Examples & board definitions for part 3 of the "Model Checking and Games" lecture series

Ruediger Ehlers

September 4, 2019

1 Definition Linear Temporal Logic

1.1 Syntax

Let AP be a set of atomic propositions. We define the syntax of an LTL formula by the following grammar in Backus-Naur Form (for $p \in \mathsf{AP}$):

$$\psi ::= \textbf{true} \mid \textbf{false} \mid p \mid \neg\psi \mid \psi \wedge \psi \mid \psi \vee \psi \mid \mathsf{X}\,\psi \mid \mathsf{G}\,\psi \mid \mathsf{F}\,\psi \mid \psi\,\mathcal{U}\,\psi \mid \psi\,\mathcal{W}\,\psi$$

So for $\mathsf{AP} = \{x, y, z\}$, the following are valid LTL formulas:

- x
- $\mathsf{GF}\,y$
- $(x\,\mathcal{U}\mathsf{F}\,y) \vee \mathsf{FG}\neg z$

The following are not valid LTL formulas:

- $x\wedge$
- $\mathsf{G}\,\mathcal{U}\,y$
- $x\,\mathsf{G}\,y$

Convention on operator precedences: Unary operators bind stronger than binary operators, $\wedge$ binds stronger than $\vee$. Avoid assuming more about operator precedences than that.

Note: LTL is an extension of propositional logic.

1.2 Semantics

LTL formulas over some set of propositions AP are interpreted on over infinite *words* with letters from 2^{AP}. Let $w = w_0 w_1 \ldots \in (2^{\mathsf{AP}})^\omega$ be a word. We use the notation w^j to denote the suffix word $w_j w_{j+1} w_{j+2} \ldots \in (2^{\mathsf{AP}})^\omega$ (for all $j \in \mathbb{N}$). We define the truth of an LTL Formula ψ *inductively* over the structure of ψ. For all ψ', ψ'' and words $w = w_0 w_1 \ldots \in (2^{\mathsf{AP}})^\omega$, we define:

$w \models \mathbf{true}$		in all cases
$w \models \mathbf{false}$		holds for no w
$w \models p$	iff	$p \in w_0$
$w \models \neg\psi$	iff	$\neg w \models \psi$
$w \models \psi \vee \psi'$	iff	$(w \models \psi) \vee (w \models \psi')$
$w \models \psi \wedge \psi'$	iff	$(w \models \psi) \wedge (w \models \psi')$
$w \models \mathsf{X}\psi$	iff	$w^1 \models \psi$
$w \models \mathsf{G}\psi$	iff	for all $j \in \mathbb{N}$, we have $w^j \models \psi$
$w \models \mathsf{F}\psi$	iff	there exists some $j \in \mathbb{N}$ such that $w^j \models \psi$
$w \models \psi \,\mathcal{U}\, \psi'$	iff	there exists some $j \in \mathbb{N}$ such that $w^j \models \psi'$ and for all $0 \leq k < j$, we have $w^k \models \psi$
$w \models \psi \,\mathcal{W}\, \psi'$	iff	$w \models \mathsf{G}\psi \vee \psi \,\mathcal{U}\, \psi'$

Note: The set natural numbers $\mathbb{N}$ starts with 0 here (as common internationally, but not so much in Germany).

Note: The last case is a definition based on the operators already defined.

1.3 Example 1

Let's evaluate whether $\psi = y \wedge \mathsf{F}(x \wedge y) \vee \mathsf{F}(x \,\mathcal{U}\, (y \wedge \mathsf{X}x)) \wedge \mathsf{X}x$ holds on a word over $\mathsf{AP} = \{x, y\}$. We mark, for every character in the word and every subformula of ψ, if the suffix word starting from the character satisfies the subformula of ψ. We use the definition of the LTL semantics in order to do so.

