An Introduction to Temporal Logics

©2001,2004 M. Lawford
Outline

• Motivation: Dining Philosophers

• Safety, Liveness, Fairness & Justice

• Kripke structures, LTS, SELTS, and Paths

• Linear Temporal Logic

• Branching Temporal Logics: CTL and CTL*

• Real-time Temporal Logics: RTTL, RTL, etc.
An Introduction to Temporal Logics

References:


Motivation:

- Want to be able to express & verify properties of system dynamics:
  - Safety (invariance): Nothing bad will happen
  - Liveness: Something good will happen

- Allows for abstract specification of properties without providing all the details

- Can express properties that are not expressible by defining 1 step transition relation (e.g. fairness)
Detailed Outline

- Motivation

- System Models
  - Kripke Structures
  - Labeled Transitions Systems (LTS)
  - State-Event Labeled Transition Systems (SELTS)
  - Duality of State & Event representations

- Temporal Logics
  - Propositional Logic
  - LTL - Linear Temporal Logic
  - CTL - Computational Tree Logic
    - CTL*

- LTL and CTL - What’s the difference?
  - Expressivity, Complexity, & Decidability
Motivation: Dining Philosophers & Deadlock

Abstraction of resource sharing problem common in many systems.

- \( n \) philosophers seated at round table with food in center
- \( n \) chop sticks, one between each pair
- Philosophers are either thinking or eating
- To eat a philosopher must use 2 chopsticks (the one to their left & one to their right)

Greedy heuristic: Hold on to any chop-stick until you get to eat.

Deadlock: When the system is prevented from taking any action (no transitions are possible since all enablement conditions are false).

Problem: System can deadlock (how?)
Motivation for Fairness

Less Greedy heuristic: Only pick up right chopstick if left present.

Assumptions:

- weak fairness: any transition that is continuously enabled eventually happens (i.e. philosopher who is eating will always eventually finish)

Still not enough!

- strong fairness: any transition that is enabled infinitely often will eventually occur. (If his/her two chop-sticks are available infinitely often, philosopher will eventually eat - and hence eat infinitely often.)
Motivation: Dining Philosophers & Livelock

Strong fairness assumption for “Less Greedy” heuristic still not enough to prevent individual starvation due to *livelock*.

**Livelock:** When system component is prevented from taking any action or a particular action (individual starvation).

Two can starve in $n = 4$ (4 philosophers) case if consecutive feedings allowed. How?

**a)** 1 starts eating, then 3.

**b)** 1 finishes, then starts feeding again before 3 finishes.

**c)** 3 finishes, then starts again before 1 finishes. . .

Even disallowing consecutive feedings for $n \geq 5$, one philosopher can still starve due “livelock”. How?
Motivation

Want to be able to express & verify properties of system dynamics:

**Safety**: Nothing bad will happen.

**Liveness**: Something good will happen.

**Fairness**: Independent processes will progress.

**Temporal logics:**

- Allows for formal abstract specification of above properties
- Can express properties that are not expressible by describing 1 step transition relation (e.g. fairness).
- Can be “effectively” model-checked for finite state systems

Predicate logic allows to reason about a state. Temporal logic allows to reason about sequences of states.
Kripke Structures

\[ \textbf{M} := \langle S, R, S_0, A, P \rangle \]

- \( S \) is a set of states
- \( R \subseteq S \times S \) is a transition relation (or equivalently \( R : S \to \mathcal{P}(S) \))
- \( S_0 \subseteq S \) is a set of initial states
- \( A \) is a set of atomic propositions (e.g. \( y = 1 \))
- \( P : S \to \mathcal{P}(A) \) labels each state with the set of atomic propositions satisfied by the state

is a **Kripke structure** (aka. labeled state transition graph)

A path in \( \textbf{M} \) is a sequence of states \( \pi \):

- \( \pi := s_0s_1 \ldots s_n \in S^+ \) and \( R(s_n) = \emptyset \) or,
- \( \pi := s_0s_1 \ldots \in S^\omega \)

such that \( s_0 \in S_0 \) and for all \( i \geq 0, (s_i, s_{i+1}) \in R \) in which case we write \( s_i \to s_{i+1} \).
Paths & Postfixes

Let $|\pi|$ be the length of the path $\pi$. Any path or computation $\pi$ in a Kripke structure satisfies the following:

i) Initialization: $s_0$ is an initial state of $M$.

ii) Succession: $0 \leq i < |\pi|$ implies $(s_i, s_{i+1}) \in R$ (i.e. $s_i \rightarrow s_{i+1}$ in $M$)

iii) Diligence: $\pi$ is finite, ending in state $s_n$ iff $R(s_n) = \emptyset$.

