### Linear Temporal Logic - Syntax

\[
\varphi ::= p \mid (\varphi) \mid \neg \varphi \mid \varphi \land \varphi' \mid \varphi \lor \varphi' \\
\mid \circ \varphi \mid \varphi U \varphi' \mid \varphi V \varphi' \mid \Box \varphi \mid \Diamond \varphi
\]

- \( p \) – a proposition over state variables
- \( \circ \varphi \) – “next”
- \( \varphi U \varphi' \) – “until”
- \( \varphi V \varphi' \) – “releases”
- \( \Box \varphi \) – “box”, “always”, “forever”
- \( \Diamond \varphi \) – “diamond”, “eventually”, “sometime”

### LTL Semantics: The Idea

- \( p \) – \( p \)
- \( \circ \varphi \) – \( \varphi \)
- \( \varphi U \varphi' \) – \( \varphi \varphi \psi \psi \psi \psi \psi \psi \psi \psi \psi \psi \psi \)
- \( \varphi V \varphi' \) – \( \psi \psi \psi \psi \psi \psi \psi \psi \psi \psi \psi \psi \)
- \( \Box \varphi \) – \( \varphi \)
- \( \Diamond \varphi \) – \( \varphi \)

### Formal LTL Semantics

Given:
- \( G = (V, F, af, ar) \) signature expressing state propositions
- \( Q \) set of states,
- \( M \) modeling function over \( Q \) and \( G \); \( M(q, p) \) is true iff \( q \) models \( p \).
- Write \( q \models p \).
- \( \sigma = q_0 q_1 \ldots q_n \ldots \) infinite sequence of state from \( Q \).
- \( \sigma^i = q_i q_{i+1} \ldots q_n \ldots \) the \( i \)th tail of \( \sigma \).

Say \( \sigma \) models LTL formula \( \varphi \), write \( \sigma \models \varphi \) as follows:
- \( \sigma \models p \) iff \( q_0 \models p \)
- \( \sigma \models \neg \varphi \) iff \( \sigma \not\models \varphi \)
- \( \sigma \models \varphi \land \psi \) iff \( \sigma \models \varphi \) and \( \sigma \models \psi \).
- \( \sigma \models \varphi \lor \psi \) iff \( \sigma \models \varphi \) or \( \sigma \models \psi \).

### Some Common Combinations

- \( \Box \Diamond p \) “\( p \) will hold infinitely often”
- \( \Diamond \Box p \) “\( p \) will continuously hold from some point on”
- \( (\Box p) \Rightarrow (\Diamond q) \) “if \( p \) happens infinitely often, then so does \( q \)”

### Some Equivalences

- \( \Box (\varphi \land \psi) = (\Box \varphi) \land (\Box \psi) \)
- \( \Diamond (\varphi \lor \psi) = (\Diamond \varphi) \lor (\Diamond \psi) \)
- \( \Box \varphi = T (\varphi U \varphi) \)
- \( \Diamond \varphi = U (\Diamond \varphi) \)
- \( \varphi U \psi = \neg (\neg \varphi) (\psi) \)
- \( \varphi V \psi = \neg (\neg \varphi) (\psi) \)
- \( \neg (\Diamond \varphi) = \Diamond (\neg \varphi) \)
- \( \neg (\Box \varphi) = \Box (\neg \varphi) \)
**Important Meta-Definitions**

- A is sound with respect to B if things that are "true" according to A are things that are "true" according to B.
- A is complete with respect to B if things that are "true" according to B are things that are "true" according to A.
- A is sound if things that are "true" according to A are true.
- A is complete if things that are "true" according to A are true.
- A is relatively complete if things that are "true" according to A are true.
- A is sound with respect to B if A is complete when B is.
- Think: A proof system, B mathematical model; or A a proof system, B a subsystem.

**Traffic Light Example**

**Basic Behavior:**
- $\Box((\text{NSC} = \text{Red}) \lor (\text{NSC} = \text{Green}) \lor (\text{NSC} = \text{Yellow}))$
- $\Box((\text{NSC} = \text{Red}) \Rightarrow ((\text{NSC} \neq \text{Green}) \land (\text{NSC} \neq \text{Yellow})))$
- Similarly for Green and Red
- $\Box((\text{NSC} = \text{Red}) \land o(\text{NSC} \neq \text{Red}) \Rightarrow o(\text{NSC} = \text{Green}))$
- Same as $\Box((\text{NSC} = \text{Red}) \Rightarrow ((\text{NSC} = \text{Red}) U (\text{NSC} = \text{Green})))$
- $\Box((\text{NSC} = \text{Green}) \land o(\text{NSC} = \text{Green}) \Rightarrow o(\text{NSC} = \text{Yellow}))$
- $\Box(((\text{NSC} = \text{Yellow}) \land o(\text{NSC} \neq \text{Yellow}) \Rightarrow o(\text{NSC} = \text{Red}))$
- Same for EWC