$w =$	$\{\}$	$\{x\}$	$\{x\}$	$\{x\}$	$\{\}$	$\{y\}$	$\{\}$	$\{x, y\}$	$\{\}$	$\{x\}$	$\{x\}$	$\{y\}$	$\{x\}$	$\{\}$	...
x	**0**	**1**	**1**	**1**	**0**	**0**	**0**	**1**	**0**	**1**	**1**	**0**	**1**	**0**	...
y	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**	**0**	**0**	**0**	**1**	**0**	**0**	...
$\mathsf{X}x$	**1**	**1**	**1**	**0**	**0**	**0**	**1**	**0**	**1**	**1**	**0**	**1**	**0**	...	...
$x \wedge y$	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	...
$y \wedge \mathsf{X}x$	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	...
$x \,\mathcal{U}\, (y \wedge \mathsf{X}x)$	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**0**	**0**	...
$\mathsf{F}(x \wedge y)$	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	...	...	...	...	...	...	...
$y \wedge \mathsf{F}(x \wedge y)$	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**	**0**	**0**	**0**	...	**0**	**0**	...
$\mathsf{F}(x \,\mathcal{U}\, (y \wedge \mathsf{X}x))$	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	...	...	...
$\mathsf{F}(x \,\mathcal{U}\, (y \wedge \mathsf{X}x)) \wedge \mathsf{X}x$	**1**	**1**	**1**	**0**	**0**	**0**	**1**	**0**	**1**	**1**	**0**	**1**	**0**	...	...
ψ	**1**	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**1**	**0**	...	...

It can be seen that we have $w_0 w_1 \ldots \models \psi$.

1.4 Ultimately periodic words

Now we want to look at whether an ultimately periodic word satisfies an LTL formula. While the satisfaction of the LTL formula by the word is now always well-defined, it is not automatically clear how to evaluate an LTL formula.

A first step towards a solution for this step is the following observation:

Proposition 1.1. *Let* AP *be a set of atomic propositions. If* $w = uv^\omega$ *is an ultimately periodic word over* 2^{AP} *with* $v = v_1 \ldots v_n$ *and* ψ *is an LTL formula over* AP*, then for all* $j \in \mathbb{N}$ *with* $j \geq |u|$ *and all* $k \in \mathbb{N}$*, we have that* $w^j \models \psi$ *if and only if* $w^{j+k\cdot n} \models \psi$*.*

Proof. This following directly from the fact that $w^j = w^{j+k\cdot n}$. □

This proposition allows us to label every character in the *cycle part* of the word by whether the subformula is satisfied "from there".

It does not immediately tell us how we can obtain such a labelling, though. We will examine this by means of the $\mathcal{U}$ operator:

Proposition 1.2. *Let* AP *be a set of atomic propositions. If* $w = uv^\omega$ *is an ultimately periodic word over* 2^{AP} *with* $v = v_1 \ldots v_n$ *and* $\psi \, \mathcal{U} \, \psi'$ *is an LTL formula over* AP.

For all $m \in \mathbb{N}$*, we have that* $v^m v^\omega \models \psi \, \mathcal{U} \, \psi'$ *is and only if either:*

- *there exists a* $k \in \{m, \ldots, n\}$ *such that* $v^k v^\omega \models \psi'$ *and for all* $j \in \{m, \ldots, k\}$*, we have* $v^j v^\omega \models \psi$*, or*
- *there exists a* $k \in \{0, \ldots, m-1\}$ *such that* $v^k v^\omega \models \psi'$ *and for all* $j \in \{m, \ldots, n, 0, 1, \ldots, k\}$*, we have* $v^j v^\omega \models \psi$*.*

Proof. $\underline{\Rightarrow}$: Let $v^m v^\omega \models \psi \, \mathcal{U} \, \psi'$. In this case, we know that there exists some $k \in \mathbb{N}$ such that $v^{m+k} \models \psi'$ and for all $j \in \{0, \ldots, k-1\}$, we have $v^{m+j} \models \psi$. We can assume, w.l.o.g., that $k < n$, as by the fact that the cycle of $v^m v^\omega$ has length n, we have $v^{m+k} v^\omega = v^{m+k+n} v^\omega$, so that we can subtract n from k as often as needed until we have $k < n$.

Now if $k < n - m$, then we have that $v^{m+k} \models \psi$ and for all $j \in \{m, \ldots, m+k\}$, we have $v^j v^\omega \models \psi$, so the first condition from above is fulfilled.

If however $k \geq n - m$, then we know that for all $j \in \{m, \ldots, n\}$, we have that $v^j v^\omega \models \psi$, and since $(vv)^j v^\omega \models \psi = v^{j-n} v^\omega$, we furthermore have that $v^j v^\omega \models \psi$ for all $j \in \{0, \ldots m + k - n\}$. Furthermore, $v^{m+k-n} \models \psi'$. Thence, the second condition is fulfilled.

$\underline{\Leftarrow}$: Similar □

The proposition tells us that that to evaluate $\psi \, \mathcal{U} \, \psi'$ on the cycle of an ultimately periodic word, we only need to mark the places at the cycle at which ψ' holds and then mark the ones to the left of them while ψ holds from there. We will look at this by means of an example.

