Deductive Verification

Based on theorem proving techniques, the method of deductive verification can be used to establish temporal properties of infinite-state reactive systems. We still rely on the FDS computational model, but the variables may range over infinite domains, and the transition relation may contain first-order operations and predicates.

We assume that all CTL* formulas are given in a positive normal form.

For simplicity of the presentation, we consider systems with no compassion requirements.

A. Pnueli

Lecture 4: Deductive Verification of CTL*
A.Pnueli

Structure of the Proof System

The structure of the deductive system we present is as follows:

Rules for each of the basic CTL formulas, i.e., formulas of the form $\Phi$ where

- Rules for each of the basic $\Phi$ formulas, i.e., formulas of the form $\Phi$ where
- A reduction rule which enables us to decompose the verification task into several subtasks, each dealing with a single basic state formula. Recall that a basic state formula is a formula of the form $\Diamond$, where $\Diamond$ contains no path quantifiers.
- A reduction rule which enables us to eliminate one basic path formula at a time, at the cost of conjointing a tester for that formula to the system we are verifying.

Recall that a basic path formula is a formula of the form of the form $\Diamond$, where $\Diamond$ contains no path quantifiers.
- A reduction rule which enables us to decompose the verification task into several

The structure of the deductive system we present is as follows:

Rules for each of the basic $\Phi$ formulas, i.e.,
Preliminary Rules

We assume the availability of an underlying proof system for assertional reasoning.

**Preliminary Rules**

1. **Generalization Rule**
   - For state formulas \( p \) and \( q \),
     
     \[
     \begin{array}{c}
     p \\
     \hline
     q
     \end{array}
     \]
   
   For an assertion, \( d \) is an abbreviation for \( b \Leftarrow d \).

   Following is the **Generalization rule**, we assume the availability of an underlying proof system for assertional reasoning.

   **Preliminary Rules**

2. **Entailment Modus Ponens Rule**
   - For any reachable state, stating that an assertion that has been proven to be generally valid holds, in particular,
     
     \[
     \begin{array}{c}
     d \\
     \hline
     p
     \end{array}
     \]

   Following is the **Generalization** rule.
The following rule \( a-inv \) can be used to prove that if assertion \( p \) holds at state \( s \), then \( b \) holds at all states reachable from \( s \). The following rule \( A-INV \) can be used to prove that if assertion \( \phi \) holds at state \( s \), then \( d \) holds at all states reachable from \( s \).

\[
\begin{align*}
\phi & \leq d \\
\phi & \leq d \lor \phi \\
\phi & \leq d \\
\phi & \leq d \\
\phi & \leq d
\end{align*}
\]

For assertions \( \phi \), \( b \), and \( d \), the auxiliary assertion \( \phi \) is often described as an inductive strengthening of \( b \). The rule itself is based on computational induction.

Finding \( \phi \) and similar auxiliary constructs is one of the most challenging problems in the application of deductive verification, and requires ingenuity and insight.
Lecture 4: Deductive Verification of CTL

Example: MUX-SEM

\[ (0 = y \leftarrow \text{at-}m_3 \wedge \text{at-}m_4) \lor (0 = y \leftarrow \text{at-}m_3 \wedge \text{at-}m_4) \lor (\text{at-}m_3 \wedge \text{at-}m_4) \]

Note that the last two conjuncts form assertions attached at the locations

As the inductive assertion we choose:

\[ \Theta \iff (\text{at-}m_3 \wedge \text{at-}m_4) \]

Wishing to establish mutual exclusion, we use rule A INV to prove
This rule can be used to establish that every state has a successor satisfying:

\[
\begin{align*}
\neg b \lor d \land \neg b & \models d \\
\neg b \lor d \land \neg b & \models d
\end{align*}
\]

For assertions and

**Rule E-NE**:

\[
\begin{align*}
\neg b & \lor d \\
\neg b \lor d & \models d
\end{align*}
\]
Following is rule E-UNTIL:

