Summary

Introduced transition systems

- a set of states $S$
- a set of labels $A$
- a set of transitions: $s \xrightarrow{a} s'$ where $s, s' \in S$, $a \in A$
Summary

Or Kripke structure where labels appear at states ("colours")
Summary

Modal logic

\[ \Phi ::= \texttt{tt} | \texttt{ff} | \Phi_1 \land \Phi_2 | \Phi_1 \lor \Phi_2 | [K] \Phi | \langle K \rangle \Phi \]

Semantics

\[ E \models \Phi \]

\[
E \models \texttt{tt} \\
E \not\models \texttt{ff} \\
E \models \Phi \land \Psi \iff E \models \Phi \text{ and } E \models \Psi \\
E \models \Phi \lor \Psi \iff E \models \Phi \text{ or } E \models \Psi \\
E \models [K] \Phi \iff \forall F \in \{ E' : E \xrightarrow{a} E' \text{ and } a \in K \}. F \models \Phi \\
E \models \langle K \rangle \Phi \iff \exists F \in \{ E' : E \xrightarrow{a} E' \text{ and } a \in K \}. F \models \Phi
\]
Summary

Kripke model = Kripke structure + $L : \text{Colours} \rightarrow S$.

Modal Logic: Syntax

\[ \Phi ::= p \mid \neg p \mid \Phi_1 \land \Phi_2 \mid \Phi_1 \lor \Phi_2 \mid [\neg] \Phi \mid \langle \neg \rangle \Phi \]
Summary

Semantics

\[ E \models p \quad \text{iff} \quad E \in L(p) \]
\[ E \models \neg p \quad \text{iff} \quad E \notin L(p) \]
\[ E \models \Phi \land \Psi \quad \text{iff} \quad E \models \Phi \quad \text{and} \quad E \models \Psi \]
\[ E \models \Phi \lor \Psi \quad \text{iff} \quad E \models \Phi \quad \text{or} \quad E \models \Psi \]
\[ E \models [\neg] \Phi \quad \text{iff} \quad \forall F \in \{ E' : E \to E' \}. \ F \models \Phi \]
\[ E \models \langle \neg \rangle \Phi \quad \text{iff} \quad \exists F \in \{ E' : E \xrightarrow{\alpha} E' \}. \ F \models \Phi \]
Summary: Bisimulation on Transition Systems

A binary relation $B$ between states of a transition system is a bisimulation provided that, whenever $(E, F) \in B$ and $a \in A$,

- if $E \xrightarrow{a} E'$ then $F \xrightarrow{a} F'$ for some $F'$ such that $(E', F') \in B$, and

- if $F \xrightarrow{a} F'$ then $E \xrightarrow{a} E'$ for some $E'$ such that $(E', F') \in B$

Two states $E$ and $F$ are bisimulation equivalent, $E \sim F$, if there is a bisimulation relation $B$ such that $(E, F) \in B$. 
Summary: Bisimulation on Kripke Models

A binary relation $B$ between states of a Kripke model is a \textit{bisimulation} provided that, whenever $(E, F) \in B$

- for all colours $p$, $E \in L(p)$ iff $F \in L(p)$

- if $E \xrightarrow{} E'$ then $F \xrightarrow{} F'$ for some $F'$ such that $(E', F') \in B$, and

- if $F \xrightarrow{} F'$ then $E \xrightarrow{} E'$ for some $E'$ such that $(E', F') \in B$

Two states $E$ and $F$ are \textit{bisimulation equivalent}, $E \sim F$, if there is a bisimulation relation $B$ such that $(E, F) \in B$. 
Summary: Invariance

- \( E \equiv F \) if for all modal \( \Phi \), \( E \models \Phi \) iff \( F \models \Phi \).

(\( E \) and \( F \) have the same modal properties.)

