Symbolic Model Checking Using Py-Z3
Reactive System Code satisfies Requirements:

- Reactive System Code
- Transition System
- Temporal Logic Formula $\phi$

Model Checking
Reactive System Code satisfies Requirements.

Requirements satisfies Temporal Logic Formula $\phi$.

Symbolic Model Checking

Transition System
Reactive System Code satisfies Requirements

Transition System satisfies Temporal Logic Formula

Represent $M$ using Boolean logic.
Check $M \models \phi$ by logic manipulations.
Variable Replacement

We often need to replace variables with other expressions. For a formula \( f \), variable \( v \), and expression \( e \), we write \( f[e/v] \) to indicate a new formula that is the same as \( f \) but with all occurrences of \( v \) replaced by \( e \).

Example: \( f = \neg x \land \neg y \)

\[
\begin{align*}
  f[z/x] &= \neg z \land \neg y \\
  f[T/x] &= \neg T \land \neg y \equiv F \land \neg y \equiv F \\
  f[F/y] &= \neg x \land \neg F \equiv \neg x \land T \equiv \neg x
\end{align*}
\]

We can do several variables at once:

\[
\begin{align*}
  f[(-w, F)/(x, y)] &= \neg \neg w \land \neg F = w
\end{align*}
\]
Existential Quantifier Elimination

For a formula $f$, we can “get rid” of a variable $\nu$ by

1. writing $\exists \nu : f$
2. plugging in all possible values of $\nu$ into $f$ and taking a disjunction.
Existential Quantifier Elimination

For a formula $f$, we can “get rid” of a variable $v$ by

1. writing $\exists v : f$
2. plugging in all possible values of $v$ into $f$ and taking a disjunction.

For Boolean formulas:

$$\exists v : f \equiv f[T/v] \lor f[F/v]$$
For a formula $f$, we can “get rid” of a variable $v$ by

1. writing $\exists v : f$
2. plugging in all possible values of $v$ into $f$ and taking a disjunction.

For Boolean formulas:

$$\exists v : f \equiv f[T/v] \lor f[F/v]$$

Example: $f = \neg x \land \neg y$
Existential Quantifier Elimination

For a formula $f$, we can “get rid” of a variable $\nu$ by

1. writing $\exists \nu : f$
2. plugging in all possible values of $\nu$ into $f$ and taking a disjunction.

For Boolean formulas:

$$\exists \nu : f \equiv f[T/\nu] \lor f[F/\nu]$$

Example: $f = \neg x \land \neg y$

$$\exists y : f \equiv : f[T/y] \lor f[F/y]$$
Existential Quantifier Elimination

For a formula $f$, we can “get rid” of a variable $v$ by

1. writing $\exists v : f$
2. plugging in all possible values of $v$ into $f$ and taking a disjunction.

For Boolean formulas:

$$\exists v : f \equiv f[T/v] \lor f[F/v]$$

Example: $f = \neg x \land \neg y$

$$\exists y : f \equiv : f[T/y] \lor f[F/y]$$

$$\equiv (\neg x \land \neg T) \lor (\neg x \land \neg F)$$
Existential Quantifier Elimination

For a formula $f$, we can “get rid” of a variable $v$ by

1. writing $\exists v : f$
2. plugging in all possible values of $v$ into $f$ and taking a disjunction.

For Boolean formulas:

$$\exists v : f \equiv f[T/v] \lor f[F/v]$$

Example: $f = \neg x \land \neg y$

$$\exists y : f \equiv f[T/y] \lor f[F/y]$$

$$\equiv (\neg x \land \neg T) \lor (\neg x \land \neg F)$$

$$\equiv F \lor \neg x \equiv \neg x$$

No more $y$
Explicit Model Representation

The transition system $\mathcal{M}$ is specified by literally listing out all of the pieces.
Explicit Model Representation

The transition system $\mathcal{M}$ is specified by literally listing out all of the pieces.
Explicit Model Representation

The transition system $\mathcal{M}$ is specified by literally listing out all of the pieces.

States: $S = \{0, 1, 2, 3\}$
The transition system $\mathcal{M}$ is specified by literally listing out all of the pieces.

States: $S = \{0, 1, 2, 3\}$

Initial States: $I = \{0\}$
The transition system $M$ is specified by literally listing out all of the pieces.

