An Introduction to Temporal Logics

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:

- E.M. Clarke, E.A. Emerson, and A.P. Sistla, "Automatic verification of finite state concurrent systems using temporal logic specifications." *ACM Trans on Prog Languages & Systems, Vol. 8, No. 2*, April 1986, pp. 244-263.

- Z. Manna and A. Pnueli. *The Temporal Logic of Reactive and Concurrent Systems*. Springer-Verlag, New York, 1992.

- A. Arnold, *Finite Transition Systems*. Prentice Hall, 1994.

- J.S. Ostroff. *Temporal Logic for Real-Time Systems*. Research Studies Press/Wiley, Taunton, UK, 1989.

- E.A. Emerson et al. "Quantitative temporal reasoning." *Real-Time Systems, No. 4*, pp. 331-352, 1992.

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 chop-stick 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

$$\mathbf{M} := \langle S, R, S_0, A, P\rangle$$

- 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* $\mathbf{M}$ is a sequence of states π:

- $\pi := s_0 s_1 \ldots s_n \in S^+$ and $R(s_n) = \emptyset$ or,
- $\pi := s_0 s_1 \ldots \in S^\omega$

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* π. Any path or *computation* π in a Kripke structure satisfies the following:

i) Initialization: s_0 is an initial state of $\mathbf{M}$.

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

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

Def: The kth *postfix of a path* $\pi = s_0 s_1 \ldots$, denoted π^k will be used to denote the k-shifted suffix of π, that is $\pi^k := s_k s_{k+1} \ldots$.

Labeled Transition Systems (LTS)

$$\mathbf{M} := \langle S, \Sigma, R_\Sigma, S_0 \rangle$$

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

is a *Labeled Transition System* (LTS)

A *path in* $\mathbf{M}$ is a sequence of states and events π:

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

such that $s_0 \in S_0$ and for all $i \geq 0, (s_i, s_{i+1}) \in \alpha_i^{\mathbf{M}}$ in which case we write $s_i \stackrel{\alpha_i}{\to} s_{i+1}$.

State Event Labeled Transition Systems (SELTS)

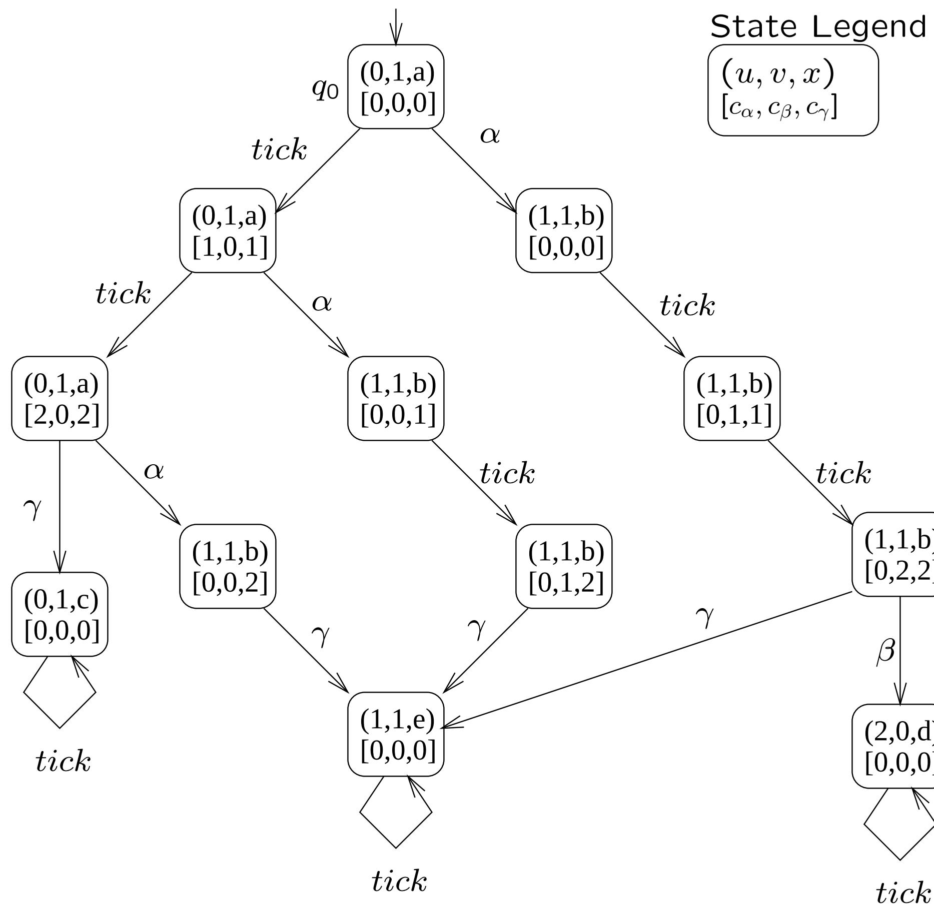
$$\mathrm{M} := \langle S, \Sigma, R_\Sigma, S_0, P\rangle$$