**Def:** The $k$th postfix of a path $\pi = s_0s_1\ldots$, denoted $\pi^k$ will be used to denote the $k$-shifted suffix of $\pi$, that is $\pi^k := s_{k}s_{k+1}\ldots$. 
Labeled Transition Systems (LTS)

\[ M := \langle S, \Sigma, R_\Sigma, S_0 \rangle \]

- \( S \) is a set of states
- \( \Sigma \) is a set of transition labels ("events")
- \( R_\Sigma = \{ \alpha^M \subseteq S \times S | \alpha \in \Sigma \} \) is a set of transition relations (or, equivalently, for each \( \alpha \in \Sigma \), \( \alpha^M : S \rightarrow \mathcal{P}(S) \))
- \( S_0 \subseteq S \) is a set of initial states

is a Labeled Transition System (LTS)

A path in \( M \) is a sequence of states and events \( \pi \):

- \( \pi := s_0 \xrightarrow{\alpha_1} s_1 \xrightarrow{\alpha_2} \ldots \xrightarrow{\alpha_{n-1}} s_n \) and 
  \((\forall \alpha \in \Sigma) \alpha^M(s_n) = \emptyset\), or

- \( \pi := s_0 \xrightarrow{\alpha_1} s_1 \xrightarrow{\alpha_2} \ldots \)

such that \( s_0 \in S_0 \) and for all \( i \geq 0, (s_i, s_{i+1}) \in \alpha_i^M \) in which case we write \( s_i \xrightarrow{\alpha_i} s_{i+1} \).
State Event Labeled Transition Systems (SELTs)

\[ M := \langle S, \Sigma, R_\Sigma, S_0, P \rangle \]

- where \( \langle S, \Sigma, R_\Sigma, S_0 \rangle \) is a LTS, and
- \( P : S \to \mathcal{P}(A) \) is a state output map,

is a State Event Labeled Transition System (SELTs)

A path in \( M \) is defined the same as for a LTS. Such paths in a transition system satisfying the “diligence” property are also known as maximal paths.
An SELTS Example

State Legend

\((u, v, x)\)
\([c_\alpha, c_\beta, c_\gamma]\)
Duality of State and Event Models

Claim 1: Any LTS has an equivalent Kripke structure representation.

Proof: For LTS \( M := \langle S, \Sigma, R_\Sigma, S_0 \rangle \) create Kripke structure \( M' := \langle S', R', S'_0, A', P' \rangle \):

Let \( S' := S \times \Sigma \). Then \((s_1, \alpha_1) \rightarrow (s_2, \alpha_2)\) in \( M' \) iff \((\exists s \in S)s_1 \xrightarrow{\alpha_1} s_2 \xrightarrow{\alpha_2} s\) in \( M \) defines \( R' \). Take

\[
S'_0 := \{ (s_0, \alpha_0) \in S' | s_0 \in S_0 \land \alpha_0^M(s_0) \neq \emptyset \}
\]

Let \( \eta \) be the next event variable. Take

\[
A' := \{ \eta = \alpha | \alpha \in \Sigma \}
\]

So \( P' : S' \rightarrow \mathcal{P}(A') \) is given by \((s, \alpha) \xmapsto{P'} (\eta = \alpha)\)

Corollary: Any SELTS has an equivalent Kripke structure representation.

Claim 2: Any Kripke structure has an equivalent LTS representation.
Linear Temporal Logic: Syntax

The definition of linear temporal logic formula adds two new operators $X$ and $U$, to the definition of a propositional formula.

**Def:** A *formula* is defined as follows:

1. If $\phi \in A \cup \{\bot, \top\}$ then $\phi$ *formula*.

2. If $\phi$ and $\psi$ are formulas, so are:

   $(\neg \phi), (\phi \land \psi), (\phi \lor \psi), (\phi \rightarrow \psi), (\phi \leftrightarrow \psi)$

3. If $\phi$ and $\psi$ are formulas, then so are:

   $X\phi$ and $\phi U \psi$
Linear Temporal Logic: Semantics

Def: (Satisfaction) For LTL formulas $\phi$, $\phi_1$ and $\phi_2$, $M$ a Kripke structure and $\pi := s_0s_1\ldots$, a path in $M$ then the satisfaction relation is defined as follows:

