9. Temporal Logic

This logic is an extension of propositional logic, but also an interpretation of modal logic where properties on time can be specified.

Modal logic

Two basic modalities on a statement p is included "possibly p" and "necessarily p"

p is possibly true, p might be true, p is usually true

When these modalities are combined with negation we observe that

"It is not necessarily true that p is true" ≡ "It is possible that p is not true"

possibly p is written $\Diamond p$
necessarily p is written $\Box p$
The equivalent sentences above can then be formulated as

$$\neg \Box p \equiv \Diamond \neg p$$

$$\boxed{\begin{array}{l} \Box p \equiv \neg \Diamond \neg p \\ \Diamond p \equiv \neg \Box \neg p \end{array}}$$

Temporal logic

In temporal logic
necessarily is replaced by always
possibly — " — eventually

Ex Heater

$p :=$ "Heater is on"

$q :=$ "It is warm"

$\Box(p \to \Diamond q) \equiv$ "Always, if heater is on, eventually it will be warm"

Linear time

Time can be represented as a tuple $(T, <)$

T = set of time instances

$t_k, t_\ell \in T$

Time is linear if for all elements in T we have that either

$t_k < t_\ell$ or $t_\ell < t_k$

For $T = \{t_0, t_1, t_2, \ldots\}$ linear time is achieved by assuming that for instance

$t_0 < t_1 < t_2 < t_3 < \ldots$

Linear temporal logic (LTL)

Basic LTL does not include explicit time instances, only linear ordering between them

In LTL $T = \mathbb{N} = \{0, 1, 2, \ldots\}$

while an extended temporal logic including specific time instances $\Rightarrow$

$T = \{0, 2.5, 5, 10, 17, \ldots\}$

Basic LTL operators

circle

1) Next operator: $\bigcirc p$ means that p is true at the next time instance

2) Eventually p: $\Diamond p$

p happens sometime in the future at least one time.