States: $S = \{0, 1, 2, 3\}$

Initial States: $I = \{0\}$

Transitions:

$$R = \{ (0, 1), (0, 2), (1, 3), (2, 3), (1, 0), (2, 0), (3, 1), (3, 2) \}$$
The transition system $\mathcal{M}$ is specified by literally listing out all of the pieces.

States: $S = \{0, 1, 2, 3\}$

Initial States: $I = \{0\}$

Transitions:

$$R = \{ (0, 1), (0, 2), (1, 3), (2, 3), (1, 0), (2, 0), (3, 1), (3, 2) \}$$

Atomic Propositions: $AP = \{p, q, r\}$
The transition system $\mathcal{M}$ is specified by literally listing out all of the pieces.

States: $S = \{0, 1, 2, 3\}$

Initial States: $I = \{0\}$

Transitions:

$$R = \{ (0, 1), (0, 2), (1, 3), (2, 3), (1, 0), (2, 0), (3, 1), (3, 2) \}$$

Atomic Propositions: $AP = \{p, q, r\}$

Labelling Function $\mathcal{L} : S \rightarrow \mathcal{P}(AP)$

$\mathcal{L}(0) = \{r\}$

$\mathcal{L}(2) = \{r, q\}$

$\mathcal{L}(1) = \{r\}$

$\mathcal{L}(1) = \{p, q\}$
Symbolic Model Representation

Represent $M$ using Boolean logic.
Symbolic Model Representation

Represent $\mathcal{M}$ using Boolean logic.

\[
\begin{array}{c}
\{r\} \\
0 \\
\{r, q\} \\
2 \\
1 \\
\{p, q\} \\
3 \\
\end{array}
\]
Symbolic Model Representation

Represent $M$ using Boolean logic.

```
\{r\} \rightarrow 0 \rightarrow 1 \rightarrow \{r\} \\
\{r, q\} \rightarrow 2 \rightarrow 3 \rightarrow \{p, q\}
```

<table>
<thead>
<tr>
<th>States</th>
<th>binary</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$x$ 0</td>
</tr>
<tr>
<td>1</td>
<td>$y$ 1</td>
</tr>
<tr>
<td>2</td>
<td>$x$ 1</td>
</tr>
<tr>
<td>3</td>
<td>$y$ 1</td>
</tr>
</tbody>
</table>
Symbolic Model Representation

Represent $M$ using Boolean logic.

\begin{center}
\begin{tabular}{c|c|c|c|c|c|c|c}
\hline
\textbf{States} & \textbf{binary} & \textbf{truth values} \\
\hline
& $x$ & $y$ & $x$ & $y$ \\
\hline
0 & 0 & 0 & $F$ & $F$ \\
1 & 0 & 1 & $F$ & $T$ \\
2 & 1 & 0 & $T$ & $F$ \\
3 & 1 & 1 & $T$ & $T$ \\
\hline
\end{tabular}
\end{center}
Symbolic Model Representation

Represent $M$ using Boolean logic.

Boolean state variables

$V = \{x, y\}$

<table>
<thead>
<tr>
<th>States</th>
<th>binary</th>
<th>truth values</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$x$</td>
<td>$y$</td>
</tr>
<tr>
<td>0</td>
<td>0 0</td>
<td>$F$ $F$</td>
</tr>
<tr>
<td>1</td>
<td>0 1</td>
<td>$F$ $T$</td>
</tr>
<tr>
<td>2</td>
<td>1 0</td>
<td>$T$ $F$</td>
</tr>
<tr>
<td>3</td>
<td>1 1</td>
<td>$T$ $T$</td>
</tr>
</tbody>
</table>
Symbolic Model Representation

Represent $\mathcal{M}$ using Boolean logic.

### Boolean state variables

$$V = \{x, y\}$$

### States and truth values

<table>
<thead>
<tr>
<th>States</th>
<th>binary</th>
<th>truth values</th>
<th>Boolean formula</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$x$</td>
<td>$y$</td>
<td>$\neg x \land \neg y$</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>$F$</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>$F$</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>$T$</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>$T$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$\neg x \land y$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$x \land \neg y$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$x \land y$</td>
</tr>
</tbody>
</table>
Symbolic Model Representation

Represent $\mathcal{M}$ using Boolean logic.