- If $\phi \in A \cup \{\bot, \top\}$, is an atomic proposition or logical constant, then $\pi \models \phi$ iff $s_0 \models \phi$ (i.e. $\phi \in P(s_0)$ or $\phi$ is $\top$)
- $\pi \models \phi_1 \lor \phi_2$ iff $\pi \models \phi_1$ or $\pi \models \phi_2$
- $\pi \models \phi_1 \land \phi_2$ iff $\pi \models \phi_1$ and $\pi \models \phi_2$
- $\pi \models \neg \phi$ iff $\pi \not\models \phi$
- $\pi \models X\phi$ iff $\pi^1$ exists and $\pi^1 \models \phi$
- $\pi \models \phi_1 U \phi_2$ iff $\pi \models \phi_2$, or $(\exists k > 0) \pi^k$ is defined, $\pi^k \models \phi_2$ and $(\forall i : 0 \leq i < k)\pi^i \models \phi_1$.

We say that state $s$ of $M$ satisfies formula $\phi$, written $M, s \models \phi$ iff for every path $\pi$ in $M$ starting at $s$, we have $\pi \models \phi$.

We say that $M \models \phi$ iff for every path $\pi$ in $M$ it is the case that $\pi \models \phi$
Derived Operators: F & G

Linear Temporal Logic (LTL) allows us to say:

• A formula will eventually be true on a path

• A formula will alway be true on a path

Consider the temporal formula $\mathcal{T} \mathcal{U} \phi$

Since $\mathcal{T}$ is true in every state, $\mathcal{T} \mathcal{U} \phi$ is satisfied by any path $\pi$ for which $(\exists k \geq 0) \pi^k \models \phi$ (i.e. EVENTUALLY $\phi$ is true in path $\pi$).

As an abbreviation for $\mathcal{T} \mathcal{U} \phi$ we write $F\phi$.

If $\phi$ is always true at every state in $\pi$, then it must be the case that $\neg \phi$ is never true. i.e. $\forall \pi \models \neg F \neg \phi$.

In this case we say that HENCEFORTH $\phi$ is true in $\pi$. As an abbreviation for $\neg F \neg \phi$ we write $G\phi$. 
Combining Temporal Operators

Let $\pi$ be an infinite path. By combining the $F$ and $G$ operators we can say:

- At a certain point, a formula is true at all future states of the path
  
  \[
  \pi \models FG\phi \iff (\exists k \geq 0)\pi^k \models G\phi \\
  \text{iff} \quad (\exists k \geq 0)(\forall i \geq k)\pi^i \models\phi
  \]

- A formula is true at infinitely many states on the path
  
  \[
  \pi \models GF\phi \iff (\forall k \geq 0)\pi^k \models F\phi \\
  \text{iff} \quad (\forall k \geq 0)(\exists i \geq k)\pi^i \models\phi
  \]
Fairness Formulas

Strong Fairness: $FG\phi_1 \rightarrow GF\phi_2$

E.g. For Dining philosophers, want paths to satisfy property:

$$FG(x_i = \text{Feed}) \rightarrow GF(x_i = \text{Think})$$

If a philosopher tries to feed forever, then he will always eventually be thinking. This simplifies to $\neg FG(x_i = \text{Feed})$ (i.e. He won’t succeed at feeding forever) for philosopher with two states.

Weak Fairness: $GF\phi_1 \rightarrow GF\phi_2$

$$GF(x_i = \text{Think}) \rightarrow GF(x_i = \text{Feed})$$

If a philosopher is thinking infinitely often, he will feed infinitely often.
Computational Tree Logic (CTL): Syntax

The definition of a CTL formula adds four new operators \( EX, AX, E(\cdot U \cdot) \) and \( A(\cdot U \cdot) \), to the definition of a propositional formula.

**Def:** A *formula* is defined as follows:

1. If \( \phi \in A \) or \( \phi \) is \( \top \) or \( \bot \) then \( \phi \) formula.

2. If \( \phi \) and \( \psi \) are formulas, so are:

\[
(\neg \phi), (\phi \land \psi), (\phi \lor \psi), (\phi \rightarrow \psi), (\phi \leftrightarrow \psi)
\]

3. If \( \phi \) and \( \psi \) are formulas, then so are:

\[
EX\phi, AX\phi, \text{ and } E(\phi U \psi), A(\phi U \psi)
\]
**CTL: Semantics**

