LTL

Linear Time Temporal Logic
Kripke Structure

- $K = \langle S, R, L \rangle$
  
  $S$: set of states (may be infinite)
  
  $R$: transition relation between states
  
  $R \subseteq S \times S$
  
  $L$: map from states to sets of propositional symbols
  
  $L(s)$ denotes the set of propositional symbols that hold at state $s \in S$
Kripke Structure

- $K = \langle S, R, L \rangle$
- $G = \langle S, R \rangle$

directed graph
Kripke Structure

• $K = \langle S, R, L \rangle$
• $L : S \rightarrow 2^{\text{Atom}}$

\text{Atom} : the set of all prop. sym.

\text{Atom} = \{P, Q\}
Infinite Sequence of States --- Execution Path

- \( \pi = \pi_0, \pi_1, \pi_2, \ldots \)
  \[ \pi_i \in S \quad (\forall i \geq 0) \]
  \[ R(\pi_i, \pi_{i+1}) \quad (\forall i \geq 0) \]

- suffix
  \[ \pi^i = \pi_i, \pi_{i+1}, \pi_{i+2}, \ldots \]
  Often denoted by \( s_i \)

\( (\pi_i, \pi_{i+1}) \in R \)
Formula

\( \varphi, \psi ::= P \quad \text{prop. symbol} \)

| \( \neg \varphi \) | negation |
| \( \varphi \land \psi \) | conjunction |
| \( \varphi \lor \psi \) | disjunction |
| \( \bigcirc \varphi \) | \( (X\varphi) \) |
| \( \Box \varphi \) | \( (G\varphi) \) |
| \( \Diamond \varphi \) | \( (F\varphi) \) |

\( \text{until is not considered here} \)
Semantics

\[ \pi \models P \iff P \in L(\pi_0) \]
\[ \pi \models \neg \phi \iff \text{not } \pi \models \phi \]
\[ \pi \models \phi \land \psi \iff \pi \models \phi \text{ and } \pi \models \psi \]
\[ \pi \models \phi \lor \psi \iff \pi \models \phi \text{ or } \pi \models \psi \]
\[ \pi \models \Diamond \phi \iff \pi^i \models \phi \text{ for some } i \geq 0 \]
\[ \pi \models \Box \phi \iff \pi^i \models \phi \text{ for any } i \geq 0 \]

\[ \phi \text{ holds in } \pi \]
\[ \pi \models \phi \text{ --- } \pi \text{ is a model of } \phi \]

The semantics of \( \Box \) and \( \Diamond \) is different from that of CTL.
$\pi \models \bigcirc \Box P$

$\pi^1 \models \Box P$

$\pi \models \bigcirc \Box P$

$P \land Q$

$\Diamond (P \land Q)$

$\bigcirc (P \land Q)$

$\Diamond (P \land Q) \land \bigcirc \Box P$
Which formula holds in this path?

1. □P
2. □¬P
3. ◇(P∧Q)
4. ◇(P∧¬Q)
Does $\Diamond \Box Q$ hold?

1. Yes
2. No
Does $\lozenge \square P$ hold?

1. Yes
2. No
Does $\Box \Diamond P$ hold?

1. Yes
2. No
\[
\Box \Diamond P
\]

- \( \pi \models \Box \Diamond P \) implies \( \pi \models \Diamond P \), so there exists \( i \) such that \( \pi^i \models P \)
- \( \pi \models \Box \Diamond P \) implies \( \pi^{i+1} \models \Diamond P \), so there exists \( j > i \) such that \( \pi^j \models P \)
- Consequently, there exist an infinite number of \( i \) such that \( \pi^i \models P \)
- Conversely, if there exist an infinite number of \( i \) such that \( \pi^i \models P \), then \( \pi \models \Box \Diamond P \) holds
Expressing Fairness

• Let E denote that a certain process is executable, and let R denote that the process is executed next

• Unconditional fairness
  \[ \Box \Diamond R \]