<table>
<thead>
<tr>
<th>States</th>
<th>binary</th>
<th>truth values</th>
<th>Boolean formula</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$x$</td>
<td>$y$</td>
<td>$x$</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>$F$</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>$F$</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>$T$</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>$T$</td>
</tr>
</tbody>
</table>

The transition diagram shows the states and transitions between them, with symbols $\{p, q\}$ and $\{r\}$.

The states are represented as $FF$, $FT$, $TF$, and $TT$, with corresponding binary and truth values.
Symbolic Model Representation

Represent $M$ using Boolean logic.

Initial State: $\neg x \land \neg y$

<table>
<thead>
<tr>
<th>States</th>
<th>binary</th>
<th>truth values</th>
<th>Boolean formula</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$x$</td>
<td>$y$</td>
<td>$x$</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>$F$</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>$F$</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>$T$</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>$T$</td>
</tr>
</tbody>
</table>
Symbolic Model Representation

Represent $\mathcal{M}$ using Boolean logic.

Initial State: $\neg x \land \neg y$

Atomic Propositions: $AP = \{p, q, r\}$

<table>
<thead>
<tr>
<th>States</th>
<th>binary</th>
<th>truth values</th>
<th>Boolean formula</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$x$</td>
<td>$y$</td>
<td>$x$</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>F</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>F</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>T</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>T</td>
</tr>
</tbody>
</table>
Symbolic Model Representation

Represent $M$ using Boolean logic.

Initial State: $\neg x \land \neg y$

Atomic Propositions: $AP = \{p, q, r\}$

Labelling Function $\mathcal{L}: S \rightarrow \mathcal{P}(AP)$

<table>
<thead>
<tr>
<th>States</th>
<th>binary</th>
<th>truth values</th>
<th>Boolean formula</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$x$</td>
<td>$y$</td>
<td>$x$ $y$</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>$F$ $F$</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>$F$ $T$</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>$T$ $F$</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>$T$ $T$</td>
</tr>
</tbody>
</table>
Symbolic Model Representation

Represent $\mathcal{M}$ using Boolean logic.

Initial State: $\neg x \land \neg y$

Atomic Propositions: $AP = \{p, q, r\}$

Labelling Function $\mathcal{L} : S \rightarrow \mathcal{P}(AP)$

### States

<table>
<thead>
<tr>
<th>States</th>
<th>binary</th>
<th>truth values</th>
<th>Boolean formula</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$x$</td>
<td>$y$</td>
<td>$x$</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Symbolic Model Representation

Represent $\mathcal{M}$ using Boolean logic.

Initial State: $\neg x \land \neg y$

Atomic Propositions: $AP = \{p, q, r\}$

Labelling Function $\mathcal{L} : S \rightarrow 2^{AP}$

$\mathcal{L} : AP \rightarrow \mathcal{F}(x, y)$

<table>
<thead>
<tr>
<th>States</th>
<th>binary</th>
<th>truth values</th>
<th>Boolean formula</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$x$</td>
<td>$y$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$x$</td>
<td>$y$</td>
<td></td>
</tr>
<tr>
<td>$0$</td>
<td>0</td>
<td>0</td>
<td>$\neg x \land \neg y$</td>
</tr>
<tr>
<td>$1$</td>
<td>0</td>
<td>1</td>
<td>$\neg x \land y$</td>
</tr>
<tr>
<td>$2$</td>
<td>1</td>
<td>0</td>
<td>$x \land \neg y$</td>
</tr>
<tr>
<td>$3$</td>
<td>1</td>
<td>1</td>
<td>$x \land y$</td>
</tr>
</tbody>
</table>

Diagram of states and transitions:
Symbolic Model Representation

Represent $M$ using Boolean logic.

Initial State: $\neg x \land \neg y$

Atomic Propositions: $AP = \{p, q, r\}$

Labelling Function $\mathcal{L} : S \to \mathcal{P}(AP)$

$p \equiv x \land y$
$q \equiv x$
$r \equiv \neg(x \land y) \equiv \neg p$

<table>
<thead>
<tr>
<th>States</th>
<th>binary</th>
<th>truth values</th>
<th>Boolean formula</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>$\neg x \land \neg y$</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>$F$</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td>$F$</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td>$F$</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>$T$</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>$T$</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>$T$</td>
</tr>
</tbody>
</table>
Symbolic Model Representation

Represent $\mathcal{M}$ using Boolean logic.
Symbolic Model Representation

Represent $\mathcal{M}$ using Boolean logic.