**Def:** (Satisfaction) For temporal formulas $\phi$, $\phi_1$ and $\phi_2$, $M$ a Kripke structure and $s_0 \in S$ a state of $M$, the satisfaction relation $\models$ is defined as follows:

- If $\phi \in A \cup \{\bot, T\}$, is an atomic proposition or logical constant, then $M, s_0 \models \phi$ iff $s_0 \models \phi$ (i.e. $\phi \in P(s_0)$ or $\phi$ is $T$)

- $M, s_0 \models \phi_1 \lor \phi_2$ iff $M, s_0 \models \phi_1$ or $M, s_0 \models \phi_2$

- $M, s_0 \models \phi_1 \land \phi_2$ iff $M, s_0 \models \phi_1$ and $M, s_0 \models \phi_2$

- $M, s_0 \models \neg \phi$ iff $M, s_0 \not\models \phi$

- $M, s_0 \models EX \phi$ iff $(\exists s' \in S)s_0 \rightarrow s'$ and $M, s' \models \phi$

- $M, s_0 \models AX \phi$ iff $(\forall s' \in S)$ if $s_0 \rightarrow s'$ then $M, s' \models \phi$
**CTL: Semantics** (cont.)

- $M, s_0 \models E(\phi_1 \cup \phi_2)$ iff
  
  - $M, s_0 \models \phi_2$, or
  
  - $(\exists \pi = s_0 \rightarrow s_1 \rightarrow \ldots s_n \rightarrow \ldots)$, a path in $M$ s.t. $(\exists k > 0)$, $M, s_k \models \phi_2$, and $(\forall i : 0 \leq i < k)M, s_i \models \phi_1$.

- $M, s_0 \models A(\phi_1 \cup \phi_2)$ iff
  
  - $M, s_0 \models \phi_2$, or
  
  - $(\forall \pi = s_0 \rightarrow s_1 \rightarrow \ldots s_n \rightarrow \ldots)$, paths in $M$,
    
    * $(\exists k > 0)$, $M, s_k \models \phi_2$, and $(\forall i : 0 \leq i < k)M, s_i \models \phi_1$

    * $\pi = s_0 \rightarrow s_1 \rightarrow \ldots s_n$ is a finite path and $(\forall i : 0 \leq i \leq n)M, s_i \models \phi_1$. 

22
Expressivity of LTL and CTL

A logic is said to be more expressive than another if it can express (say) more things.

In terms of expressivity, LTL and CTL are not comparable in the sense that each logic can say things that the other cannot, e.g.

- LTL cannot express the existence of a path like CTL can (e.g. \(EX\phi\))

- CTL cannot express fairness constraints such as the LTL formula

\[GF(\eta = \text{tick}) \rightarrow GF(\eta = \beta)\]

This motivates the creation of CTL*, a logic that is more expressive than both LTL and CTL.
**CTL*: Syntax**

In terms of expressivity CTL* is a superset of both LTL and CTL.

A *state formula* is any formula of the form:

\[ \phi ::= p | F | (\neg \phi) | (\phi \land \phi) | A[\alpha] | E[\alpha] \]

where \( p \) is any atomic proposition and \( \alpha \) is a path formula and

A *path formula* is any formula of the form:

\[ \alpha ::= \phi | (\neg \alpha) | (\alpha \land \alpha) | \alpha U \alpha | \alpha X \alpha \]

where \( \phi \) is any state formula.
Real Time Temporal Logic (RTTL)

Assume we are dealing with a SELTS $M$.

Consider path:

$$
\pi := s_0 \xrightarrow{\alpha_1} s_1 \xrightarrow{\alpha_2} \ldots
$$

For an event $\alpha \in \Sigma$, define

$$
\#\alpha(\pi, k) = \begin{cases} 
\text{number of } \alpha \text{'s from } s_0 \text{ and } s_k \\
\text{undefined if } k > |\pi|
\end{cases}
$$

- $\pi \models F_1 \mathcal{U}_{[l,u]} F_2$ iff $\exists k \geq 0$ such that $\pi^k$ is defined, $\pi^k \models F_2$ and $\forall i, 0 \leq i < k, \pi^i \models F_1$ and $l \leq \#\alpha(\pi, k) \leq u$.

If we have a distinguished event $\text{tick}$ that represents the tick of a global clock, then

$$
\pi \models F_1 \mathcal{U}_{[l,u]}^{\text{tick}} F_2
$$

iff path $\pi$ satisfies $F_1$ until $F_2$ between the $l$th and $u + 1$th $\text{tick}$ transition.