- **Proposition** if \( E \sim F \) then \( E \equiv F \)

- **Proposition** if \( E \) and \( F \) belongs to a finitely branching transition system/Kripke model and \( E \equiv F \) then \( E \sim F \).
Summary: Unfolding

A transition system/Kripke model can be unfolded into a (bisimulation equivalent) possibly infinite tree.

\[ \begin{align*}
\text{p, q} & \rightarrow \text{s} \\
\text{s} & \rightarrow \text{q} \\
\text{r} & \rightarrow \text{s'} \\
\text{s'} & \rightarrow \text{q} \\
\text{s''} & \rightarrow \text{s'}
\end{align*} \]

Becomes
Computational Properties

- **Satisfiability Problem**  “Given a modal formula \( \Phi \), is \( \Phi \) satisfiable (realisable)?” is \textbf{NP}-complete.

- **Finite Tree Property**  If \( \Phi \) is satisfiable then \( \Phi \) is satisfiable in a transition system/Kripke model that is a finite tree. (Hence, also \textbf{Finite Model Property}: if \( \Phi \) is satisfiable then \( \Phi \) is satisfiable in a finite transition system/Kripke model.)

- **Model Checking Problem**  “Given a finite transition system/Kripke model, a state \( E \) of it, and a modal formula, does \( E \models \Phi \)?” is \textbf{P}-complete.
Mutual Exclusion: Crucial Properties

• Mutual exclusion

• Absence of deadlock

• Absence of starvation

**PROBLEM**: None of these properties is expressible in modal logic!
Summary: Runs

- A run from state \(E_0\) is a finite or infinite length sequence of transitions
  \[ E_0 \xrightarrow{a_1} E_1 \xrightarrow{a_2} \ldots \xrightarrow{a_n} E_n \xrightarrow{a_{n+1}} \ldots \] with “maximal” length.

- Similarly, for a Kripke model.

- A run is a branch in the unfolded transition system/Kripke model.
Beyond Modal Logic

Runs provide a means for expressing long term features.

- **Mutual exclusion**: no run has the property that two components are in their critical section at the same time.

- **Absence of deadlock**: every run has infinite length.

- **Absence of starvation**: in every run if a component requests entry into critical section then eventually that component will be in its critical section.
Summary: Temporal Operators on Runs

• **Next:** \((K)\Phi\)

\[
E_0 \xrightarrow{a_1} E_1 \quad E_1 \xrightarrow{a_2} \ldots \quad E_i \xrightarrow{a_{i+1}} \ldots
\]

\[
a_1 \in K \quad \models \quad \Phi
\]

• **Until:** \(\Phi U \Psi\). Note: the index \(i\) can be 0

\[
E_0 \xrightarrow{a_1} E_1 \xrightarrow{a_2} \ldots \quad E_i \xrightarrow{a_{i+1}} \ldots
\]

\[
\models \quad \models \quad \ldots \quad \models
\]

\[
\Phi \quad \Phi \quad \Psi
\]
Summary: Temporal Operators on Runs II

- **Eventually:** $F \Psi = tt U \Psi$. The index $i$ can be 0

\[
\begin{array}{cccccc}
E_0 & \overset{a_1}{\rightarrow} & E_1 & \overset{a_2}{\rightarrow} & \ldots & E_i & \overset{a_{i+1}}{\rightarrow} \ldots \\
\parbox{1.5cm}{\centering \vdots} & = & \parbox{1.5cm}{\centering \vdots} & = & \parbox{1.5cm}{\centering \vdots} & = \parbox{1.5cm}{\centering \ldots} \\
\Psi & = & \Psi & = & \Psi & \vdots \\
\end{array}
\]

