shows a panel with Red on, Green off, and Blue on:



Background Questions

Have you taken a course on formal methods or verification?*

- ○ Yes
- ○ No
- ○ [ ] Other (explain below)

Is there anything we should know about your prior coursework?

Do you have experience with LTL?*

- Yes
- No
- Other (explain below)

Briefly describe your experience with LTL.*

Reject Traces

Part 1/5: Reject Traces

The following questions present an LTLf formula and a finite trace that **does not satisfy** the formula. Your task is to decide why the trace fails to satisfy the formula.

Example reasons:

- Only an infinite trace can satisfy the formula
- Trace too long, the formula accepts no traces of its length.
- Trace too short, the formula accepts no traces of its length.
- Trace content mismatch, wrong lights on/off in some state.

For reference, the formulas use five temporal operators:

- `G(a)`: (always) `a` holds for every remaining state in the trace
- `F(a)`: (eventually) `a` holds for some state in the trace
- `a U b`: (strong until) `b` holds for some state in the trace, and `a` holds for every state between that state and the current state
- `X(a)`: (next) there must be a next state and `a` must hold in it

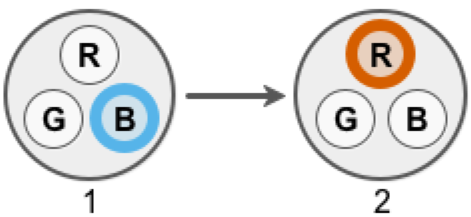
- `Xw(a)`: (weak next) if there is a next state, then `a` must hold in it

We first give two **Examples** to illustrate the questions and the style of answers that we are expecting.

Example Question: Why does the formula

`X(X(X(Blue)))`

reject this trace?



Example Answer:

- The trace is too short. No traces of length 2 can satisfy this formula with three X's.

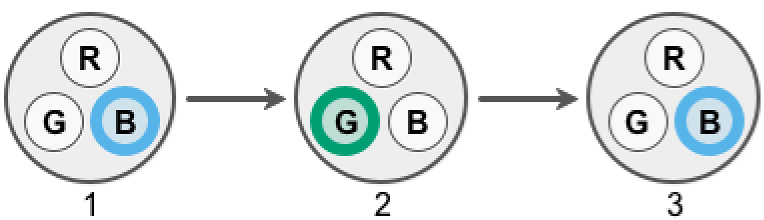
Do the **Example Question** and **Example Answer** make sense to you?*

- ◯ Yes
- ◯ [ ] No (please explain)

Example Question: Why does the formula

`F(Red)`

reject this trace?



Example Answer:

- Content mismatch: the trace has no Red states, but the formula requires at least one with Red on.

Do the **Example Question** and **Example Answer** make sense to you?*

- ◯ Yes

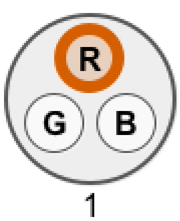
○ [ ] No (please explain)

The actual task begins now.

Q. Why does the formula

```
X(G(Red))
```

reject this trace?*

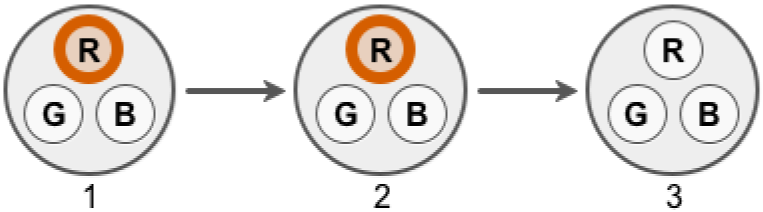


[ ]

