Introduction to Linear-Time Temporal Logic
Outline

• Motivation for TL in general
• Types of properties to be expressed in TL
• Structures on which LTL formulas are evaluated
• Syntax & semantics of LTL formulas
• Example LTL identities
• Example specifications
Why Temporal Logic?

- Classical logic is good for describing static structures: a fixed world
- Most computing systems are dynamic: worlds change over time
- Introduce **temporal operators** to describe how worlds change over time w/o explicitly referring to time
Why Temporal Logic?

• Want to be able to specify and verify properties about how a system changes over time (world = system state)

• Properties of interest include
  – Safety: “Nothing bad will happen”
  – Liveness: “Something good will happen”
  – Fairness: Independent subsystems make progress

• Properties not directly expressible as constraints on a 1-step transition relation
Resource sharing and deadlock

• Dining philosophers: abstracts a resource sharing paradigm
  – \(n \geq 5\) philosophers alternately think and eat at a round table
  – \(n\) chopsticks, one between each pair
  – a philosopher needs the chopstick on the left and on the right to eat

• Greedy heuristic:
  – Each phil repeatedly: picks up an available chopstick and holds onto it; when other is available, picks it up and eats; puts both chopsticks

• **Deadlock**: The system should not reach a state in which no further action is possible
Greedy algorithm: Deadlock

CSE 814  Introduction to LTL  6
Resource sharing and starvation

• Less greedy heuristic:
  – pick up right chopstick only if left is also available

• Prevents deadlock

• *Livelock, Starvation*: When a subsystem is prevented from taking any action because of resource contention
Less Greedy algorithm: Starvation

CSE 814

Introduction to LTL

STARVATION!
Fairness Assumptions

• **Weak Fairness**: Any action that is continuously enabled will eventually occur
  – Guarantees that a philosopher holding both chopsticks eventually puts them down
  – Does not prevent previous scenario for less greedy philosophers

• **Strong Fairness**: Any action that is enabled infinitely often will eventually occur
  – Prevents previous scenario (since picking up right fork is enabled infinitely often)
  – But is it enough to guarantee all philosophers eat infinitely often?
Why Temporal Logic?

Allows specification of properties such as

- **Safety**: Nothing bad ever happen
  - e.g., no deadlock
  - e.g., no violation of mutual exclusion

- **Liveness**: Something good eventually happens
  - e.g., if a chopstick is held, it is eventually put down
  - e.g., a program eventually terminates

- **Fairness**: Describe assumptions necessary to guarantee that a subsystem makes progress
  - e.g., if a philosopher’s chopstick is available infinitely often, the philosopher will eventually get to pick it up (strong fairness)
Temporal Logics

• Introduce *temporal operators* to describe how worlds change over time w/o explicitly referring to time

• Vary in how they view time
  – Linear time: quantify over the worlds in an execution path

  ![Linear time diagram]

  – Branching time: quantify over both execution paths and worlds

  ![Branching time diagram]

• Underlying both: Computation abstracted as a finite-state labeled transition system (LTS)
Propositions and States

• LTL is defined relative to a set of primitive propositions, $\mathcal{P}$

• Example primitive proposition:
  – $\text{wait}_{i,L}$ “phil $i$ is waiting to pick up the left fork”
  – $\text{eat}_i$ “phil $i$ is eating”
  – $\text{turn}_{i,j}$ “it is phil $i$’s turn to use fork $j$”
  – $\text{inCS}_A$ “process A is executing in a critical section”
  – $x = 0$ “the value of variable $x$ is 0”
  – $p@1$ “process $p$ is at line 1”

• A state, $s$, is defined by the set of primitive propositions that are true in that state
  – i.e., $s \subseteq \mathcal{P}$
LTL Temporal Operators

- Quantify over the states in a sequence of states.

- Informally, given PLTL formulas, $f$ and $g$
  - $\bigcirc f$  \hspace{1cm} \text{Next } f: \text{“} f \text{ is true at the next state”}
  - $\square f$  \hspace{1cm} \text{Henceforth } f: \text{“} f \text{ is true at all future states”}
  - $\Diamond f$  \hspace{1cm} \text{Eventually } f: \text{“} f \text{ is true at some future state”}
  - $f \mathbin{U} g$  \hspace{1cm} \text{f until } g: \text{“} g \text{ is true at some future state and } f \text{ is true at each state until then”}

- Alternate notations:
  - X for “Next” instead of circle, G for “Henceforth” instead of a box
  - F for “Eventually” instead of a diamond
Syntax of LTL

• The set of well-formed LTL formulas (relative to $\mathcal{P}$) is defined inductively by the following formation rules:

<table>
<thead>
<tr>
<th>Propositions</th>
<th>$p$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean operators</td>
<td>$\neg f$, $f \land g$, $f \lor g$, $f \Rightarrow g$</td>
</tr>
<tr>
<td>Temporal operators</td>
<td>$\Diamond f$, $\Box f$, $f \lor g$</td>
</tr>
</tbody>
</table>

If $p \in \mathcal{P}$ and $f$ and $g$ are LTL formulas, then the following are LTL formulas:
Semantics of LTL

The semantics of an LTL formula is defined by a satisfaction relation, $\models \subseteq (\mathbb{N} \rightarrow 2^P) \times \mathbb{N} \times LTL$.

For $\sigma : \mathbb{N} \rightarrow 2^P$, $j \in \mathbb{N}$, and $f \in LTL$, we write $(\sigma, j) \models f$ to mean $(\sigma, j, f) \in \models$.

