Outline

Introduction

Models

- Temporal Specifications
  - General Definitions
  - (Linear) Temporal Specifications
  - Branching Temporal Specifications
  - CTL
  - CTL*  

Satisfiability and Model Checking

More on Temporal Specifications

Static and dynamic properties

Example: Static properties
- Mutual exclusion
- Safety properties are often static.
- They can be reduced to reachability.

Example: Dynamic properties
- Every elevator request should be eventually granted.
- The elevator should not cross a level for which a call is pending without stopping.

Temporal Structures

Definition: Flows of time
- A flow of time is a strict order \((T, <)\) where \(T\) is the nonempty set of time points and \(<\) is an irreflexive transitive relation on \(T\).

Example: Flows of time
- \((\{0, \ldots, n\}, <)\): Finite runs of sequential systems.
- \((\mathbb{N}, <)\): Infinite runs of sequential systems.
- \((\mathbb{R}, <)\): Runs of real-time sequential systems.
- Trees: Finite or infinite run-trees of sequential systems.
- Mazurkiewicz traces: Runs of distributed systems (partial orders).
- And also \((\mathbb{Z}, <)\) or \((\mathbb{Q}, <)\) or \((\omega^2, <)\), ...

Definition: Temporal Structures

Let \(AP\) be a set of atoms (atomic propositions).
A temporal structure over a class \(C\) of time flows and \(AP\) is a triple \((T, <, h)\) where \((T, <)\) is a time flow in \(C\) and \(h : AP \to 2^T\) is an assignment.
If \(p \in AP\) then \(h(p) \subseteq T\) gives the time points where \(p\) holds.

Linear behaviors and specifications

Let \(M = (S, T, I, AP, \ell)\) be a Kripke structure.

Definition: Runs as temporal structures
An infinite run \(\sigma = s_0s_1s_2 \cdots\) of \(M\) with \((s_i, s_{i+1}) \in T\) for all \(i \geq 0\) defines a linear temporal structure \(\ell(\sigma) = (\mathbb{N}, <, h)\) where \(h(p) = \{i \in \mathbb{N} \mid p \in \ell(s_i)\}\).

Such a temporal structure can be seen as an infinite word over \(\Sigma = 2^{AP}\):
\(\ell(\sigma) = \ell(s_0)\ell(s_1)\ell(s_2) \cdots = (\mathbb{N}, <, w)\) with \(w(i) = \ell(s_i) \in \Sigma\).

Linear specifications only depend on runs.
Example: The printer manager is fair.
On each run, whenever some process requests the printer, it eventually gets it.

Remark:
Two Kripke structures having the same linear temporal structures satisfy the same linear specifications.
Branching behaviors and specifications

The system has an infinite active run, but it may always reach an inactive state.

Definition: Computation-tree or run-tree: unfolding of the TS

Let $M = (S, T, I, AP, \ell)$ be a Kripke structure. Wlog. $I = \{s_0\}$ is a singleton.
Let $D$ be a finite set with $|D|$ the outdegree of the transition relation $T$.

The computation-tree of $M$ is an unordered tree $t : D^* \to S$ (partial map) s.t.
- $t(\varepsilon) = s_0$,
- For every node $u \in \text{dom}(t)$ labelled $s = t(u)$, if $T(s) = \{s_1, \ldots, s_k\}$ then $u$ has exactly $k$ children which are labelled $s_1, \ldots, s_k$.

Associated temporal structure $\ell(t) = (\text{dom}(t), <, h)$ where
- $<$ is the strict prefix relation over $D^*$,
- and $h(p) = \{u \in \text{dom}(t) \mid p \in \ell(t(u))\}$.

(Linear) runs of $M$ are branches of the computation-tree $t$.

First-order vs Temporal

First-order logic
- FO($<$) has a good expressive power
- but FO($<$)-formulae are not easy to write and to understand.
- FO($<$) is decidable
- but satisfiability and model checking are non elementary.

Temporal logics
- no variables: time is implicit.
- quantifications and variables are replaced by modalities.
- Usual specifications are easy to write and read.
- Good complexity for satisfiability and model checking problems.
- Good expressive power.