We consider the formula $x \mathcal{U} y$:

$w =$	$\{\}$	$\{y\}$	$\{y\}$	$\{\}$	$\{x\}$	$(\{x,y\}$	$\{\}$	$\{x\})^\omega$
x	**0**	**0**	**0**	**0**	**1**	**1**	**0**	**1**
y	**0**	**1**	**1**	**0**	**0**	**1**	**0**	**0**
$x\,\mathcal{U}\,y$	**0**	**1**	**1**	**0**	**1**	**1**	**0**	**1**

For the case of the F operator, we can use the fact that for all LTL subformulas ψ, the subformula ψ is equivalent to **true** $\mathcal{U}\psi$. Hence, we can just apply the result to F subformulas as well:

$w =$	$\{\}$	$\{y\}$	$\{y\}$	$\{\}$	$\{x\}$	$(\{x,y\}$	$\{\}$	$\{x\})^\omega$
x	**0**	**0**	**0**	**0**	**1**	**1**	**0**	**1**
y	**0**	**1**	**1**	**0**	**0**	**1**	**0**	**0**
$x\,\mathcal{U}\,y$	**0**	**1**	**1**	**0**	**1**	**1**	**0**	**1**
$\mathsf{F}(x\,\mathcal{U}\,y)$	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

Similarly, we can use the fact that $\mathsf{G}\psi$ can also be rewritten in the form $\mathsf{G}\psi = \neg\mathsf{F}\neg\psi$.

$w =$	$\{\}$	$\{y\}$	$\{y\}$	$\{\}$	$\{x\}$	$(\{x,y\}$	$\{\}$	$\{x\})^\omega$
x	**0**	**0**	**0**	**0**	**1**	**1**	**0**	**1**
y	**0**	**1**	**1**	**0**	**0**	**1**	**0**	**0**
$x\,\mathcal{U}\,y$	**0**	**1**	**1**	**0**	**1**	**1**	**0**	**1**
$\mathsf{X}x$	**0**	**0**	**0**	**1**	**1**	**0**	**1**	**1**
$\mathsf{X}x \vee x$	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**
$\mathsf{G}(x \vee \mathsf{X}x)$	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**

2 Computation trees of labelled transition systems (LTS)

Let's start with a formal definition of what a tree is. Henceforth, we denote the empty word with ϵ.

Definition 2.1. *Let D and Σ be sets. A* tree *is a tuple $\langle T, \tau\rangle$ with $T \subseteq D^*$ and $\tau : T \to \Sigma$ such that T is* prefix-closed, *i.e., for every $t_0 \dots t_n \in T$, we also have that $t_0 \dots t_{n-j} \in T$ for all $0 \leq j \leq n$.*

We define the *computation tree* of a Kripke structure as follows:

Definition 2.2. *Let $\mathcal{K} = (S, S_0, T, \mathsf{AP}, L)$ be a Kripke structure with $S_0 = \{s_0\}$. We define the computation tree $\langle \mathcal{T}, \tau\rangle$ induced by $\mathcal{K}$ as follows:*

$$\mathcal{T} = \{s_0 \dots s_n \in S^* \mid \forall 0 \leq j \leq n-1 : (s_j, s_{j+1}) \in T\}$$
$$\forall s_0 \dots s_n \in \mathcal{T} : \tau(s_0 \dots s_n) = L(s_0 \dots s_n)$$

Thus, the tree $\langle \mathcal{T}, \tau\rangle$ has 2^{AP} as the co-domain of τ and $D = S$ as the set of directions.

We say that $s_0 s_1 \ldots \in D^\omega$ is an infinite *branch* of $\langle \mathcal{T}, \tau \rangle$ if for all $j \in \mathbb{N}$, we have $s_0 \ldots s_j \in T$. We say that some word $\rho = \rho_0 \rho_1 \ldots \in \Sigma^\omega$ is the *branch labelling* for $s_0 s_1 \ldots \in D^\omega$ if for all $j \in \mathbb{N}$, we have $\tau(s_0 s_1 \ldots s_{j-1}) = \rho_j$. We denote the set of all infinite branches of $\langle T, \tau \rangle$ as $\mathsf{Branches}(T)$.

Proposition 2.1. *Let $\mathcal{K}$ be a Kripke structure and $\langle \mathcal{T}, \tau \rangle$ be the corresponding computation tree. The set of traces of $\mathcal{K}$ is exactly the set of branch labellings for some branches of $\langle \mathcal{T}, \tau \rangle$*

Proof. Proof omitted. □

A computation tree is thus a representation of the set of traces of the corresponding Kripke structure.