- 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



Duality of State and Event Models

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

Proof: For LTS $\mathbf{M} := \langle S, \Sigma, R_\Sigma, S_0 \rangle$ create Kripke structure $\mathbf{M}' := \langle S', R', S'_0, A', P' \rangle$:

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

$$S'_0 := \{(s_0, \alpha_0) \in S' | s_0 \in S_0 \wedge \alpha_0^{\mathbf{M}}(s_0) \neq \emptyset\}$$

Let η be the *next event* variable. Take

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

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

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 ϕ *formula*.
2. If ϕ and ψ are formulas, so are:

$$(\neg\phi), (\phi \wedge \psi), (\phi \vee \psi), (\phi \to \psi), (\phi \leftrightarrow \psi)$$

3. If ϕ and ψ are formulas, then so are:

$$\mathsf{X}\phi \text{ and } \phi\mathsf{U}\psi$$

Linear Temporal Logic: Semantics

Def: (Satisfaction) For LTL formulas ϕ, ϕ_1 and ϕ_2, $\mathbf{M}$ a Kripke structure and $\pi := s_0 s_1 \ldots$, a path in $\mathbf{M}$ 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 ϕ is $\top$)
- $\pi \models \phi_1 \vee \phi_2$ iff $\pi \models \phi_1$ or $\pi \models \phi_2$
- $\pi \models \phi_1 \wedge \phi_2$ iff $\pi \models \phi_1$ and $\pi \models \phi_2$
- $\pi \models \neg\phi$ iff $\pi \not\models \phi$
- $\pi \models \mathsf{X}\phi$ iff π^1 exists and $\pi^1 \models \phi$
- $\pi \models \phi_1 \mathsf{U} \phi_2$ iff $\pi \models \phi_2$, or $(\exists k > 0)$ π^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 $\mathbf{M}$ satisfies formula ϕ, written $\mathbf{M}, s \models \phi$ iff for every path π in $\mathbf{M}$ starting at s, we have $\pi \models \phi$.

We say that $\mathbf{M} \models \phi$ iff for every path π in $\mathbf{M}$ 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 $\top \mathsf{U} \phi$

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

As an abbreviation for $\top \mathsf{U} \phi$ we write $\mathsf{F}\phi$.

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

In this case we say that *HENCEFORTH* ϕ is true in π. As an abbreviation for $\neg\mathsf{F}\neg\phi$ we write $\mathsf{G}\phi$.

Combining Temporal Operators

Let π 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

$$\begin{array}{lll} \pi \models \mathsf{FG}\phi & \text{iff} & (\exists k \geq 0)\pi^k \models \mathsf{G}\phi \\ & \text{iff} & (\exists k \geq 0)(\forall i \geq k)\pi^i \models \phi \end{array}$$

- A formula is true at infinitely many states on the path

$$\begin{array}{lll} \pi \models \mathsf{GF}\phi & \text{iff} & (\forall k \geq 0)\pi^k \models \mathsf{F}\phi \\ & \text{iff} & (\forall k \geq 0)(\exists i \geq k)\pi^i \models \phi \end{array}$$

Fairness Formulas

Strong Fairness: $\mathsf{FG}\phi_1 \rightarrow \mathsf{GF}\phi_2$

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

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

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

Weak Fairness: $\mathsf{GF}\phi_1 \rightarrow \mathsf{GF}\phi_2$

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

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 $EX, AX, E(\cdot \mathsf{U} \cdot)$ and $A(\cdot \mathsf{U} \cdot)$, to the definition of a propositional formula.