Linear Temporal Logic (LTL) over $(\mathbb{N}, <)$ introduced by Pnueli (1977) as a convenient specification language for verification of systems.

First-order Specifications

Definition: Syntax of FO($AP, <$)
Let $\text{Var} = \{x, y, \ldots\}$ be first-order variables.

$$\varphi ::= \bot \mid p(x) \mid x = y \mid x < y \mid \neg \varphi \mid \varphi \lor \varphi \mid \exists x \varphi$$

where $p \in AP$.

Definition: Semantics of FO($AP, <$)
Let $w = (\mathbb{T}, <, h)$ be a temporal structure over $AP$.
Let $\nu : \text{Var} \to \mathbb{I}$ be an assignment of first-order variables to time points.

$$w, \nu \models p(x) \quad \text{if} \quad \nu(x) \in h(p)$$
$$w, \nu \models x = y \quad \text{if} \quad \nu(x) = \nu(y)$$
$$w, \nu \models x < y \quad \text{if} \quad \nu(x) < \nu(y)$$
$$w, \nu \models \exists x \varphi \quad \text{if} \quad w, \nu(x \mapsto t) \models \varphi \text{ for some } t \in \mathbb{T}$$

where $\nu(x \mapsto t)$ maps $x$ to $t$ and $y \neq x$ to $\nu(y)$.

Previous specifications can be written in FO($<$) (except the branching one).

Temporal Specifications

Definition: Syntax of TL($AP, SU, SS$)

$$\varphi ::= \bot \mid p\ (p \in AP) \mid \neg \varphi \mid \varphi \lor \varphi \mid \varphi \land \varphi \mid \varphi \land SS \varphi \mid \varphi \land SU \varphi$$

Definition: Semantics: $w = (\mathbb{T}, <, h)$ temporal structure and $i \in \mathbb{I}$

$$w, i \models p \quad \text{if} \quad i \in h(p)$$
$$w, i \models \neg \varphi \quad \text{if} \quad w, i \not\models \varphi$$
$$w, i \models \varphi \lor \psi \quad \text{if} \quad w, i \models \varphi \text{ or } w, i \models \psi$$
$$w, i \models \varphi \land SU \psi \quad \text{if} \quad \exists k \ i < k \text{ and } w, k \models \psi \text{ and } \forall j \ i < j < k \rightarrow w, j \models \varphi$$
$$w, i \models \varphi \land SS \psi \quad \text{if} \quad \exists k \ i > k \text{ and } w, k \models \psi \text{ and } \forall j \ i > j > k \rightarrow w, j \models \varphi$$

Previous specifications can be written in TL($AP, SU, SS$) (except the branching one).
Temporal Specifications

Definition: non-strict versions of until and since

\[ \varphi U \psi \overset{def}{=} \psi \lor (\varphi \land SU \psi) \quad \varphi S \psi \overset{def}{=} \psi \lor (\varphi \land SS \psi) \]

\[ w, i \models \varphi U \psi \text{ if } \exists k i \leq k \text{ and } w, k \models \psi \land \forall j (i \leq j < k \to w, j \models \varphi) \]

\[ w, i \models \varphi S \psi \text{ if } \exists k i \geq k \text{ and } w, k \models \psi \land \forall j (i \geq j > k \to w, j \models \varphi) \]

Definition: Derived modalities

\[ X \varphi \overset{def}{=} SU \varphi \quad \text{Next} \quad Y \varphi \overset{def}{=} SS \varphi \quad \text{Yesterday} \]

\[ w, i \models X \varphi \text{ if } \exists k i < k \text{ and } w, k \models \varphi \land \forall j (i < j < k) \]

\[ w, i \models Y \varphi \text{ if } \exists k i > k \text{ and } w, k \models \varphi \land \forall j (i > j > k) \]

\[ F \varphi \overset{def}{=} T U \varphi \quad P \varphi \overset{def}{=} T S \varphi \]

\[ G \varphi \overset{def}{=} \neg F \neg \varphi \quad H \varphi \overset{def}{=} \neg P \neg \varphi \]

\[ \varphi W \psi \overset{def}{=} (G \varphi) \lor (\varphi U \psi) \quad \text{Weak Until} \]

\[ \varphi R \psi \overset{def}{=} (G \psi) \lor (\psi U (\varphi \land \psi)) \quad \text{Release} \]

Discrete linear time flows

Definition: discrete linear time flows \((T, <)\)

A linear time flow is **discrete** if \(SF T \to X T\) and \(SP T \to Y T\) are valid formulae.