- **Always:** $G \Psi = \neg F \neg \Psi$

\[
\begin{array}{cccccc}
E_0 & \overset{a_1}{\rightarrow} & E_1 & \overset{a_2}{\rightarrow} & \ldots & E_i & \overset{a_{i+1}}{\rightarrow} \ldots \\
\parbox{1.5cm}{\centering \vdots} & = & \parbox{1.5cm}{\centering \vdots} & = & \parbox{1.5cm}{\centering \vdots} & = \parbox{1.5cm}{\centering \ldots} \\
\Psi & = & \Psi & = & \Psi & \vdots \\
\end{array}
\]
Summary: Linear Time Temporal Logic (LTL)

Syntax

\[ \Phi ::= p \mid \neg \Phi \mid \Phi_1 \land \Phi_2 \mid (\neg) \Phi \mid \Phi U \Psi \]

Semantics

A state \( E \) of a Kripke model satisfies an LTL formula \( \Phi \), written \( E \models \Phi \), if for any run \( \pi \) from \( E \), the run \( \pi \models \Phi \).
Summary: Invariance

$E \equiv F$ if for all LTL $\Phi$, $E \models \Phi$ iff $F \models \Phi$.

**Proposition** If $E \sim F$ then $E \equiv F$.

**Proof Sketch** Bisimulation equivalence preserves runs.
Summary: Computational Properties I

- **Satisfiability Problem**
  
  "Given an LTL formula $\Phi$, is $\Phi$ satisfiable (realisable)?"

  is $\text{PSPACE}$-complete.

- **"Tree" Model Property**
  
  If $\Phi$ is satisfiable then $\Phi$ is satisfiable in a transition system/Kripke model that is a (regular) infinite tree whose branching degree is one.
Summary: Computational Properties II

- **Finite Model Property**  If $\Phi$ is satisfiable then $\Phi$ is satisfiable in a finite transition system/Kripke model (that is eventually cyclic).

- **Model Checking Problem**  “Given a finite transition system/Kripke model, a state $E$ of it, and an LTL formula, does $E \models \Phi$?” is PSPACE-complete.
Branching Time Logics

For each temporal operator such as F, create two variants

- $A F$ “for all runs eventually” (strong)
- $E F$ “for some run eventually” (weak)

Modal operators are also branching time temporal operators

- $[K] = A \neg (K) \neg$
- $\langle K \rangle = E (K)$

Therefore, we can extend modal logic with branching time temporal operators.
Computation Tree Logic (CTL)

Syntax

\[ \Phi ::= \text{tt} \mid \neg \Phi \mid \Phi_1 \land \Phi_2 \mid \langle K \rangle \Phi \mid A(\Phi_1 U \Phi_2) \mid E(\Phi_1 U \Phi_2) \]

Semantics

Define when a state of a transition system satisfies a CTL formula.
Semantics of CTL

The new clauses

\[ E \models \neg \Phi \iff E \not\models \Phi \]

\[ E_0 \models A(\Phi U \Psi) \iff \text{for all runs } E_0 \xrightarrow{a_1} E_1 \xrightarrow{a_2} \ldots \text{ there is } i \geq 0 \text{ with } E_i \models \Psi \text{ and for all } j : 0 \leq j < i, E_j \models \Phi \]

\[ E_0 \models E(\Phi U \Psi) \iff \text{for some run } E_0 \xrightarrow{a_1} E_1 \xrightarrow{a_2} \ldots \text{ there is } i \geq 0 \text{ with } E_i \models \Psi \text{ and for all } j : 0 \leq j < i, E_j \models \Phi \]
Derived Operators

\[ AF \Phi = A(tt U \Phi) \]
\[ EF \Phi = E(tt U \Phi) \]
\[ AG \Phi = \neg EF \neg \Phi \]
\[ EG \Phi = \neg AF \neg \Phi \]

- **Safety** “nothing bad ever happens”: in every run bad is never true. \( AG \) good
- **Liveness** “something good eventually happens”: in every run good is eventually true. \( AF \) good
- **Weak Safety** in some run bad is never true. \( EG \) good
- **Weak Liveness** in some run good is eventually true. \( EF \) good
Example: Crossing