**Rule E-UNTIL**

\[ \mathcal{A} \mathcal{E} \mathcal{U} \mathcal{N} \mathcal{T} \mathcal{I} \mathcal{L} : \]

\[ \begin{align*}
& (\sigma < \sigma' \lor \phi \lor d) \land \exists ! \mathcal{E} \land \mathcal{U} \land \mathcal{N} \land \mathcal{T} \land \mathcal{I} \\
& \sigma' \Leftrightarrow \phi' \land \mathcal{U}.
\end{align*} \]

\[ \mathcal{A} \mathcal{E} \mathcal{U} \mathcal{T} \mathcal{L} \mathcal{I} \]

\[ \mathcal{A} \mathcal{E} \mathcal{U} \mathcal{N} \mathcal{T} \mathcal{I} : \]

\[ \begin{align*}
& (\sigma < \sigma' \lor \phi \lor d) \land \exists ! \mathcal{E} \land \mathcal{U} \land \mathcal{N} \land \mathcal{T} \land \mathcal{I} \\
& \phi' \Leftrightarrow \phi \land \mathcal{U}.
\end{align*} \]

\[ \mathcal{A} \mathcal{E} \mathcal{U} \mathcal{T} \mathcal{L} \mathcal{I} \]

\[ \mathcal{A} \mathcal{E} \mathcal{U} \mathcal{N} \mathcal{T} \mathcal{I} : \]

\[ \begin{align*}
& (\sigma < \sigma' \lor \phi \lor d) \land \exists ! \mathcal{E} \land \mathcal{U} \land \mathcal{N} \land \mathcal{T} \land \mathcal{I} \\
& \phi' \Leftrightarrow \phi \land \mathcal{U}.
\end{align*} \]

\[ \mathcal{A} \mathcal{E} \mathcal{U} \mathcal{T} \mathcal{L} \mathcal{I} \]

\[ \mathcal{A} \mathcal{E} \mathcal{U} \mathcal{N} \mathcal{T} \mathcal{I} : \]

\[ \begin{align*}
& (\sigma < \sigma' \lor \phi \lor d) \land \exists ! \mathcal{E} \land \mathcal{U} \land \mathcal{N} \land \mathcal{T} \land \mathcal{I} \\
& \phi' \Leftrightarrow \phi \land \mathcal{U}.
\end{align*} \]
Example: BAKERY-2

We prove, using rule E-MONIT, the property

\[ 0 = \tau_0 \quad \text{if } \tau_1 \text{ is the function which yields the natural } j \text{ if } \tau_1 = j. \]

\[
\begin{align*}
(\forall \varepsilon) \quad & (\lnot \forall \varepsilon) \\
0 = \tau_0 \quad & 0 = \tau_0 \\
\lnot \forall \varepsilon \quad & \forall \varepsilon \\
\forall \varepsilon \quad & \lnot \forall \varepsilon \\
\Theta \quad & \Theta
\end{align*}
\]

Choose as follows:

\[ \begin{align*}
\text{m}_0 : \text{Loop forever do} \\
\text{m}_1 : \text{Non-Critical} \\
\text{m}_2 : \text{Critical} \\
\text{m}_3 : \text{wait} \\
\text{m}_4 : \text{Critical} \\
\text{m}_5 : \text{wait} \\
\text{m}_6 : \text{Critical} \\
\end{align*} \]

\[ \begin{align*}
0 = \tau_0 \\
0 = \tau_1 \\
\tau_0 \geq \tau_1 \\
0 = \tau_0 \\
\tau_0 > \tau_1 \\
\tau_0 \geq \tau_2 \\
\tau_0 = \tau_0 \\
\end{align*} \]

local \( \tau_1, \tau_2, \tau_3 \quad \text{natural initially} \quad \tau_1, \tau_2, \tau_3 \text{ initially} 0 = \tau_2 = \tau_1 = 0 \]
For assertions $p, q, \ldots, q_m$, whose justice requirements are $J_0, \ldots, J_m$, and there exists a computation which visits $q_0, \ldots, q_m$ in a round-robin fashion. The rule requires identifying auxiliary assertions $p_0, \ldots, p_m$, such that each $p_i$ implies $q_i$ and there exists a computation.

\[
\begin{array}{c}
\text{Rule E-INV} \\
\text{A. Pnueli}
\end{array}
\]
Lecture 4: Deductive Verification of CTL

A. Pnueli

Universal Response - Rule

\( \forall f \forall \mathcal{A} \Rightarrow \mathcal{D} \)

For justice requirements and ranking functions and well-founded domain (\( \forall \mathcal{A}, \mathcal{D} \) and well-founded domain (\( \forall \mathcal{A}, \mathcal{D} \))

\( \exists \mathcal{A}, \mathcal{D} \)
**Example:** bakery-2

- Initially: \( y_1 = y_2 = 0 \)
- Loop forever: do
  - Non-Critical: \( y_1 := y_2 + 1 \)
  - Critical: \( y_1 := 0 \)

We prove, using rule 3.