\((N, <)\) and \((Z, <)\) are discrete. \((Q, <)\) and \((R, <)\) are not discrete.

Exercise: For discrete linear time flows \((T, <)\)

\[ \varphi SU \psi \equiv X(\varphi U \psi) \]

\[ \varphi SS \psi \equiv Y(\varphi S \psi) \]

\[ \neg X \varphi \equiv \neg X T \lor X \neg \varphi \]

\[ \neg Y \varphi \equiv \neg Y T \lor Y \neg \varphi \]

\[ \neg(\varphi U \psi) \equiv (G \neg \psi) \lor (\neg \psi U (\neg \varphi \land \neg \psi)) \]

\[ \neg \psi W (\neg \varphi \land \neg \psi) \]

\[ \neg \varphi R \neg \psi \]

Temporal Specifications

Example: Specifications on the time flow \((\mathbb{N}, <)\)

- Safety: \(G \text{ good} \)
- Mutual exclusion: \(\neg F(\text{crit}_{1} \land \text{crit}_{2})\)
- Liveness: \(GF \text{ active} \)
- Response: \(G(\text{request} \to F \text{ grant}) \)
- Response': \(G(\text{request} \to (\neg \text{request} SU \text{ grant})) \)
- Release: \(\text{reset} R \text{ alarm} \)
- Strong fairness: \((GF \text{ request}) \to (GF \text{ grant})\)
- Weak fairness: \((FG \text{ request}) \to (FG \text{ grant})\)

Model checking for linear behaviors

Definition: Model checking problem

Input: A Kripke structure \(M = (S, T, I, AP, \ell)\)

A formula \(\varphi \in \text{LTL}(AP, SU, SS)\)

Question: Does \(M \models \varphi\)?

- Universal MC: \(M \models \varphi\) if \(\ell(\sigma), 0 \models \varphi\) for all initial infinite runs \(\sigma\) of \(M\).
- Existential MC: \(M \models \exists \varphi\) if \(\ell(\sigma), 0 \models \varphi\) for some initial infinite run \(\sigma\) of \(M\).

\[ M \models \varphi \quad \text{iff} \quad M \not\models \exists \neg \varphi \]

Theorem [10, Sistla, Clarke 85], [9, Lichtenstein & Pnueli 85]

The Model checking problem for LTL is PSPACE-complete. Proof later
**Weaknesses of linear behaviors**

**Example:**
\[ \psi: \text{Whenever } p \text{ holds, it is possible to reach a state where } q \text{ holds.} \]
\[ \varphi \text{ cannot be checked on linear behaviors.} \]
We need to consider the computation-trees.

---

**Weaknesses of FO specifications**

**Example:**
\[ \psi: \text{The system has an infinite active run, but it may always reach an inactive state.} \]
\[ \varphi \text{ cannot be expressed in FO.} \]
We need quantifications on runs:
\[ \psi = EG(\text{Active} \land EF \neg \text{Active}) \]

- E: for some infinite run
- A: for all infinite runs

---

**MSO Specifications**

**Definition: Syntax of MSO(AP, <)**
\[ \varphi ::= \bot \mid p(x) \mid x = y \mid x < y \mid x \in X \mid \neg \varphi \mid \varphi \lor \varphi \mid \exists x \varphi \mid \exists X \varphi \]
where \( p \in \text{AP} \), \( x, y \) are first-order variables and \( X \) is a second-order variable.