Def: A *formula* is defined as follows:

1. If $\phi \in A$ or ϕ is $\top$ or $\bot$ then ϕ formula.
2. If ϕ and ψ are formulas, so are:

   $$(\neg\phi), (\phi \wedge \psi), (\phi \vee \psi), (\phi \to \psi), (\phi \leftrightarrow \psi)$$

3. If ϕ and ψ are formulas, then so are:

   $$EX\phi, AX\phi, \text{ and } E(\phi \mathsf{U} \psi), A(\phi \mathsf{U} \psi)$$

CTL: Semantics

Def: (Satisfaction) For temporal formulas ϕ, ϕ_1 and ϕ_2, $\mathbf{M}$ a Kripke structure and $s_0 \in S$ a state of $\mathbf{M}$, the satisfaction relation $\models$ is defined as follows:

- If $\phi \in A \cup \{\bot, \top\}$, is an atomic proposition or logical constant, then $\mathbf{M}, s_0 \models \phi$ iff $s_0 \models \phi$ (i.e. $\phi \in P(s_0)$ or ϕ is $\top$)
- $\mathbf{M}, s_0 \models \phi_1 \vee \phi_2$ iff $\mathbf{M}, s_0 \models \phi_1$ or $\mathbf{M}, s_0 \models \phi_2$
- $\mathbf{M}, s_0 \models \phi_1 \wedge \phi_2$ iff $\mathbf{M}, s_0 \models \phi_1$ and $\mathbf{M}, s_0 \models \phi_2$
- $\mathbf{M}, s_0 \models \neg\phi$ iff $\mathbf{M}, s_0 \not\models \phi$
- $\mathbf{M}, s_0 \models EX\phi$ iff $(\exists s' \in S) s_0 \rightarrow s'$ and $\mathbf{M}, s' \models \phi$
- $\mathbf{M}, s_0 \models AX\phi$ iff $(\forall s' \in S)$ if $s_0 \rightarrow s'$ then $\mathbf{M}, s' \models \phi$

CTL: Semantics (cont.)

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

- $\mathbf{M}, s_0 \models A(\phi_1 \mathsf{U} \phi_2)$ iff
  - $\mathbf{M}, s_0 \models \phi_2$, or
  - $(\forall \pi = s_0 \rightarrow s_1 \rightarrow \ldots s_n \rightarrow \ldots)$, paths in $\mathbf{M}$,
    * $(\exists k > 0)$, $\mathbf{M}, s_k \models \phi_2$, and $(\forall i : 0 \leq i < k) \mathbf{M}, s_i \models \phi_1$
    * $\pi = s_0 \rightarrow s_1 \rightarrow \ldots s_n$ is a finite path and $(\forall i : 0 \leq i \leq n) \mathbf{M}, s_i \models \phi_1$.

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

$$\mathsf{GF}(\eta = tick) \rightarrow \mathsf{GF}(\eta = \beta)$$

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:

$$\phi ::= p|\top|(\neg\phi)|(\phi \wedge \phi)|A[\alpha]|E[\alpha]$$

where p is any atomic proposition and α is a path formula and

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

$$\alpha ::= \phi|(\neg\alpha)|(\alpha \wedge \alpha)|\alpha \mathsf{U} \alpha|\mathsf{X}\alpha$$

where ϕ is any state formula.

Real Time Temporal Logic (RTTL)

Assume we are dealing with a SELTS $\mathbb{M}$.

Consider path:

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

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

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

- $\pi \models F_1 \mathcal{U}^{\alpha}_{[l,u]} F_2$ iff $\exists k \geq 0$ such that π^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 *tick* that represents the tick of a global clock, then

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

iff path π satisfies F_1 until F_2 between the lth and $u+1$th *tick* transition.