LTL: Linear-time logic

LTL is another important temporal logic.

In LTL the future is seen as a sequence of states, so the future is seen as a path.

We can consider a set of paths, however we don't have an existential path quantifier (the **E** of CTL), when we interpret an LTL formula over a set of paths the formula has to be true of all paths to be true, i.e we are always quantifying universally over all possible paths in the set.

In LTL propositions are interpreted not over trees (like CTL does) but over individual paths,

As usual there are atomic proposition $p_1, \ldots, p_n, \ldots$

Other formulas are as follows:

$$\phi ::= \texttt{true} | \texttt{false} | \phi_1 \wedge \phi_2 | \phi_1 \vee \phi_2 | \neg\phi | \phi \rightarrow \phi |$$

$$\texttt{G}\phi | \texttt{F}\phi | \texttt{X}\phi | [\phi \texttt{U} \phi] | [\phi \texttt{W} \phi] | [\phi \texttt{R} \phi]$$

So we are missing few CTL operators (the temporal modalities) and adding two new: `W`, `R` whose meaning is `W`eak until and `R`elease

Regarding the absence of path quantifiers $\mathtt{A}$ and $\mathtt{E}$.

We said that this is because a formula is evaluated on a path, or a set of paths

A set of paths satisfy ϕ iff every path in the set satisfy ϕ.

That mean that implicitly all formulas are universally path quantified, i.e.

$[p\mathtt{U}q]$ stands for $\mathtt{A}[p\mathtt{U}q]$ etc...

We don't have path quantifiers, so we know we can't write things like

E (p U q)

or

E F p or

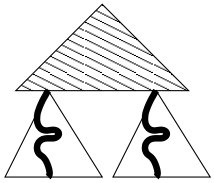
A G E F p

however there are things we can write in LTL that we couldn't write in CTL.

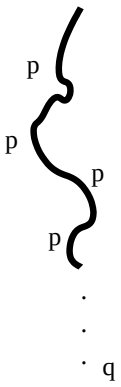
Consider $\mathtt{F}p \rightarrow \mathtt{F}q$ and compare it to his closest CTL relative: $\mathtt{AF}p \rightarrow \mathtt{AF}q$

There are things we can write in CTL but not in LTL, things like "on all paths is always true that exists a path such that ..."

There are things we can write in LTL but not in CTL, like "If it is infinitely often the case that p then at some point q"



A G E G p



G F p → Fq

Remember that in CTL the semantics was defined as a relation

$\mathcal{K}, s \models \phi$ where s is a state in the KTS

LTL semantics will be defined as a relation

$\mathcal{K}, \pi \models \phi$ where π is a path in the KTS.

we will note paths as follows:

$\pi = s_0 \rightarrow s_1 \rightarrow \ldots s_n \rightarrow$

and $\pi^i = s_i \rightarrow \ldots s_n \rightarrow$

i.e. π^i is the subpath of π starting at s_i.

Let's see how to give a semantics to LTL.

We consider the same structure we used for the semantics of CTL, i.e. a KTS $\mathcal{K}$=(S,$\rightarrow$,I)

Given a KTS $\mathcal{K}$, we are going to define the relation

$\mathcal{K}, \pi \models \phi$

meaning "the path π in $\mathcal{K}$ satisfies the formula ϕ" .

The definition is by induction on the structure of ϕ

- $\mathcal{K}, \pi \models \texttt{true} \quad \mathcal{K}, \pi \not\models \texttt{false}$
- $\mathcal{K}, \pi \models p$ iff $p \in I(s_0)$
- $\mathcal{K}, \pi \models \neg\phi$ iff $\mathcal{K}, \pi \not\models \phi$
- $\mathcal{K}, \pi \models \phi_1 \wedge \phi_2$ iff $\mathcal{K}, \pi \models \phi_1$ and $\mathcal{K}, \pi \models \phi_2$
- $\mathcal{K}, \pi \models \phi_1 \vee \phi_2$ iff $\mathcal{K}, \pi \models \phi_1$ or $\mathcal{K}, \pi \models \phi_2$
- $\mathcal{K}, \pi \models \phi_1 \rightarrow \phi_2$ iff $\mathcal{K}, \pi \not\models \phi_1$ or $\mathcal{K}, \pi \models \phi_2$
- $\mathcal{K}, \pi \models \texttt{G}\phi$ iff for all paths π^i for all $i \geq 0$ we have $\mathcal{K}, \pi^i \models \phi$
- $\mathcal{K}, \pi \models \texttt{F}\phi$ iff for some path π^i for some $i \geq 0$ we have $\mathcal{K}, \pi^i \models \phi$

- $\mathcal{K}, \pi \models \mathtt{X}\phi$ iff $\mathcal{K}, \pi^1 \models \phi$

