6. Symbolic CTL Model Checking
Symbolic CTL model checking using
- OBDD
- lattices
- fixpoints
MODEL CHECKING

System model

AG (start → AF heat)

Property specification

Model Checker

Property satisfied

Property violated

Jan Kofroň: Behaviour Models and Verification
MODEL CHECKING

System model

**CTL**

\( \text{AG (start } \rightarrow \text{ AF heat)} \)

Property specification

Symbolic Model Checking

Model Checker

Property satisfied

Property violated
**RECALL: LAT\(T\)ICE**

- **Lattice** \(L\) is a structure consisting of a partially ordered set \(S\) of elements where every two elements have:
  - unique *supremum* (least upper bound or join) and
  - unique *infimum* (greatest lower bound or meet)

- Set \(P(S)\) of all subsets of \(S\) forms a complete lattice.

- Each element \(E \in L\) can also be thought as a *predicate* on \(S\).

- Greatest element of \(L\) is \(S (⊤, \text{true})\).

- Least element of \(L\) is \(∅ (⊥, \text{false})\).

- \(τ : P(S) \mapsto P(S)\) is called a *predicate transformer*.
EXAMPLE: SUBSET LATTICE OF \{1, 2, 3, 4\}
Let $\tau : P(S) \leftrightarrow P(S)$ be predicate transformer

- $\tau$ is monotonic $\equiv Q \subseteq R \implies \tau(Q) \subseteq \tau(R)$
- $Q$ is fixpoint of $\tau \equiv \tau(Q) = Q$
**FIXPOINT COMPUTATION**

```plaintext
function LFP(τ: PredicateTransformer): Predicate
    Q := false
    Q' := τ(Q)
    while Q ≠ Q' do
        Q := Q'
        Q' := τ(Q)
    end while
    return Q
end function
```

Function Gfp differs just in initialization $Q := true$
Let $\tau(Q) = Q \cup \{1\}$.

What are fixpoints of $\tau$?
Let $\tau(Q) = Q \cup \{1\}$.

What are fixpoints of $\tau$?
Let $\tau(Q) = Q \cup \{1\}$.

What is the least fixpoint of $\tau$?
EXAMPLE OF FIXPOINTS

Let $\tau(Q) = Q \cup \{1\}$.

What is the least fixpoint of $\tau$?
**CTL OPERATORS AS FIXPOINTS**

- We identify CTL formula $f$ with set/predicate $\{s | M, s \models f\}$ in $P(S)$.

- $EG$ and $EU$ may be characterized as least or greatest fixpoints of an appropriate predicate transformer:
  - $EG \ q = \nu Z. \ (q \land EX \ Z)$
  - $E[p \ U \ q] = \mu Z. \ (q \lor (p \land EX \ Z))$

- The same holds for $EF$, $AG$, $AF$, $AU$, however, those operators can be expressed using $EG$, $EU$.

- Intuitively:
  - least fixpoints correspond to eventualities
  - greatest fixpoints correspond to properties that should hold forever
$M, s_0 \models \text{EG } q$

EG $q = \nu Z. (q \land \text{EX } Z)$

$\tau(Z) = \{ s : s \models q \land (\exists t : s \rightarrow t \land t \in Z) \}$
EU AS FIXPOINT

\[ M, s_0 \models E[p \cup q] \]

\[ E[p \cup q] = \mu Z. (q \lor (p \land \text{EX} Z)) \]

\[ \tau(Z) = \{ s : s \models q \} \lor \{ s : s \models p \land (\exists t : s \rightarrow t \land t \in Z) \} \]
Explicit model checking—e.g., Spin—is linear in size of generated state space
- usually exponential in size of input model
- resulting in state space explosion

Symbolic model checking operates on sets of states in each step of algorithm
- can mitigate state-space-explosion impact substantially
QUANTIFIED BOOLEAN FORMULAE