We define that $\langle \mathcal{T}, \tau \rangle \models \psi$ for some LTL formula ψ if all branch labelings of $\langle \mathcal{T}, \tau \rangle$ satisfy ψ. Likewise, we write $\mathcal{K} \models \psi$ if we have $\langle \mathcal{T}, \tau \rangle \models \psi$ for the corresponding computation tree $\langle \mathcal{T}, \tau \rangle$.

3 Model checking with `spin`

First model:

```
int turn = 0;
int flag0 = 0;
int flag1 = 0;
active proctype A() {
    do
        :: {
            flag0 = 1;
            turn = 0;
            do
                :: flag1==1 && turn==0 -> skip;
                :: else -> break;
            od
            do
                :: skip -> skip;
                :: break;
            od
            flag0 = 0;
            do
                :: skip -> skip;
                :: break;
            od
        }
    od
}

active proctype B() {
    do
        :: {
            flag1 = 1;
            turn = 1;
            do
                :: flag0==1 && turn==0 -> skip;
                :: else -> break;
            od
            do
                :: skip -> skip;
                :: break;
            od
            flag1 = 0;
            do
                :: skip -> skip;
                :: break;
            od
        }
    od
}

#define p (flag0==0)
ltl { []<>(p); }
```

This checks if it is always the case that `flag0` infinitely often has a value of 0.

Notes:

- In `iSpin`, click on **Use never claim**
- When changing the specification and using the `iSpin` user interface, click **save** before running the verification step again.

3.1 Checking mutual exclusion

```
int turn = 0;
int flag0 = 0;
int flag1 = 0;
active proctype A() {
    do
        :: {
            flag0 = 1;
            turn = 0;
            do
                :: flag1==1 && turn==0 -> skip;
                :: else -> break;
            od
            do
                :: skip ->
                critA: skip;
                :: break;
            od
            flag0 = 0;
            do
                :: skip -> skip;
                :: break;
            od
        }
    od
}

active proctype B() {
    do
        :: {
            flag1 = 1;
            turn = 1;
            do
                :: flag0==1 && turn==0 -> skip;
                :: else -> break;
            od
            do
                :: skip -> critB: skip;
                :: break;
            od
            flag1 = 0;
            do
                :: skip -> skip;
                :: break;
            od
        }
    od
}

#define p (A@critA)
#define q (B@critB)
ltl { [](!p | !q); }
```

We can see that the property is violated. Fixing the `turn == 0` condition in the `B` process to `turn==1` yields a model with a correct mutual exclusion result.

4 Computation tree logic

4.1 Syntax

Let AP be a set of atomic propositions. We define the syntax of a CTL formula by the following grammar in Backus-Naur Form:

$$\psi ::= \mathbf{true} \mid \mathbf{false} \mid p \mid \neg\psi \mid \psi \wedge \psi \mid \psi \vee \psi \mid \mathsf{AX}\,\psi \mid \mathsf{AG}\,\psi \mid \mathsf{AF}\,\psi \mid \mathsf{A}(\psi\,\mathcal{U}\,\psi) \mid \mathsf{A}(\psi\,\mathcal{W}\,\psi)$$

$$\mid \mathsf{EX}\,\psi \mid \mathsf{EG}\,\psi \mid \mathsf{EF}\,\psi \mid \mathsf{E}(\psi\,\mathcal{U}\,\psi) \mid \mathsf{E}(\psi\,\mathcal{W}\,\psi)$$

So for $\mathsf{AP} = \{x, y, z\}$, the following are valid CTL formulas:

- x
- $\mathsf{AGAF}\,y$
- $\mathsf{A}(x\,\mathcal{U}\mathsf{AF}\,y) \vee \mathsf{EF}\neg z$

The following are not valid CTL formulas:

- $x\wedge$
- $\mathsf{G}\,\mathcal{U}\,y$
- $\mathsf{A}(x\,\mathsf{G}\,y)$

Convention on operator precedences: Unary operators bind stronger than binary operators, $\wedge$ binds stronger than $\vee$. Avoid assuming more about operator precedences than that.

Note: CTL is an extension of propositional logic.

4.2 Semantics

CTL formulas over some set of propositions AP are interpreted on over infinite *trees* that are labelled by letters from 2^{AP}.

Given a tree $\langle T, \tau\rangle$ over some set of directions D and some $t \in T$, we define the *subtree* rooted at t, written as $\langle T, \tau\rangle_t$ as the tree $\langle T', \tau'\rangle$ with:

- $T' = \{t' \in D^* \mid tt' \in T\}$
- For all $t' \in T'$, $\tau'(t') = \tau(tt')$.