Transitions:
Let the “next” state variables be $V' = \{x', y'\}$
Represent $\mathcal{M}$ using Boolean logic.

Transitions:
Let the "next" state variables be $V' = \{x', y'\}$

\[
R \equiv (x' = x \land y' = \neg y) \lor (x' = \neg x \land y' = y)
\]
Symbolic Model Representation

Represent $\mathcal{M}$ using Boolean logic.

Transitions:
Let the “next” state variables be $V' = \{x', y'\}$

\[ R \equiv (x' = x \land y' = \neg y) \lor (x' = \neg x \land y' = y) \]

“we can get from one state to the next by keeping one variable the same and negating the other”
Symbolic Model Representation

Represent $\mathcal{M}$ using Boolean logic.

Transitions:
Let the “next” state variables be $V' = \{x', y'\}$

\[
R \equiv (x' = x \land y' = \neg y) \lor (x' = \neg x \land y' = y)
\]

Explicit transitions:

<table>
<thead>
<tr>
<th>(0, 1)</th>
<th>(2, 3)</th>
<th>(1, 3)</th>
<th>(0, 2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1, 0)</td>
<td>(3, 2)</td>
<td>(3, 1)</td>
<td>(2, 0)</td>
</tr>
</tbody>
</table>

“we can get from one state to the next by keeping one variable the same and negating the other”
Reactive System Code satisfies $\models$ Requirements

Transition System satisfies $\models$ Temporal Logic Formula $\phi$

Symbolic Model Checking

Represent $M$ using Boolean logic. Check $M \models \phi$ by logic manipulations.
The Algorithm for $EX \phi$

After labelling all states $s$ that satisfy $\phi$, label and state $s'$ with $EX\phi$ if there is a transition from $s'$ to $s$. 

![Diagram with states and transitions](image)
The Algorithm for $EX\phi$

After labelling all states $s$ that satisfy $\phi$, label and state $s'$ with $EX\phi$ if there is a transition from $s'$ to $s$. 

![Diagram showing state transitions and labels $EX\phi$, $s'$, $s$, and $\phi$.]
The Algorithm for $EX \phi$

After labelling all states $s$ that satisfy $\phi$, label and state $s'$ with $EX\phi$ if there is a transition from $s'$ to $s$.

Call this process $SAT_{EX}(\phi)$
Symbolic Model Checking

How to compute $EX \phi$ symbolically.
Symbolic Model Checking

How to compute $EX \phi$ symbolically.

$$EX \phi \equiv \exists V' \ R \land \phi[V'/V]$$
Symbolic Model Checking

How to compute $EX \ \phi$ symbolically.

$$EX \ \phi \equiv \exists V' \ R \ \land \ \phi[V'/V]$$

exists a path where $\phi$ holds in the next state.
Symbolic Model Checking

How to compute $EX \phi$ symbolically.

$$EX \phi \equiv \exists V' \ R \land \phi[V' / V]$$

exists a path where $\phi$ holds in the next state

there is some assignment for the next state variables
Symbolic Model Checking

How to compute $EX \phi$ symbolically.

$$EX \phi \equiv \exists V' \quad R \land \phi[V'/V]$$

- exists a path where $\phi$ holds in the next state
- there is some assignment for the next state variables
- obeys the transition relation
Symbolic Model Checking

How to compute $EX \phi$ symbolically.