Q. Why does the formula

```
Red U (!Red & F(Blue))
```

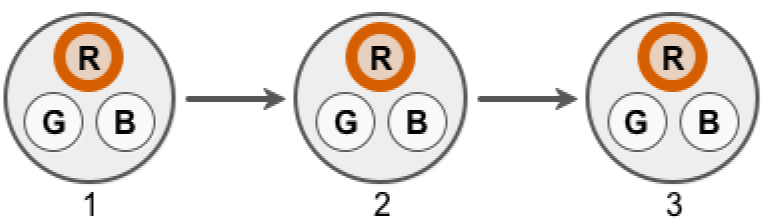
reject this trace?*



Q. Why does the formula

`G(Red) & Xw(Xw(!Red))`

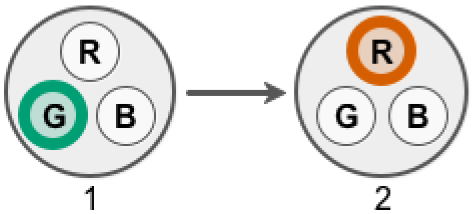
reject this trace?*



Q. Why does the formula

```
F(G(!Red))
```

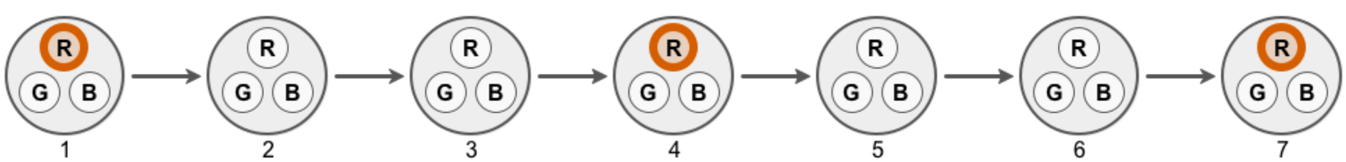
reject this trace?*



Q. Why does the formula

```
Red & G(Red => X(X(X(Red))))
```

reject this trace?*



Traces true-false

Part 2/5: Match traces and formulas

The following questions present an LTLf formula and a finite trace. Your task is to decide whether the trace satisfies the formula.

For reference, the formulas use five temporal operators:

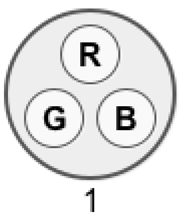
- `G(a)`: (always) `a` holds for every remaining state in the trace
- `F(a)`: (eventually) `a` holds for some state in the trace
- `a U b`: (strong until) `b` holds for some state in the trace, and `a` holds for every state between that state and the current state
- `X(a)`: (next) there must be a next state and `a` must hold in it
- `Xw(a)`: (weak next) if there is a next state, then `a` must hold in it

We first give two **Examples** to illustrate the questions and the style of answers that we are expecting.

Example Question: Is the formula

`F(Red)`

satisfied by this trace?



Example Answer: No, because Red is off in every state, because either Red or Blue is on in each state.

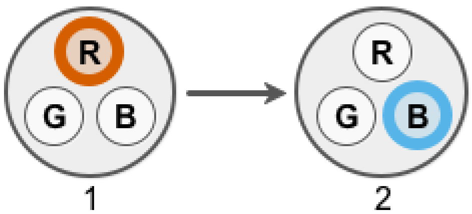
Do the **Example Question** and **Example Answer** make sense to you?*

- ◯ Yes
- ◯ [ ] No (please explain)

Example Question: Is the formula

`Red & G(Xw(Blue))`

satisfied by this trace?



Example Answer: Yes, because Red is on in the first state and Blue is on in every following state.

Do the **Example Question** and **Example Answer** make sense to you?*

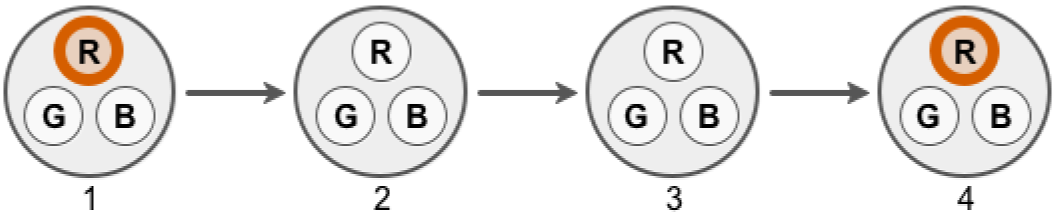
- Yes
- No (please explain)

The actual task begins now.