We define the satisfaction of a CTL formula by a computation tree recursively over the structure of the CTL formula. For all proposition sets AP, all CTL subformulas ψ and ψ' and trees $\langle T, \tau\rangle$ with τ having 2^{AP} as co-domain, we define:

$\langle T, \tau\rangle \models$ **true**		in all cases
$\langle T, \tau\rangle \models$ **false**		holds for no w
$\langle T, \tau\rangle \models p$	iff	$p \in \tau(\epsilon)$
$\langle T, \tau\rangle \models \neg\psi$	iff	$\neg\langle T, \tau\rangle \models \psi$
$\langle T, \tau\rangle \models \psi \vee \psi'$	iff	$(\langle T, \tau\rangle \models \psi) \vee (\langle T, \tau\rangle \models \psi')$
$\langle T, \tau\rangle \models \psi \wedge \psi'$	iff	$(\langle T, \tau\rangle \models \psi) \wedge (\langle T, \tau\rangle \models \psi')$
$\langle T, \tau\rangle \models \mathsf{AX}\psi$	iff	for all $d \in D$ with $d \in T, \langle T, \tau\rangle_d \models \psi$
$\langle T, \tau\rangle \models \mathsf{EX}\psi$	iff	there exists some $d \in D$ with $d \in T \wedge \langle T, \tau\rangle_d \models \psi$
$\langle T, \tau\rangle \models \mathsf{AG}\psi$	iff	for all $t \in T, \langle T, \tau\rangle_t \models \psi$
$\langle T, \tau\rangle \models \mathsf{EG}\psi$	iff	there exists some $d_0 d_1 \ldots \in \mathsf{Branches}(T)$ s.t. $\forall j \in \mathbb{N}, \langle T, \tau\rangle_{d_0 d_1 \ldots d_j} \models \psi$
$\langle T, \tau\rangle \models \mathsf{AF}\psi$	iff	for all $d_0 d_1 \ldots \in \mathsf{Branches}(T).\exists j \in \mathbb{N}.\langle T, \tau\rangle_{d_0 d_1 \ldots d_j} \models \psi$
$\langle T, \tau\rangle \models \mathsf{EF}\psi$	iff	there exists some $t \in T.\langle T, \tau\rangle_t \models \psi$
$\langle T, \tau\rangle \models \mathsf{A}(\psi\,\mathcal{U}\,\psi')$	iff	for all $d_0 d_1 \ldots \in \mathsf{Branches}(T).\exists j \in \mathbb{N}.\langle T, \tau\rangle_{d_0 d_1 \ldots d_j} \models \psi'$

$$
\begin{array}{lll}
 & & \text{and for all } 0 \leq k < j. \langle T, \tau \rangle_{d_0 d_1 \ldots d_k} \models \psi \\
\langle T, \tau \rangle \models \mathsf{E}(\psi \, \mathcal{U} \, \psi') & \text{iff} & \text{there exists some } d_0 d_1 \ldots \in \mathsf{Branches}(T) \text{ and } j \in \mathbb{N}. \langle T, \tau \rangle_{d_0 d_1 \ldots d_j} \models \psi' \\
 & & \text{and for all } 0 \leq k < j. \langle T, \tau \rangle_{d_0 d_1 \ldots d_k} \models \psi \\
\langle T, \tau \rangle \models \mathsf{A}(\psi \, \mathcal{W} \, \psi') & \text{iff} & \text{for all } d_0 d_1 \ldots \in \mathsf{Branches}(T). (\exists j \in \mathbb{N}. \langle T, \tau \rangle_{d_0 d_1 \ldots d_j} \models \psi' \\
 & & \text{and for all } 0 \leq k < j. \langle T, \tau \rangle_{d_0 d_1 \ldots d_k} \models \psi) \vee (\forall j \in \mathbb{N}. \langle T, \tau \rangle_{d_0 d_1 \ldots d_j} \models \psi) \\
\langle T, \tau \rangle \models \mathsf{E}(\psi \, \mathcal{W} \, \psi') & \text{iff} & \text{there exists some } d_0 d_1 \ldots \in \mathsf{Branches}(T). (j \in \mathbb{N}. \langle T, \tau \rangle_{d_0 d_1 \ldots d_j} \models \psi' \\
 & & \text{and for all } 0 \leq k < j. \langle T, \tau \rangle_{d_0 d_1 \ldots d_k} \models \psi) \vee (\forall j \in \mathbb{N}. \langle T, \tau \rangle_{d_0 d_1 \ldots d_j} \models \psi)
\end{array}
$$