$EX \phi \equiv \exists V' \ R \land \phi[V'/V]$

- exists a path where $\phi$ holds in the next state
- there is some assignment for the next state variables
- obeys the transition relation
- $\phi$ holds when variables are updated with the new state variables
Symbolic Model Checking

Initial State: \( \neg x \land \neg y \)

Atomic Propositions: \( AP = \{ p, q, r \} \)

Labelling Function \( \mathcal{L} : AP \rightarrow \mathcal{F}(x, y) \)

\[
p \equiv x \land y \\
q \equiv x \\
r \equiv \neg(x \land y)
\]

Transition Relation:

\[
R \equiv (x' = x \land y' = \neg y) \lor (x' = \neg x \land y' = y)
\]
Symbolic Model Checking

Initial State: \( \neg x \land \neg y \)

Atomic Propositions: \( AP = \{ p, q, r \} \)

Labelling Function \( \mathcal{L} : AP \rightarrow F(x, y) \)

\[
\begin{align*}
p & \equiv x \land y \\
qu & \equiv x \\
r & \equiv \neg(x \land y)
\end{align*}
\]

Transition Relation:

\[
R \equiv (x' = x \land y' = \neg y) \lor (x' = \neg x \land y' = y)
\]

Let's compute \( EX \ p \)
Symbolic Model Checking

Initial State: \( \neg x \land \neg y \)

Atomic Propositions: \( AP = \{p, q, r\} \)

Labelling Function \( \mathcal{L} : AP \rightarrow \mathcal{F}(x, y) \)

\[
p \equiv x \land y \quad q \equiv x \quad r \equiv \neg(x \land y)
\]

Transition Relation:

\[
R \equiv (x' = x \land y' = \neg y) \lor (x' = \neg x \land y' = y)
\]

Let's compute \( EX \ p \)
Symbolic Model Checking

Initial State: \( \neg x \land \neg y \)

Atomic Propositions: \( AP = \{ p, q, r \} \)

Labelling Function : \( L: AP \rightarrow \mathcal{F}(x, y) \)

- \( p \equiv x \land y \)
- \( q \equiv x \)
- \( r \equiv \neg(x \land y) \)

Transition Relation:

\[
R \equiv (x' = x \land y' = \neg y) \lor (x' = \neg x \land y' = y)
\]

Let’s compute \( EX \ p \)

\[
EX \ p \equiv \exists V' \ R \land p[V' /V]
\]
Symbolic Model Checking

Initial State: $\neg x \land \neg y$

Atomic Propositions: $AP = \{p, q, r\}$

Labelling Function $\mathcal{L} : AP \rightarrow \mathcal{F}(x, y)$

\[
p \equiv x \land y \quad q \equiv x \quad r \equiv \neg(x \land y)
\]

Transition Relation:

\[
R \equiv (x' = x \land y' = \neg y) \lor (x' = \neg x \land y' = y)
\]

Let's compute $EX \ p$

\[
EX \ p \equiv \exists V' \ R \land p[V'/V]
\]

\[
EX \ p \equiv \exists x', y' \ (x' = x \land y' = \neg y) \lor (x' = \neg x \land y' = y) \land (x' \land y')
\]
Symbolic Model Checking

Initial State: \(\neg x \land \neg y\)

Atomic Propositions: \(AP = \{p, q, r\}\)

Labelling Function \(L : AP \rightarrow F(x, y)\)
\[p \equiv x \land y\]
\[q \equiv x\]
\[r \equiv \neg(x \land y)\]

Transition Relation:
\[R \equiv (x' = x \land y' = \neg y) \lor (x' = \neg x \land y' = y)\]

Let's compute \(EX\ p\)

\[EX\ p \equiv \exists V'\ R \land p[V'/V]\]
\[EX\ p \equiv \exists x', y'\ (x' = x \land y' = \neg y) \lor (x' = \neg x \land y' = y) \land (x' \land y')\]

...some Boolean simplifications ...
Initial State: \( \neg x \land \neg y \)

Atomic Propositions: \( AP = \{p, q, r\} \)

Labelling Function \( \mathcal{L} : AP \to \mathcal{F}(x, y) \)
\[
\begin{align*}
p & \equiv x \land y \\
q & \equiv x \\
r & \equiv \neg(x \land y)
\end{align*}
\]

Transition Relation:
\[
R \equiv (x' = x \land y' = \neg y) \lor (x' = \neg x \land y' = y) \land (x' \land y')
\]

Let's compute \( EX \ p \)

\[
EX \ p \equiv \exists V' \ R \land p[V'/V]
\]
\[
EX \ p \equiv \exists x', y' \ (x' = x \land y' = \neg y) \lor (x' = \neg x \land y' = y) \land (x' \land y')
\]