**Definition: Semantics of MSO(AP, <)**

Let \( w = (T, <, h) \) be a temporal structure over \( \text{AP} \).
An assignment \( \nu \) maps first-order variables to time points in \( T \) and second-order variables to sets of time points.
The semantics of first-order constructs is unchanged.
\[ w, \nu \models x \in X \quad \text{if} \quad \nu(x) \in \nu(X) \]
\[ w, \nu \models \exists X \varphi \quad \text{if} \quad w, \nu[X \mapsto T] \models \varphi \text{ for some } T \subseteq T \]
where \( \nu[X \mapsto T] \) maps \( X \) to \( T \) and keeps unchanged the other assignments.

---

**MSO vs Temporal**

**MSO logic**
- MSO(<) has a good expressive power
  ... but MSO(<)-formulae are not easy to write and to understand.
- MSO(<) is decidable on computation trees
  ... but satisfiability and model checking are non elementary.

**We need a temporal logic**
- with no explicit variables,
- allowing quantifications over runs,
- usual specifications should be easy to write and read,
- with good complexity for satisfiability and model checking problems,
- with good expressive power.

**Computation Tree Logic CTL^\ast** introduced by Emerson & Halpern (1986).
The model checking problem for CTL* (Emerson & Halpern 86)

Definition: Syntax of the Computation Tree Logic CTL*(AP, SU)
\[ \varphi ::= \bot \mid p \ (p \in AP) \mid \neg \varphi \mid \varphi \lor \varphi \mid \varphi \lor \varphi \mid E \varphi \mid A \varphi \]

We may also add the past modality SS

Definition: Semantics of CTL*(AP, SU)
Let \( M = (S, T, I, AP, \ell) \) be a Kripke structure. Let \( \sigma = s_0 s_1 s_2 \cdots \) be an infinite run of \( M \).

\( M, \sigma, i \models p \) if \( p \in \ell(s_i) \)

\( M, \sigma, i \models E \varphi \) if \( M, \sigma', i \models \varphi \) for some infinite run \( \sigma' \) such that \( \sigma'[i] = \sigma[i] \)

\( M, \sigma, i \models A \varphi \) if \( M, \sigma', i \models \varphi \) for all infinite runs \( \sigma' \) such that \( \sigma'[i] = \sigma[i] \)

where \( \sigma[i] = s_0 \cdots s_i \).

Remark:
- \( \sigma'[i] = \sigma[i] \) means that future is branching but past is not.

Model checking of CTL*

Definition: Existential and universal model checking
Let \( M = (S, T, I, AP, \ell) \) be a Kripke structure and \( \varphi \in CTL^* \) a formula.

\( M \models \exists \varphi \) if \( M, \sigma, 0 \models \varphi \) for some initial infinite run \( \sigma \) of \( M \).

\( M \models \forall \varphi \) if \( M, \sigma, 0 \models \varphi \) for all initial infinite runs \( \sigma \) of \( M \).

Remark: \( M \models \forall \varphi \) iff \( M \not\models \exists \neg \varphi \)

Definition: Model checking problems MC_{CTL^*} and MC_{CTL^3}
Input: A Kripke structure \( M = (S, T, I, AP, \ell) \) and a formula \( \varphi \in CTL^* \)

Question: Does \( M \models \forall \varphi \)? or Does \( M \models \exists \varphi \)?

Theorem: The model checking problem for CTL* is PSPACE-complete. Proof later

CTL* (Emerson & Halpern 86)

Example: Some specifications
- EF \( \varphi \): \( \varphi \) is possible
- AG \( \varphi \): \( \varphi \) is an invariant
- AF \( \varphi \): \( \varphi \) is unavoidable
- EG \( \varphi \): \( \varphi \) holds globally along some path

Remark: Some equivalences
- \( A \varphi \equiv \neg E \neg \varphi \)
- \( E(\varphi \lor \psi) \equiv E \varphi \lor E \psi \)
- \( A(\varphi \land \psi) \equiv A \varphi \land A \psi \)

State formulae and path formulae

Definition: State formulae
\( \varphi \in CTL^* \) is a state formula if \( \forall M, \sigma, i, j \) such that \( \sigma(i) = \sigma(j) \) we have

\[ M, \sigma, i \models \varphi \iff M, \sigma', j \models \varphi \]