It is never the case that a train and a car can cross at the same time

$$AG([\text{tcross}]ff \lor [\text{ccross}]ff)$$
Example: Mutual Exclusion

- **Mutual exclusion**: $\text{AG} ([\text{exit1}]\text{ff} \lor [\text{exit2}] \text{ff})$

- **Absence of deadlock**: $\text{AG} \langle - \rangle \text{tt}$

- **Absence of starvation (for one component)**: $\text{AG} [\text{req1}] \text{AF} \langle \text{exit1} \rangle \text{tt}$
Which of the following are true?

\[ E \models A(p \cup q) \quad E \models A F r \]
\[ E \models E F r \quad E \models E G p \]
\[ E \models A F(E G r \lor E G p) \]
Exercise: Which are Valid, Unsatisfiable, Neither?

<table>
<thead>
<tr>
<th></th>
<th>V</th>
<th>U</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathbf{A \ G \ \Phi \rightarrow A \ F \ \Phi}$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mathbf{E \ G \ \Phi \rightarrow A \ F \ \Phi}$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mathbf{A \ F (\Phi \lor \Psi) \rightarrow A \ F \ \Phi \lor A \ F \ \Psi}$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mathbf{A \ G \ \Phi \rightarrow (\Phi \land [\neg] A \ G \Phi}$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mathbf{A \ G \ A \ F \Phi \rightarrow E \ F \ E \ G \Phi}$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mathbf{A \ F \ A \ G \ \Phi \rightarrow A \ G \ A \ F \ \Phi}$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mathbf{A \ G (\neg \Phi \lor (\neg) \Phi) \land \Phi \land A \ F \ \neg \Phi}$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mathbf{A \ G (\Phi \rightarrow \Psi) \rightarrow (A \ G \ \Phi \rightarrow A \ G \ \Psi)}$</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Exercise

Let $E \equiv F$ if for all CTL $\Phi$, $E \models \Phi$ iff $F \models \Phi$.

Prove the following:

**Proposition** If $E \sim F$ then $E \equiv F$. 
Computational Properties

- **Satisfiability Problem** “Given an CTL formula $\Phi$, is $\Phi$ satisfiable?” is EXPTIME-complete.

- **Tree Model Property** If $\Phi$ is satisfiable then $\Phi$ is satisfiable in a transition system/Kripke model that is a (regular) infinite tree.

- **Finite Model Property** If $\Phi$ is satisfiable then $\Phi$ is satisfiable in a finite transition system/Kripke model.

- **Model Checking Problem** “Given a finite transition system/Kripke model, a state $E$ of it, and a CTL formula, does $E \models \Phi$?” is P-complete.
Incomparability of LTL and CTL

A formula of $\Phi$ of LTL is equivalent to a formula $\Psi$ of CTL if for every model and state $E$, $E \models \Phi$ iff $E \models \Psi$.

- CTL and LTL are expressively incomparable.

- $\text{EFA}Gp$ is not expressible in LTL and $\text{FG}p$ is not expressible in CTL
• **Open Question:** which formulas of LTL are in CTL? (Known: which formulas of CTL are in LTL.)
CTL*  

Syntax  

Allow the facility to have branching time formulas with arbitrary embeddings of linear time and boolean operators.

$$\Phi ::= p | \neg \Phi | \Phi_1 \land \Phi_2 | (\neg) \Phi | \Phi U \Psi | A \Phi$$

Example formula: $A(F G \Phi \land G F \Psi)$
Semantics of CTL*

A state $E$ of a Kripke model satisfies an CTL* formula $\Phi$, written $E \models \Phi$, if for any run $\pi$ from $E$, the run $\pi \models \Phi$.

$\pi(0)$ is initial state of $\pi$

$$\pi \models A\Phi \text{ iff } \text{ for any run } \pi' \text{ if } \pi'(0) = \pi(0) \text{ then } \pi' \models \Phi$$

CTL* contains both LTL and CTL.