Q. Is the formula

`G(F(Red))`

satisfied by this trace?*



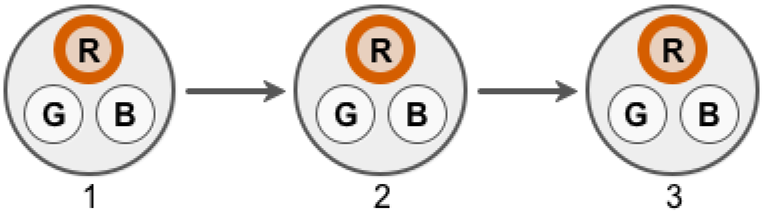
- Yes
- No

(Optional) Feel free to explain your reasoning

Q. Is the formula

`Red U (!Red)`

satisfied by this trace?*



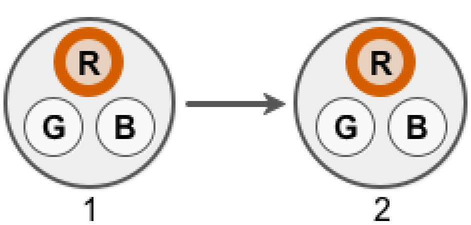
- ◯ Yes
- ◯ No

(Optional) Feel free to explain your reasoning

Q. Is the formula

`F(!X(Red))`

satisfied by this trace?*



◯ Yes

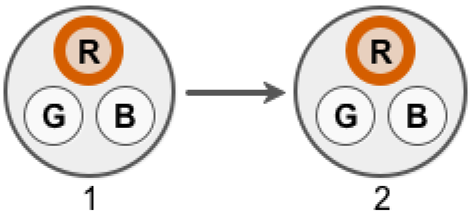
◯ No

(Optional) Feel free to explain your reasoning

Q. Is the formula

```
F(X(!Red))
```

satisfied by this trace?*



◯ Yes

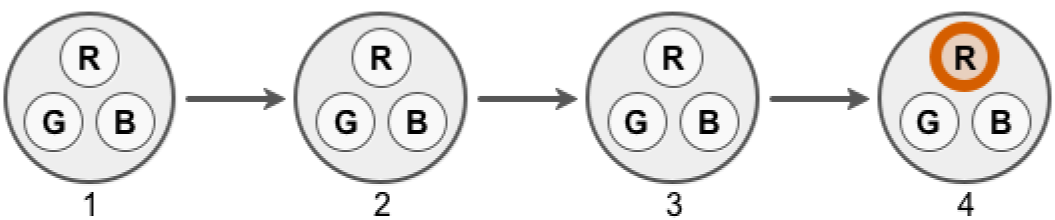
◯ No

(Optional) Feel free to explain your reasoning

Q. Is the formula

`F(G(Red))`

satisfied by this trace?*



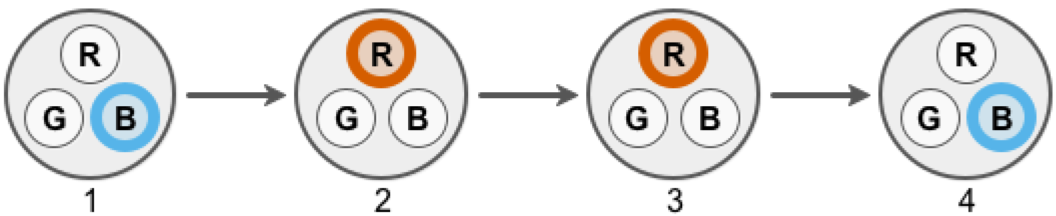
- Yes
- No

(Optional) Feel free to explain your reasoning

Q. Is the formula

`F(Red & F(Blue))`

satisfied by this trace?*



○ Yes

○ No

(Optional) Feel free to explain your reasoning

Describe Formulas

Part 3/5: Describe formulas in English

The formulas in this section are valid in both LTL and LTLf.

For each formula, provide two descriptions of its behavior: first in classic LTL (infinite-trace), then in LTLf (finite-trace).

- Your LTLf descriptions may ignore the possibility of empty traces.
- Write "same" in the LTLf box if the behavior is similar in LTL and LTLf.
  Conversely, if the behaviors are different, be sure to bring across the difference in your descriptions!
- Write "I don't know" if you have no idea how to describe a formula.

For reference, the formulas use five temporal operators:

- `G(a)`: (always) `a` holds for every remaining state in the trace
- `F(a)`: (eventually) `a` holds for some state in the trace
- `a U b`: (strong until) `b` holds for some state in the trace, and `a` holds for every state between that state and the current state
- `X(a)`: (next) there must be a next state and `a` must hold in it

- `Xw(a)`: (weak next) if there is a next state, then `a` must hold in it

We first show one **Example** to illustrate the questions and the style of answers that we are expecting.