...some Boolean simplifications ...

\[
EX \ p \equiv \exists x', y' \ (x' \land x \land y' \land \neg y) \lor (x' \land \neg x \land y' \land y)
\]
Symbolic Model Checking

Initial State: \( \neg x \land \neg y \)

Atomic Propositions: \( AP = \{p, q, r\} \)

Labelling Function \( \mathcal{L} : AP \rightarrow \mathcal{F}(x, y) \)

\[
\begin{align*}
p & \equiv x \land y \\
q & \equiv x \\
r & \equiv \neg(x \land y)
\end{align*}
\]

Transition Relation:

\[
R \equiv (x' = x \land y' = \neg y) \lor (x' = \neg x \land y' = y)
\]

Let's compute \( EX \ p \)

\[
EX \ p \equiv \exists V' \ R \land p[V' / V]
\]

\[
EX \ p \equiv \exists x', y' \ (x' = x \land y' = \neg y) \lor (x' = \neg x \land y' = y) \land (x' \land y')
\]

…some Boolean simplifications …

\[
EX \ p \equiv \exists x', y' \ (x' \land x \land y' \land \neg y) \lor (x' \land \neg x \land y' \land y)
\]

…existential quantifier elimination …
Symbolic Model Checking

Initial State: \( \neg x \land \neg y \)

Atomic Propositions: \( AP = \{ p, q, r \} \)

Labelling Function \( L : AP \rightarrow F(x, y) \)
\[ p \equiv x \land y \quad q \equiv x \quad r \equiv \neg(x \land y) \]

Transition Relation:
\[ R \equiv (x' = x \land y' = \neg y) \lor (x' = \neg x \land y' = y) \land (x' \land y') \]

Let's compute \( EX \ p \)

\[
EX \ p \equiv \exists V' \ R \land p[ V' / V ]
\]

\[
EX \ p \equiv \exists x', y' \ (x' = x \land y' = \neg y) \lor (x' = \neg x \land y' = y) \land (x' \land y')
\]

...some Boolean simplifications ...

\[
EX \ p \equiv \exists x', y' \ (x' \land x \land y' \land \neg y) \lor (x' \land \neg x \land y' \land y)
\]

...existential quantifier elimination ...

\[
EX \ p \equiv (x \land \neg y) \lor (\neg x \land y)
\]
Symbolic Model Checking

Initial State: \( \neg x \land \neg y \)

Atomic Propositions: \( AP = \{ p, q, r \} \)

Labelling Function
\[
\mathcal{L} : AP \rightarrow \mathcal{F}(x, y) \\
p \equiv x \land y \\
q \equiv x \\
r \equiv \neg (x \land y)
\]

Transition Relation:
\[
R \equiv (x' = x \land y' = \neg y) \lor (x' = \neg x \land y' = y) \land (x' \land y')
\]

Let's compute \( EX \ p \)

\[
EX \ p \equiv \exists x', y' \ (x' = x \land y' = \neg y) \lor (x' = \neg x \land y' = y) \land (x' \land y')
\]

...some Boolean simplifications ...

\[
EX \ p \equiv \exists x', y' \ (x' \land x \land y' \land \neg y) \lor (x' \land \neg x \land y' \land y)
\]

...existential quantifier elimination ...

\[
EX \ p \equiv (x \land \neg y) \lor (\neg x \land y)
\]

Which states does this formula represent?
Symbolic Model Checking

Initial State: \( \neg x \land \neg y \)

Atomic Propositions: \( AP = \{ p, q, r \} \)

Labelling Function \( \mathcal{L} : AP \rightarrow \mathcal{F}(x, y) \)
\[
    p \equiv x \land y \\
    q \equiv x \\
    r \equiv \neg(x \land y)
\]

Transition Relation:
\[
    R \equiv (x' = x \land y' = \neg y) \lor (x' = \neg x \land y' = y) \land (x' \land y')
\]

Let's compute \( \text{EX } p \)

\[
    \text{EX } p \equiv \exists V' \ R \land p[V' / V]
\]

\[
    \text{EX } p \equiv \exists x', y' \ (x' = x \land y' = \neg y) \lor (x' = \neg x \land y' = y) \land (x' \land y')
\]

...some Boolean simplifications ...