Let $E \equiv F$ if for all $\Phi \in \text{CTL}^*$, $E \models \Phi$ iff $F \models \Phi$.

**Proposition** If $E \sim F$ then $E \equiv F$. 
Computational Properties

- **Satisfiability Problem** “Given a CTL* formula $\Phi$, is $\Phi$ satisfiable?” is $\mathsf{2EXPTIME}$-complete.

- **Tree Model Property** If $\Phi$ is satisfiable then $\Phi$ is satisfiable in a transition system/Kripke model that is a (regular) infinite tree.

- **Finite Model Property** If $\Phi$ is satisfiable then $\Phi$ is satisfiable in a finite transition system/Kripke model.

- **Model Checking Problem** “Given a finite transition system/Kripke model, a state $E$ of it, and a CTL* formula, does $E \models \Phi$?” is $\mathsf{PSPACE}$-complete.
Model Checking CTL Formulas

\[ \| \Phi \| = \{ E : E \models \Phi \} \] in a fixed model.

Model checking is “bottom up” by computing \( \| \Psi \| \) for any subformula of \( \Phi \) and then computing \( \| \Phi \| \).

- \[ \| \neg \Phi_1 \| = - \| \Phi_1 \| \]
- \[ \| \Phi_1 \land \Phi_2 \| = \| \Phi_1 \| \cap \| \Phi_2 \| \]
- \[ \| \langle K \rangle \Phi \| = \{ F : \exists F' \in \| \Phi \|, a \in K. F \xrightarrow{a} F' \} \]
Model Checking CTL

- $\parallel E(\Phi U \Psi) \parallel = \bigcup S_i$ where $S_1 = \parallel \Psi \parallel$ and
  
  $S_{i+1} = S_i \cup \{ F \in \parallel \Phi \parallel : \exists a, F' \in S_i. F \xrightarrow{a} F' \}$

- $\parallel A(\Phi U \Psi) \parallel = \bigcup S_i$ where $S_1 = \parallel \Psi \parallel$ and
  
  $S_{i+1} = S_i \cup \{ F \in \parallel \Phi \parallel : \exists a, F' \in S_i. F \xrightarrow{a} F' \text{ and } \forall a, F'. \text{ if } F \xrightarrow{a} F' \text{ then } F' \in S_i \}$

Direct backwards reachability computation for $E U$ in transition graph. For $A U$ it is forward reachability which can be implemented efficiently using depth first search.

Both are fixed point computations: essence of model checking.
Temporal Operators as Fixed points

1. \( E(\Phi \cup \Psi) \equiv \Psi \lor (\Phi \land \langle - \rangle E(\Phi \cup \Psi)) \)

2. \( A(\Phi \cup \Psi) \equiv \Psi \lor (\Phi \land \langle - \rangle tt \land [-] A(\Phi \cup \Psi)) \)

Syntactically: property \( X \) such that

1. \( X \equiv \Psi \lor (\Phi \land \langle - \rangle X) \)

2. \( X \equiv \Psi \lor (\Phi \land \langle - \rangle tt \land [-] X) \)
Temporal Operators as Fixed points

Semantically: set of states $S = f(S)$ where $f$ is

1. $\lambda x. \| \Psi \lor (\Phi \land \langle \neg \rangle x) \|$

2. $\lambda x. \| \Psi \lor (\Phi \land \langle \neg \rangle \text{tt} \land [\neg]x) \|$

If $S = f(S)$ then $S$ is a fixed point of $f$.