• Weak fairness
  \[ \Box \Diamond (\neg E \lor R) \]
  \[ \Box \Diamond (E \supset R) \]

• Strong fairness
  \[ \neg \Box \Diamond E \lor \Box \Diamond R \]
  \[ \Box \Diamond E \supset \Box \Diamond R \]

Consider their negation
Model Checking in LTL

• Given a formula $\varphi_0$, a Kripke structure $K$, and its initial state $s$, does there exist a path $\pi$ starting from $s$ such that $\pi \models \varphi_0$?
  – If you want to verify that $\varphi_0$ holds w.r.t. any path $\pi$ starting from $s$ in $K$, then you should negate $\varphi_0$ and solve the model checking problem on $\neg \varphi_0$
Example: $\square(P \supset \Diamond B)$

• Negate it: $\Diamond (P \land \square \neg B)$
  – Check negation!

• Does there exist a path $\pi$ starting from $s$ such that $\pi \models \Diamond (P \land \square \neg B)$?

• If there is no such path from $s$, then $\pi \models \square (P \supset \Diamond B)$ holds for any path $\pi$ from $s$

• Write $\neg B$ as $Q$ and consider $\Diamond (P \land \square Q)$
\( \Diamond (P \land \Box Q) \)

- Does there exist a path \( \pi \) starting from \( s \) such that \( \pi \models \Diamond (P \land \Box Q) \) ?
In the case of a finite Kripke structure
\( \Diamond (P \land \Box Q) \)

- Does there exist a path \( \pi \) starting from \( s \) such that \( \pi \models \Diamond (P \land \Box Q) \) ?

Equivalent to

- Do there exist a state \( e \) that is reachable from \( s \) such that \( P \in L(e) \), and a path \( \pi \) starting from \( e \) such that \( \pi_i = \pi_j \) for some \( i > 0 \) and \( j > i \), and \( Q \in L(\pi_k) \) for all \( k < j \) ?
Under (Unconditional) Fairness

- Paths should contain execution of each process infinitely often
Each process is executed infinitely often.
Each process is executed at least once in the loop.

In the case of a finite Kripke structure.
Another Approach

- $K = \langle S, R, L \rangle$
- Let $K'$ be $\langle S', R', L' \rangle$
  - $S' = \{0\} \times S \cup \{1\} \times (\{s \in S \mid Q \in L(s)\})$
  - $R' = \{((0,s),(0,s')) \mid (s,s') \in R\} \cup \{((0,s),(1,e)) \mid (s,e) \in R, P \in L(e), Q \in L(e)\} \cup \{((1,s),(1,s')) \mid (s,s') \in R, Q \in L(s), Q \in L(s')\}$
  - $L'((0,s)) = L(s)$
  - $L'((1,s)) = L(s)$
\(\Diamond (P \land \Box Q)\) again

- Does there exist a path \(\pi\) in \(K\) starting from \(s\) such that \(\pi \models \Diamond (P \land \Box Q)\) ?

Equivalent to

- Does there exist an path in \(K'\) starting from \((0, s)\) that visits states of the form \((1, s')\) infinitely often?
Start here

Eventually reach here
General Strategy

• Given $\varphi_0$, construct a state transition system (called $\omega$-automaton) that characterizes $\varphi_0$

• Make the synchronous product of $K$ and the $\omega$-automaton
  – This corresponds to $K'$

• Check whether a certain kind of loop exists in the synchronous product

• Refer to the slides for the last year
until

\( \pi \models \varphi \text{ until } \psi \)

iff \( \pi^i \models \psi \) for some \( i \geq 0 \)

and

\( \pi^j \models \varphi \) for any \( j < i \)
SPIN

- One of the most popular model checkers
- The target system is described in Promela, a CSP-like concurrent language
- The property is defined in LTL and translated into a NEVER clause of Promela
- The synchronous product is verified
- Applied to verify protocols, algorithms, (software) designs, etc.

http://spinroot.com/spin/whatispin.html