If \( \varphi \) is a state formula and \( M = (S, T, I, AP, \ell) \), define

\[ M, s \models \varphi \] if \( M, \sigma, 0 \models \varphi \) for some infinite run \( \sigma \) of \( M \) with \( \sigma(0) = s \)

and \( [\varphi]^M = \{ s \in S \mid M, s \models \varphi \} \)

Example: State formulae
Atomic propositions are state formulae: \( [p] = \{ s \in S \mid p \in \ell(s) \} \)
State formulae are closed under boolean connectives.

\[ [\neg \varphi] = \{ S \setminus \{ s \} \} \]
\[ [\varphi_1 \lor \varphi_2] = \{ \varphi_1 \lor \varphi_2 \} \]

Formulae of the form \( E \varphi \) or \( A \varphi \) are state formulae, provided \( \varphi \) is future.

Remark: \( M \models \varphi \) iff \( \{ s \mid [\varphi]^S \neq \emptyset \} \)

Definition: Alternative syntax
State formulae \( \varphi ::= \bot \mid p \ (p \in AP) \mid \neg \varphi \mid \varphi \lor \varphi \mid E \varphi \mid A \varphi \)
Path formulae \( \psi ::= \varphi \mid \neg \psi \mid \psi \lor \psi \mid E \psi \mid A \psi \mid \psi SU \psi \)
Definition: Computation Tree Logic (CTL)(AP, X, U)

Syntax:

\[ \phi ::= \perp \mid p \ (p \in AP) \mid \neg \phi \mid \phi \lor \psi \mid \text{EX} \phi \mid \text{AX} \phi \mid \text{E} \phi \ U \ \psi \mid \text{A} \phi \ U \ \psi \]

The semantics is inherited from CTL*.

Remark: All CTL formulae are state formulae.

\[ [\phi]^M = \{ s \in S \mid M, s \models \phi \} \]

Examples: Macros

- EF \phi = E \ U \ \phi \quad \text{and} \quad AG \phi = \neg EF \neg \phi
- AF \phi = A \ U \ \phi \quad \text{and} \quad EG \phi = \neg AF \neg \phi
- AG(req \rightarrow F\text{grant})
- AG(req \rightarrow AF \text{grant})

Example:

\[
\begin{array}{c}
\text{p, q} \\
5 & 6 & 7 & 8 \\
\text{p, r} \\
1 & 2 & 3 & 4 \\
\text{q} \\
\end{array}
\]

\[
\begin{align*}
[\text{EX}p] &= \{1, 2, 5, 6\} \\
[\text{AX}p] &= \{1, 3, 4\} \\
[\text{EF}p] &= \{1, 2, 3, 4\} \\
[\text{AF}p] &= \{1, 2, 3, 4\} \\
[\text{E}q \ U \ r] &= \{3\} \\
[\text{A}q \ U \ r] &= \{3\}
\end{align*}
\]
Model checking of $\mathbf{CTL}$

Definition: Existential and universal model checking

Let $M = (S, T, I, \mathbf{AP}, \ell)$ be a Kripke structure and $\varphi \in \mathbf{CTL}$ a formula.

- $M \models \exists \varphi$ if $M,s \models \varphi$ for some $s \in I$.
- $M \models \forall \varphi$ if $M,s \models \varphi$ for all $s \in I$.

Remark:

- $M \models \exists \varphi$ iff $I \cap \mathbb{[}\varphi] \neq \emptyset$
- $M \models \forall \varphi$ iff $I \subseteq \mathbb{[}\varphi]$
- $M \models \forall \varphi$ iff $M \not\models \neg \varphi$

Definition: Model checking problems $\mathbf{MC}_\exists^{\mathbf{CTL}}$ and $\mathbf{MC}_\forall^{\mathbf{CTL}}$

Input: A Kripke structure $M = (S, T, I, \mathbf{AP}, \ell)$ and a formula $\varphi \in \mathbf{CTL}$

Question: Does $M \models \forall \varphi$? or Does $M \models \exists \varphi$?

Theorem:

Let $M = (S, T, I, \mathbf{AP}, \ell)$ be a Kripke structure and $\varphi \in \mathbf{CTL}$ a formula.

The model checking problem $M \models \exists \varphi$ is decidable in time $O(|M| \cdot |\varphi|)$.

References