- Event: \( \forall f \phi \iff \Diamond f \phi \)
- \( \phi = ( \forall \phi_1 \land ( \neg \phi_2 \lor \phi_3 \land \phi_4 \land \phi_5 ) \)
A Pnueli

Advanced Topics in Reactive Verification, NYU, Spring, 2004

Lecture 4: Deductive Verification of CTL

Veriﬁcation Diagrams

An A-EVENT proof can also be presented in a veriﬁcation diagram.
Summary of First Part

In the first part of the deductive verification system, we provided rules for the properties $A_p$, $E_X q$, $q E U r$, $E f q$, and $A f q$. Note that fair can be inferred from $E f 1$.

Derived rules can be constructed for all the remaining combinations of CTL operators.

In the first part of the deductive verification system, we provided rules for the properties $A_p$, $E_X q$, $q E U r$, $E f q$, and $A f q$. Note that fair can be inferred from $E f 1$.

${\{\text{M}, \text{N}, \Diamond, \square, \bigcirc\} \times \{\text{A}, \text{E}, \text{A}, \text{E}, \text{E}, \text{E}\}}$ taken from
II. Decomposing a Proof into Proofs of Basic State Formulas

For a formula \( \phi \), containing occurrences of the basic state formula \( \psi \), and an assertion \( p \), the following rule allows us to decompose a proof of an arbitrary state formula into proofs of basic state formulas:

\[
\frac{\phi \rightarrow f}{(d)f} \quad \text{R2.}
\]

\[
\phi \leftrightarrow d \quad \text{R1.}
\]

Recall that a basic state formula is a formula of the form \( \psi \), where \( \psi \) contains no path quantifiers.
We wish to prove for this system the property claiming the existence of a run from each of whose states it is possible to reach a state at which

\[ \forall I, E(x) \]

\[ I = x \]

Note that, as the assertion \( d \), we have chosen \( x = 0 \). The design of an appropriate

\[ (0 = x) \quad \square E \quad (I = x) \]

\[ (I = x) \quad \lozenge E \leftrightarrow (0 = x) \]

Using BASIC-STATE, it is possible to reduce the task of verifying the non-basic formula

and ingenuity in the application of BASIC-STATE.
Recall that a basic path formula is a path formula whose principal operator is temporal and which does not contain any additional temporal operators or path quantifiers.

The following rule enables to eliminate basic path formulas from a bigger formula:

\[
\frac{\phi(f) \rightarrow \alpha}{\phi(x)f \rightarrow \phi L \parallel \alpha}
\]

where \(x\) is the fresh variable introduced by the tester \(T\).

### Eliminating Basic Path Formulas

For a fair basic state formula \(f\), containing occurrences of the basic path formula \(\phi\), containing

\[\phi(f)(x)\]

an \(RDS\) of the following form:

\[\phi L \parallel \alpha\]
Verifying Assertional Basic State Formulas

For assertions $p$ and $q$,

$$p \land q \implies A f p$$

The dual rule $E f$-assert deals with existential assertional formulas.

For assertions $p$ and $q$,

$$p \lor q \implies E f p$$
Lecture 4: Deductive Verification of CTL

Example

The combined system is presented in the next slide:

\[ 0^f \forall x \Leftrightarrow \Diamond x \Leftarrow \quad \Diamond L \parallel \Diamond L \parallel A \]

Applying rule \( \forall f \text{-assert} \):

\[ x^f \forall x \Leftrightarrow I = \quad \Diamond L \parallel \Diamond L \parallel A \]

Introducing a tester \( L \) for \( x \):

\[ x \Diamond = \Diamond x \quad \text{for } x \Diamond \]

Introducing a tester \( L \) for \( x \) even:

\[ (x) \text{ even} \Diamond = \Diamond (x) \text{ even} \quad \text{for } (x) \text{ even} \Diamond \]

We proceed with the following goal-directed chain of reductions:

\[ \Box (x) \text{ even} \Diamond \forall x \Leftrightarrow I = \quad \Diamond L \parallel A \]

For this system, we wish to verify the formula \( A \). We proceed...
Applying rule $\forall^f$-ASRT, we conclude $\diamond x$. 

Combined System

A. Pnueli
Completing the Temporal Picture

In a paper

M. Pnueli

Z. Manna and P. Compieling the Temporal Picture

Lecture 4: Deductive Verification of CTL

A. Pnueli

Advanced Topics in Reactive Verification, NYU, Spring, 2004

Formulas. The theory can be extended to full CTL, and completeness can be based on the two reduction principles which successively eliminate basic state formulas and basic path formulas. The lectures here present an even more complete picture, where we show that the claim for completeness was based on the presentation of every temporal formula in a conjunctive form which is a conjunction of reactivity properties, where atomic and arbitrary past formulas are included.

The theory includes the two books with Zohar, where we presented a complete proof theory for LTL.