Terminology:
- $f$ holds at position $j$ of $\sigma$
- $\sigma$ satisfies $f$ at $j$
- $\sigma$ satisfies $f$ at $\sigma(j)$
Semantics of LTL

Proposition $p$:

$$(\sigma, j) \models p \text{ iff } p \in \sigma(j)$$

Next $f$:

$$(\sigma, j) \models \circ f \text{ iff } (\sigma, j + 1) \models f$$
Semantics of LTL

Henceforth $f$:

$$(\sigma, j) \models \square f \text{ iff } \forall k \cdot (k \geq j \Rightarrow (\sigma, k) \models f)$$

Eventually $f$:

$$(\sigma, j) \models \Diamond f \text{ iff } \exists k \cdot (k \geq j \land (\sigma, k) \models f)$$
**Semantics of LTL**

$f$ until $g$:

\[(\sigma, j) \models f \mathbin{\mathbf{U}} g \iff \exists k \cdot (k \geq j \land (\sigma, k) \models g \land (\forall i \cdot (j \leq i < k \Rightarrow (\sigma, i) \models f)))\]

\[
\begin{array}{cccccccc}
\text{f} & \text{f} & \text{f} & \text{f} & \text{f} & \text{f} & \text{g} \\
0 & 1 & \cdots & j & j+1 & \cdots & k & \cdots
\end{array}
\]

*Strong until* -- because $g$ must be satisfied in the future
Semantics of LTL

Boolean operators defined in the usual manner

\[(\sigma, j) \models \neg f \iff \neg((\sigma, j) \models f)\]

\[(\sigma, j) \models f \land g \iff (\sigma, j) \models f \land (\sigma, j) \models g\]

\[(\sigma, j) \models f \lor g \iff (\sigma, j) \models f \lor (\sigma, j) \models g\]

\[
\ldots
\]
Some LTL identities

• Henceforth and Eventually are duals:
  \[ \square f \iff \neg (\Diamond \neg f) , \quad \Diamond f \iff \neg (\square \neg f) \]

• Henceforth and Eventually are idempotent:
  \[ \square \square f \iff \square f , \quad \Diamond \Diamond f \iff \Diamond f \]

• Henceforth, Eventually are derivable from Until
  \[ \Diamond f \iff true \ U f , \quad \square f \iff \neg (true \ U \neg f) \]

• Waits for (weak until) operator, \( \mathcal{W} \), can be defined
  \[ f \mathcal{W} g \iff f \ U g \lor \square f \]
Some LTL identities

• Distributive properties
  $\Box (f \land g) \Leftrightarrow \Box f \land \Box g$
  $\Diamond (f \lor g) \Leftrightarrow \Diamond f \lor \Diamond g$
  $\Diamond (f \lor g) \Leftrightarrow (\Diamond f) \lor (\Diamond g)$
  $\Diamond (f \land g) \Leftrightarrow (\Diamond f) \land (\Diamond g)$
  $\Box (f \lor g) \Leftrightarrow (\Box f) \lor (\Box g)$

• Many other identities …
Some example specifications

Suppose $inCS_X$ denotes “process $X$ is in the critical section”

- Process A and B are never both in their critical sections at the same time (mutual exclusion)
  \[\Box(\neg inCS_A \lor \neg inCS_B)\]

- Neither process monopolizes the critical section
  \[\Box(inCS_A \Rightarrow \Diamond \neg inCS_A)\]
  \[\Box(inCS_B \Rightarrow \Diamond \neg inCS_B)\]
Some example specifications

• Given primitive proposition:
  - \( wait_{i,R} \) “phil i is waiting to pick up the right fork”
  - \( eat_i \) “phil i is eating”
  - \( turn_{i,j} \) “it is phil i’s turn to use fork j”

• A philosopher waits continuously until she eats

\[
\bigwedge_{i=1,\ldots,5} \square ( wait_{i,R} \rightarrow wait_{i,R} \cup eat_i )
\]

Corollary: No (hungry) philosopher starves
Some example specifications

- The philosophers take turns “fairly”
  \[ \bigwedge_{i=1,...,5} \Box (eat_i \Rightarrow \Diamond (\neg turn_{i,i} \land \neg turn_{i,(i+1)_{mod5}})) \]

- \[ \bigwedge_{i=1,...,5} \Box (turn_{i,(i+1)_{mod5}} \land wait_{i,L} \Rightarrow eat_{(i+1)_{mod5}} \cup (\neg eat_{(i+1)_{mod5}} \cup eat_i)) \]

- \[ \bigwedge_{i=1,...,5} \Box (turn_{i,i} \land wait_{i,R} \Rightarrow eat_{(i-1)_{mod5}} \cup (\neg eat_{(i-1)_{mod5}} \cup eat_i)) \]
Exercise

• Given primitive proposition:
  – $begin_p$ “control is at the start of procedure $p$”
  – $pre_p$ “procedure $p$’s precondition is true”
  – $end_p$ “control is at the end of procedure $p$”
  – $post_p$ “procedure $p$’s post condition is true”
  – $in_p$ “control is in procedure $p$”

• Write a temporal logic formula that expresses partial correctness

• Write a temporal logic formula that expresses total correctness
Acknowledgements

• Formal Methods: Lecture III: Linear Time Temporal Logic, A. Artale, Free University of Bolzano, [http://web.iitd.ac.in/~sumeet/slide3.pdf](http://web.iitd.ac.in/~sumeet/slide3.pdf)
• An Introduction to Temporal Logics, M. Lawford, McMaster University, [https://www.cas.mcmaster.ca/~lawford/.../model.pdf](https://www.cas.mcmaster.ca/~lawford/.../model.pdf)