In both cases $f$ is monotonic: $S \subseteq S' \rightarrow f(S) \subseteq f(S')$

$f$ is essentially modal (using $\langle \neg \rangle$ and $[\neg]$).
Fixed points

$S$ is a **prefix fixed point** of $f$, if $f(S) \subseteq S$

$S$ is a **postfix fixed point** of $f$, if $S \subseteq f(S)$

**Proposition** If $f$ is monotonic (w.r.t $\subseteq$) then $f$

1. has a **least** fixed point, $\bigcap \{S : f(S) \subseteq S\}$

2. has a **greatest** fixed point, $\bigcup \{S : S \subseteq f(S)\}$

**Exercise:** Prove this.
Example

- $S = f(S)$ when $f = \lambda x.\|\Psi \lor (\Phi \land (\neg x))\|$

- There can be many different fixed points of $f$.

- The one wanted that expresses $E(\Phi \cup \Psi)$ is the least fixed point.

- **Exercise:** what does the greatest fixed point of $f$ express?
Example

• $A G \Phi \equiv \Phi \land [\neg]A G \Phi$. Computing $\| A G \Phi \|$: Simple depth first search, that stops when reach a state in $\| \neg \Phi \|$.

• $\| A G \Phi \| = \bigcap S_i$ where $S_1 = \| \Phi \|$ and

  $S_{i+1} = S_i \cap \{ F \in S_i : \forall a, F \xrightarrow{a} F' \text{ implies } F' \in S_i \}$

• **Syntactically:** Property $X$ such that $X \equiv \Phi \land [\neg]X$

  **Semantically:** fixed point of $f = \lambda x.\| \Phi \land [\neg]x \|

• Required property, $A G \Phi$, is **greatest** fixed point of $f$

• **Exercise:** What does the least fixed point of $f$ express?
Exercise

What property is defined by the following fixed points?

1. \( X \equiv \Phi \lor \langle - \rangle X \) least

2. \( X \equiv \Phi \land \langle - \rangle X \) greatest

3. \( X \equiv \Phi \land [\langle - \rangle \langle - \rangle] X \) greatest
A Scheduler

Problem: assume $n$ tasks when $n > 1$.

$a_i$ initiates the $i$th task and $b_i$ signals its completion

The scheduler plans the order of task initiation, ensuring

1. actions $a_1 \ldots a_n$ carried out cyclically and tasks may terminate in any order

2. but a task can not be restarted until its previous operation has finished.
   ($a_i$ and $b_i$ happen alternately for each $i$. )

More complex temporal properties. Not expressible in CTL* (“not first order” but are “regular”).

Expressible using fixed points
Modal Logic

\( Z \) ranges over propositional variables

\[ \Phi ::= Z \mid \text{tt} \mid \text{ff} \mid \Phi_1 \land \Phi_2 \mid \Phi_1 \lor \Phi_2 \mid [K] \Phi \mid \langle K \rangle \Phi \]

- \( \models \) refined to \( \models_V \) where \( V \) is a **valuation** that assigns a set of states \( V(X) \) to each variable \( X \)

\[ E \models_V X \iff E \in V(X) \]

- \( \| \Phi \| \) refined too: \( \| \Phi \|_V = \{ E : E \models_V \Phi \} \)

- \( V[S/X] \) is valuation \( V' \) like \( V \) except \( V'(X) = S \).
Modal Logic II

**Proposition** The function $\lambda x. \| \Phi \|_{V[x/X]}$ is monotonic for any modal $\Phi$.

- If $\neg$ explicitly in logic then above not true: $\neg X: \lambda x. \neg x$ not monotonic.

  However, define when $\Phi$ is **positive** in $X$: if $X$ occurs within an even number of negations in $\Phi$

  **Proposition** If $\Phi$ is positive in $X$ then $\lambda x. \| \Phi \|_{V[x/X]}$ is monotonic

1. Property given by **least** fixed point of $\lambda x. \| \Phi \|_{V[x/X]}$ is written $\mu X. \Phi$.

2. Property given by **greatest** fixed point of $\lambda x. \| \Phi \|_{V[x/X]}$ is written $\nu X. \Phi$.

Alternative basis for temporal logic: **modal logic + fixed points**