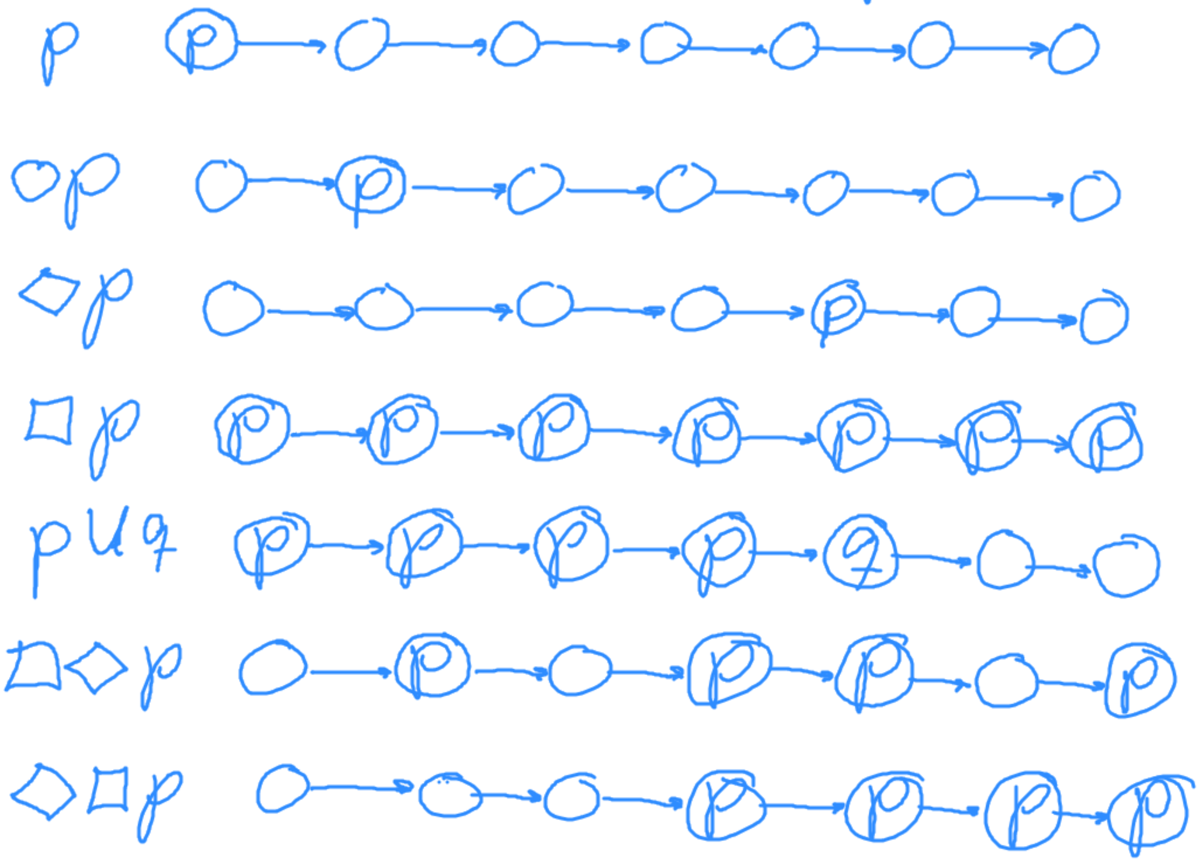
3) Always p: $\Box p$

p is always true

4) p until q: $p \,\mathcal{U}\, q$

p is true at all time instances until q is true at least at one time instance

Examples of sequences, see Fig 9.1 page 212.



Empty circle ○ means don't care (any statement on p and q are accepted)

Syntax of LTL

An LTL formula is defined inductively by the grammar

$$\varphi ::= p \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \bigcirc\varphi \mid \varphi_1 \mathcal{U} \varphi_2$$

where $p \in AP$ = set of atomic propositions

$\varphi_1 = p$, $\varphi_2 = q \in AP$

$\varphi = \varphi_1 \wedge \varphi_2 = p \wedge q$

$\bigcirc\varphi = \bigcirc(p \wedge q)$

$\neg\varphi = \neg(p \wedge q) = \neg p \vee \neg q$

$\varphi_1 \mathcal{U} \varphi_2 = p \mathcal{U} q$

Derived operators

$\varphi_1 \vee \varphi_2 \overset{def}{=} \neg(\neg\varphi_1 \wedge \neg\varphi_2)$

$\varphi_1 \to \varphi_2 \overset{def}{=} \neg\varphi_1 \vee \varphi_2$

$\Diamond\varphi \overset{def}{=} \top \,\mathcal{U}\, \varphi$ $\quad$ $\Box\varphi \overset{def}{=} \neg\Diamond\neg\varphi$

$\top \overset{def}{=} \varphi \vee \neg\varphi$ $\quad$ $\bot \overset{def}{=} \neg\top$

$\Box\Diamond\varphi \equiv$ "infinitely often φ"

$\Diamond\Box\varphi \equiv$ "eventually always φ"

precedence order

$p \to \Diamond q \equiv p \to (\Diamond q)$

$p \wedge q \,\mathcal{U}\, r \equiv p \wedge (q \,\mathcal{U}\, r)$

$\Box p \wedge q \not\equiv \Box(p \wedge q)$

Ex LTL formulas for a simple software program

```
while true do
   if y > 0 then
      y := y - 1
   else
      y := n
   end if
end while
```

Assume initial value $y = n$

$y = k \leftrightarrow y_k$

$y_n \to y_{n-1} \to \cdots \to y_1 \to y_0 \to y_n$

$0 \le y \le n$ will always be valid

This can also be specified by LTL as

$\Box(y_0 \vee y_1 \vee \ldots \vee y_n)$

as well as the LTL formula
$\Box(0 \leq y \leq n)$

The LTL formula $\Box\Diamond y_0$ also holds since $y=0$ will be repeated an infinite number of times. Thus, infinitely often $y=0$.

Introduce the boolean function even(y) and the LTL formula

$$\Box(\text{even}(y) \rightarrow \bigcirc\bigcirc\,\text{even}(y)) \quad (*)$$

$n=2 \qquad 2, 1, 0, 2, 1, 0, \ldots$

$n=3 \qquad 3, 2, 1, 0, 3, 2, 1, 0, \ldots$

The LTL formula $(*)$ is valid for odd n but not when n is even.

Computational Tree Logic (CTL)

For automata including alternative sequences, an LTL formula holds if it is true for all sequences. In computational tree logic (CTL), the quantifers for all $\forall$ and there exists $\exists$ are also introduced

CTL: $\forall\Box\varphi \qquad \forall\Diamond\varphi$

LTL: $\Box\varphi \qquad \Diamond\varphi$

The CTL formulas $\exists\Box\varphi$ and $\exists\Diamond\varphi$ cannot be specified in LTL.