QBFs are useful in symbolic CTL model checking

Quantification does not introduce greater expressive power:

\[ \exists x \, f \equiv f|_{x=\bot} \lor f|_{x=\top} \]
\[ \forall x \, f \equiv f|_{x=\bot} \land f|_{x=\top} \]
Symbolic CTL Model Checking

General approach identical to explicit model checking
- decomposing formula into sub-formulae
- identifying sets of states satisfying particular sub-formulae

Computing states satisfying particular formula types based on manipulation with OBDDs
Computing OBDD($f$) for formula $f$ depends on top-most operand
- note that only $\neg$, $\land$, $\lor$, EX, EG, and EU are needed, others can be eliminated

- $f \in \text{AP}$: return OBDD defined for $f$
- $f : \neg g$, $f \land g$, or $f \lor g$: use logical operation upon OBDD
  - described in previous lecture
- $f = \text{EX} g$: OBDD for $\exists \langle v' \rangle (o(\langle v' \rangle) \land R(\langle v \rangle, \langle v' \rangle))$
  - $o(\langle v \rangle)$ stands for OBDD representing states satisfying formula $g$
- $f = \text{E}[f \cup g]$: compute least fixpoint $\text{E}[f \cup g] = \mu Z. (g \lor (f \land \text{EX} Z))$
  - using LfP procedure
- $f = \text{EG} f$: compute greatest fixpoint $\text{EG} f = \nu Z. (f \land \text{EX} Z)$
  - using GfP procedure
$AF = \neg EG (\neg x) \lor EG (\neg x) \lor \neg x \lor x$
\[\text{AF} \, x = \neg \text{EG} (\neg x) \]

\[
\begin{align*}
\text{EG} (\neg x) \\
\neg x \\
x
\end{align*}
\]
OBDD for states satisfying $x$:
TR:

Example

OBDD for states satisfying $x$:

OBDD for states satisfying $\neg x$:
We have OBDD for states satisfying \( \neg x \) and now, we can proceed to EG (\( \neg x \)) and compute OBDD for it.

We compute greatest fixpoint of predicate transformer: EG (\( \neg x \)) : \( \nu Z. (\neg x \land EX Z) \).

- computation starts with trivial OBDD for \( \top (Z) \).
- single step: \( Z = \neg x \land (\exists x'_0, x'_1 : Z' \land TR) \)
  - \( Z' \) denotes OBDD \( Z \) where all variables get primed (\( x \rightarrow x' \))
- if \( Z \) changes, repeat previous step, otherwise fixpoint reached and computation is over
We have OBDD for states satisfying $\neg x$ and now, we can proceed to $\text{EG} (\neg x)$ and compute OBDD for it.

We compute greatest fixpoint of predicate transformer: $\text{EG} (\neg x) : \nu Z. (\neg x \land \text{EX} Z)$.

- computation starts with trivial OBDD for $\top (Z)$.
- single step: $Z = \neg x \land (\exists x_0', x_1' : Z' \land \text{TR})$
  - $Z'$ denotes OBDD $Z$ where all variables get primed ($x \rightarrow x'$)
- if $Z$ changes, repeat previous step, otherwise fixpoint reached and computation is over

$$\text{EG} (\neg x) :$$

\[
\begin{array}{c}
\neg x \\
\top \\
\bot
\end{array}
\]

\[
\begin{array}{c}
x_0 \\
\top \\
\bot
\end{array}
\]

\[
\begin{array}{c}
\text{TR} \\
\top \\
\bot
\end{array}
\]
We have OBDD for states satisfying $\text{EG} (\neg x)$ and now, we can trivially compute its negation $\neg \text{EG} (\neg x) = \text{AF} x$. This corresponds to states 00 and 10 of Kripke structure.
During symbolic CTL model checking, all operation performed just upon OBDDs as application of logical operations and fixpoint computations.

Usually highly efficient comparing to explicit model checking.