**Traffic Light Example**

**Basic Safety**
- $\Box((\text{NSC} = \text{Red}) \lor (\text{EWC} = \text{Red}))$
- $\Box((\text{NSC} = \text{Red}) \land (\text{EWC} = \text{Red})) V ((\text{NSC} \neq \text{Green}) \Rightarrow (\Box (\text{NSC} = \text{Red}) \lor (\text{NC} \neq \text{Green})))$

**Basic Liveness**
- $\Diamond(\text{NSC} = \text{Red}) \land (\Diamond (\text{NSC} = \text{Green}) \lor (\Diamond (\text{NSC} = \text{Yellow})))$
- $\Diamond(\text{EWC} = \text{Red}) \land (\Diamond (\text{EWC} = \text{Green}) \lor (\Diamond (\text{EWC} = \text{Yellow})))$

**Proof System for LTL**

- First step: View $\varphi \lor \psi$ as macro: $\varphi \lor \psi = \neg ((\neg \varphi) U (\neg \psi))$
- Second Step: Extend all rules of Prop Logic to LTL
- Third Step: Add one more rule: $\Box \varphi \Rightarrow \varphi$
- Fourth Step: Add a collection of axioms (a sufficient set of 8 exists)
  - A1: $\Box \varphi \Rightarrow \neg (\Diamond \neg \varphi)$
  - A2: $\Box (\varphi \Rightarrow \psi) \Rightarrow (\Box \varphi \Rightarrow \Box \psi)$
  - A3: $\Box \varphi \Rightarrow (\varphi \land \Diamond \varphi)$
  - A4: $\Diamond \varphi \Rightarrow o \varphi$
  - A5: $o (\varphi \Rightarrow \Box \varphi) \Rightarrow (o \varphi \Rightarrow o \varphi)$
  - A6: $\Box (\varphi \Rightarrow \Diamond \varphi) \Rightarrow (\Box \varphi \Rightarrow \Box \Diamond \varphi)$
  - A7: $\varphi U \psi \Rightarrow (\varphi \land \Diamond (\Box \varphi \land \Diamond \psi))$
  - A8: $\varphi U \psi \Rightarrow o \Diamond \varphi$
- Result: a sound and relatively complete proof system
- Can implement in Isabelle in much the same way as we did Hoare Logic
**What is Model Checking?**

Most generally **Model Checking** is
- an automated technique, that given
- a finite-state model \( M \) of a system
- and a logical property \( \varphi \),
- checks whether the property holds of model: \( M \models \varphi \)?

---

**Model Checking**

- Model checkers usually give example of failure if \( M \not\models \varphi \).
- This makes them useful for debugging.
- **Problem**: Can only handle finite models: unbounded or continuous data sets can’t be directly handled
  - Symbolic model checking can handle limited cases of finitely presented models
- **Problem**: Number of states grows exponentially in the size of the system.
- **Answer**: Use abstract model of system
- **Problem**: Relationship of results on abstract model to real system?

---

**LTL Model Checking**

- **Model Checking Problem**: Given model \( M \) and logical property \( \varphi \) of \( M \), does \( M \models \varphi \)?
- Given transition system with states \( Q \), transition relation \( \delta \) and initial state \( I \), say \( (Q, \delta, I) \models \varphi \) for LTL formula \( \varphi \) if every run of \( (Q, \delta, I), \sigma \) satisfies \( \sigma \models \varphi \).

**Theorem**

The Model Checking Problem for finite transition systems and LTL formulæ is decidable.

- Treat states \( q \in Q \) as letters in an alphabet.
- Language of \( (Q, \delta, I) \), \( L(Q, \delta, I) \) (or \( L(Q) \) for short) is set of runs in \( Q \)
- Language of \( \varphi \), \( L(\varphi) = \{ \sigma | \sigma \models \varphi \} \)
- Question: \( L(Q) \subseteq L(\varphi) \)?
- Same as: \( L(Q) \cap L(\neg \varphi) = \emptyset \)?

---

**How to Decide the Model Checking Problem?**

- How to answer \( L(Q) \cap L(\neg \varphi) = \emptyset \)?
- **Common approach**:
  - Build automaton \( A \) such the \( L(A) = L(Q) \cap L(\neg \varphi) \)
  - Are accepting states of \( A \) reachable? (Infinitely often?)
- How to build \( A \)?
  - One possible answer: Build a series of automata by recursion on structure of \( \neg \varphi \).
  - Another possible answer: Build an automaton \( B \) such \( L(B) = L(\neg \varphi) \); take \( A = B \times Q \)
- Will do at least one approach if time after Spin