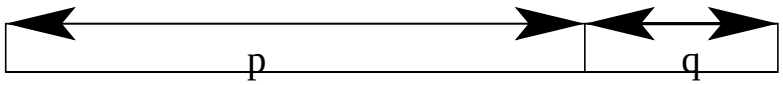
- $\mathcal{K}, \pi \models [\phi_1 \mathtt{U} \phi_2]$ iff exists $i \geq 0$ such that $\mathcal{K}, \pi^i \models \phi_2$ and for all $j < i$, $\mathcal{K}, \pi^j \models \phi_1$

- $\mathcal{K}, \pi \models [\phi_1 \mathtt{W} \phi_2]$ iff exists $i \geq 0$ such that $\mathcal{K}, \pi^i \models \phi_2$ and for all $j < i$, $\mathcal{K}, \pi^j \models \phi_1$ or for all $i \geq 0$ $\mathcal{K}, \pi^i \models \phi_1$

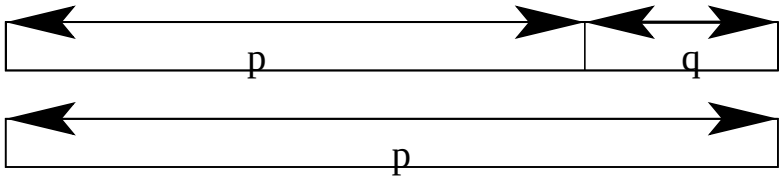
- $\mathcal{K}, \pi \models [\phi_1 \mathtt{R} \phi_2]$ iff exists $i \geq 0$ such that $\mathcal{K}, \pi^i \models \phi_1$ and for all $j \leq i$, $\mathcal{K}, \pi^j \models \phi_2$ or for all $i \geq 0$ $\mathcal{K}, \pi^i \models \phi_2$

Notes:

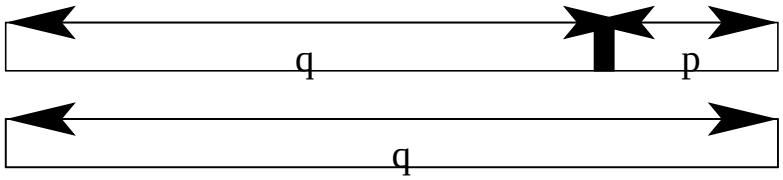
- $[\phi_1 \mathtt{W} \phi_2]$ is the same as $[\phi_1 \mathtt{U} \phi_2] \vee \mathtt{G}\phi_1$
- $[\phi_1 \mathtt{R} \phi_2]$ is the same as $\neg[\neg\phi_1 \mathtt{U} \neg\phi_2]$



p Until q



p Weak Until q

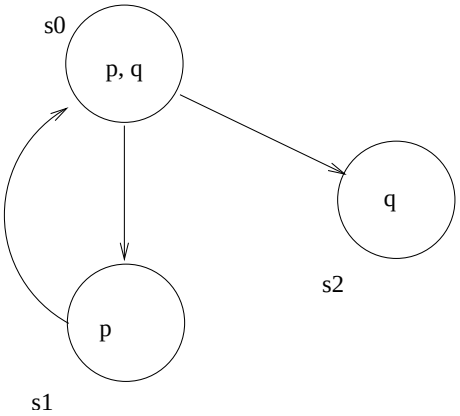


p Releases q

Few words about paths:

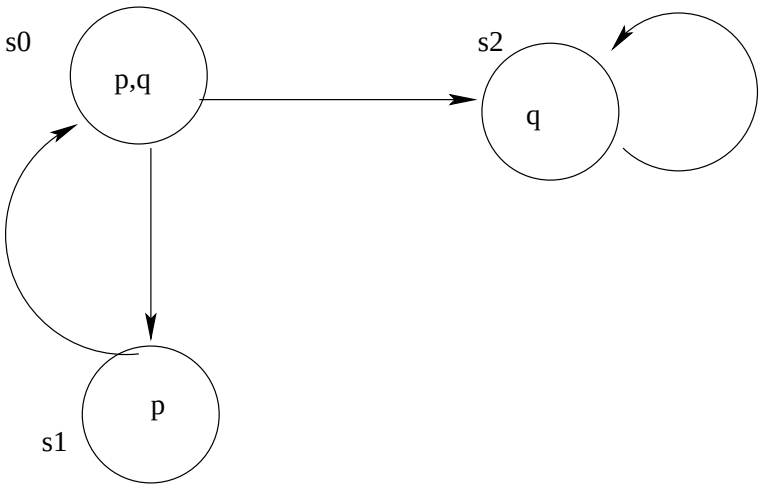
Strictly speaking, given a KTS and a state, the paths starting at that state s are all sequences of states starting from s such that if s' is in the path and $s' \rightarrow s$'' is a transition in the KTS then s'' is in the path as well.