Example Question:

`G(X(Red))`

Example Answer:

- *LTL description*: The Red light is on in every state after the first state.
- *LTLf description*: Every state must be followed by a state with Red on. No finite traces satisfy the formula.

Do the **Example Question** and **Example Answer** make sense to you?*

- ○ Yes
- ○ [ ] No (please explain)

The actual task begins now.

```
F(Red => F(!Red))
```

LTL description

LTLf description

(Optional) Feel free to explain your reasoning

`(Red U Blue) & G(Red)`

LTL description

LTLf description

(Optional) Feel free to explain your reasoning

```
Red & !X(Blue)
```

LTL description

LTLf description

(Optional) Feel free to explain your reasoning

```
G(Red => X(!Red & X(Red)))
```

LTL description

LTLf description

(Optional) Feel free to explain your reasoning

Write Formulas

Part 4/5: Translate English to formulas

Translate the following English sentences to both LTL (infinite-trace) and LTLf (finite-trace).

- Write "inexpressible" if no translation exists.
- Write "same" in the LTLf box if the same formula works for both.

Use the optional comment boxes to explain your reasoning.

As a reminder, we have been using the following atoms and connectives:

- `Red`, `Green`, `Blue`
- `&`, ||, `=>`, `!`
- `G(a)`: (always) `a` holds for every remaining state in the trace
- `F(a)`: (eventually) `a` holds for some state in the trace
- `a U b`: (strong until) `b` holds for some state in the trace, and `a` holds for every state between that state and the current state
- `X(a)`: (next) there must be a next state and `a` must hold in it
- `Xw(a)`: (weak next) if there is a next state, then `a` must hold in it

We first show one **Example** to illustrate the questions and the style of answers that we are expecting.

Example Question: Whenever the Red light is on, there is a next state that has Blue on.

Example Answer:

- LTL formula: `G(Red => X(Blue))`
- LTLf formula: `same`

Do the **Example Question** and **Example Answer** make sense to you?*

- ○ Yes
- ○ [ ] No (please explain)

The actual task begins now.

Whenever the Red light is on, there cannot be a next state in which Green is on.

LTL formula

LTLf formula

(Optional) Feel free to explain your reasoning

There are at least two states in which the Blue light is on.

LTL formula

LTLf formula

(Optional) Feel free to explain your reasoning

Green is on in the final state.

LTL formula

LTLf formula

(Optional) Feel free to explain your reasoning

Blue is on in the first state, off in the second state, and alternates on/off for the remaining states.

LTL formula

LTLf formula

(Optional) Feel free to explain your reasoning

Red is on exactly once.

LTL formula

LTLf formula

(Optional) Feel free to explain your reasoning

Compare Identities

Part 5/5: Check Equations

For each of the following equalities, most of which hold in LTL, decide whether it holds in LTLf for **non-empty** traces.

In the formulas, `a` and `b` stand for arbitrary LTL / LTLf terms. An equation holds when all substitutions of these variables yield a true statement.

For reference, the formulas use five temporal operators:

- `G(a)`: (always) `a` holds for every remaining state in the trace
- `F(a)`: (eventually) `a` holds for some state in the trace
- `a U b`: (strong until) `b` holds for some state in the trace, and `a` holds for every state between that state and the current state
- `X(a)`: (next) there must be a next state and `a` must hold in it
- `Xw(a)`: (weak next) if there is a next state, then `a` must hold in it

We first show one **Example** to illustrate the questions and the style of answers that we expect.

Example Question:

`G(!a) == !F(a)`

Example Answer: Yes, that equation is valid in LTLf when the variable `a` is replaced by any LTLf term.

Do the **Example Question** and **Example Answer** make sense to you?*

- ○ Yes
- ○ [ ] No (please explain)

The actual task begins now.

`!X(a) == X(!a)`

- ○ True in LTLf
- ○ False in LTLf

(Optional) Feel free to explain your reasoning

`G(a) == a & X(G(a))`

- True in LTLf
- False in LTLf

(Optional) Feel free to explain your reasoning

`F(a) == a || X(F(a))`

- ○ True in LTLf
- ○ False in LTLf

(Optional) Feel free to explain your reasoning

(False in LTL) `G(F(a)) == F(G(a))`

- ○ True in LTLf
- ○ False in LTLf

(Optional) Feel free to explain your reasoning

Wrap up

Click the right arrow (->) below to submit.

On the next page, you will receive a PDF copy of your responses.