So the paths (starting at s_0) in



are the sequences of the form $(s_0 s_1)^n s_0 s_2$ where $n \geq 0$

Most authors require paths in LTL to be infinite; as such our previous KTS would have no path. However is easily seen that we can always transform finite paths into infinite ones by adding loops on the ending states. Here is what our previous KTS would become



Hence to keep things simple we don't require paths to be infinite. Also we may want sometimes to use finite prefixes of paths:

In the KTS just seen and starting at s_0 we have

$s_0 s_2$,
$s_0 s_1$
$s_0 s_1 s_0$
$\vdots$
$(s_0 s_1)^n s_0$
$(s_0 s_1)^n s_0 s_2$
$(s_0 s_1)^n s_0 s_1$

You should read the paths ending with s_0 and s_1 as finite prefixes of paths as you could always extend a path ending with s_0 with an s_2 and one ending with s_1 with $s_0 s_2$

Here we use finite prefixes to illustrate the meaning of LTL formulas:

$(s_0 s_1)^n s_0 \not\models \mathtt{G}q$, $(s_0 s_1)^n s_0 \models \mathtt{F}q$,
$(s_0 s_1)^n s_0 \models \mathtt{GF}q$

$(s_0 s_1)^n s_0 s_1 \models \mathtt{G}p, s_0 s_2 \models \mathtt{G}q$ but
$\{(s_0 s_1)^n s_0 s_1, s_0 s_2\} \not\models \mathtt{G}p \wedge \mathtt{G}q$ but
$\{(s_0 s_1)^n s_0 s_1, s_0 s_2\} \models \mathtt{G}p \vee \mathtt{G}q$

$(s_0 s_1)^n s_0 \models (p \mathtt{U} q)$

what about $(s_0 s_1)^n s_0 s_1 \models (q \mathtt{R} p)$?

Few equivalences of LTL formulas:

$\neg \mathtt{G}\phi \equiv \mathtt{F}\neg\phi \quad \neg \mathtt{F}\phi \equiv \mathtt{G}\neg\phi \quad \neg \mathtt{X}\phi \equiv \mathtt{X}\neg\phi$

$\neg(\phi \mathtt{U} \psi) \equiv \neg\phi \mathtt{R} \neg\psi \quad \phi \mathtt{U} \psi \equiv (\phi \mathtt{W} \psi) \wedge \mathtt{F}\psi$

$\mathtt{F}\phi \equiv \mathtt{true\ U}\phi \quad \mathtt{G}\phi \equiv \mathtt{false\ R}\phi$

What we can express with LTL formulas?

Over the years few classes of properties of systems have emerged as suitable to be studied by these temporal logics:

what kind of properties make sense for all possible systems modeled by KTS?

Things like "the power is on", "there is enough fuel" etc... are not general properties

but the following are:

a Reachability : "A particular state is reachable from the present state"

b Safety: "A (bad) property will never be satisfied"

c Liveness: "A (good) property will always be satisfied by some state in the future"

c Deadlock freedom: "a dead end state will never be reached"

Safety is to rule out bad (catastrophic) behaviours of the system.

For example in a vending machine an example of safety is: ”a state where a drink is given with no money inserted is never met”

Some more examples:
”A missile is never launched unless the launch button is pressed”
”A traffic light is not green if its orthogonal light is green”
”A train will not cross a train crossing with the gates in the up position”

Liveness is to ensure that the system does what is is meant to do.

Example of liveness:
"If enough money is in the machine a drink will be released"

"if the launch button is pressed the missile will be launched"
"If the light is orange then it will become red next"
"If the train is approaching the crossing the gates will go down"

So a live and safe system is a system which does what is meant to do and doesn't do things it is not meant to do. That should ensure that the system is well-behaving, shouldn't it?

Consider two traffic light in a junction satisfying the liveness and safety conditions described above.

Would that be properly working?

Solution: Have you ever been at a red traffic light waiting for the light to become green, and waiting ... and waiting ...

You can have systems where components are waiting forever for some other component to finish.

If A is waiting for B who is waiting for A you have a *deadlock*.

If A is waiting for B, B is properly working but it never gets back to A you have *un-fairness*

In KTS terms deadlock means that the system has reached a state s which has no successor (i.e. there is no outgoing transition out of s).

Informally fairness means that a state that is available infinitely often will eventually be reached, for example at a junction if it is possible to turn left then at some point some car will turn left.

Roughly speaking most systems which are safe, live, deadlock free and fair are systems working properly