\[
    \text{EX } p \equiv \exists x', y' \ (x' \land x \land y' \land \neg y) \lor (x' \land \neg x \land y' \land y)
\]

...existential quantifier elimination ...

\[
    \text{EX } p \equiv (x \land \neg y) \lor (\neg x \land y)
\]
Symbolic Model Checking

All of the boolean operations we have described for performing symbolic model checking (conjunction, disjunction, existential variable elimination) can be accomplished by:

1. Boolean algebra
2. Using BDDs
3. Using a theorem prover
We can translate the \( EX \phi \) formula into Z3.

\[
EX \phi \equiv \exists V' \ R \land \phi[V' / V]
\]

Example: \( R \equiv (x' = x \land y' = \neg y) \lor (x' = \neg x \land y' = y) \)

\( \phi \equiv p \equiv x \land y \)

(declare-const x Bool)
(declare-const y Bool)
(assert
(exists ((x_ Bool) (y_ Bool))
(and
(or
(and (= x_ x) (= y_ (not y)))
(and (= x_ (not x)) (= y_ y)))
(and x_ y_)))
(apply qe)
(check-sat)
We can translate the $EX \phi$ formula into Z3.

$$EX \phi \equiv \exists V' R \land \phi[V'/V]$$

Example: $R \equiv (x' = x \land y' = \neg y) \lor (x' = \neg x \land y' = y)$

$\phi \equiv p \equiv x \land y$

```
(declare-const x Bool)
(declare-const y Bool)
(assert
(exists ((x_ Bool) (y_ Bool))
(and
(or
(and (= x_ x) (= y_ (not y)))
(and (= x_ (not x)) (= y_ y)))
(and x_ y_))))
(apply qe)
(check-sat)
```
Symb. Mod. Check. using a Theorem Prover

We can translate the $EX \phi$ formula into Z3.

$$EX \phi \equiv \exists V' \ [R] \land \phi[V'/V]$$

Example: $R \equiv (x' = x \land y' = \neg y) \lor (x' = \neg x \land y' = y)$

$\phi \equiv p \equiv x \land y$

(declare-const x Bool)
(declare-const y Bool)
(assert
(exists ((x_ Bool) (y_ Bool))
(and
(or
(and (= x_ x) (= y_ (not y)))
(and (= x_ (not x)) (= y_ y)))
(and x_ y_))))
(apply qe)
(check-sat)
We can translate the $EX \phi$ formula into Z3.

$$EX \phi \equiv \exists V' \ R \land \phi[V'/V]$$

Example:

$$R \equiv (x' = x \land y' = \neg y) \lor (x' = \neg x \land y' = y)$$

$$\phi \equiv p \equiv x \land y$$

```
(declare-const x Bool)
(declare-const y Bool)
(assert
(exists ((x_ Bool) (y_ Bool))
(and
(or
(and (= x_ x) (= y_ (not y)))
(and (= x_ (not x)) (= y_ y)))
(and x_ y_))))
(apply qe)
(check-sat)
```
We can translate the $EX \phi$ formula into Z3.

$$EX \phi \equiv \exists V' \ R \land \phi[V'/V]$$

Example: $R \equiv (x' = x \land y' = \neg y) \lor (x' = \neg x \land y' = y)$

$$\phi \equiv p \equiv x \land y$$

(declare-const x Bool)
(declare-const y Bool)
(assert
(exists ((x_ Bool) (y_ Bool))
(and
(or
(and (= x_ x) (= y_ (not y)))
(and (= x_ (not x)) (= y_ y)))
(and x_ y_)))
(apply qe)
(check-sat)
We can translate the $EX \phi$ formula into Z3.

$$EX \phi \equiv \exists V' \ R \land \phi[V'/V]$$

Example: 

$$R \equiv (x' = x \land y' = \neg y) \lor (x' = \neg x \land y' = y)$$

$$\phi \equiv p \equiv x \land y$$

```lisp
(declare-const x Bool)
(declare-const y Bool)
(assert
(exists ((x_ Bool) (y_ Bool))
    (and
        (or
            (and (= x_ x) (= y_ (not y)))
            (and (= x_ (not x)) (= y_ y)))
        (and x_ y_))))
(apply qe)
(check-sat)
```

$$\phi \equiv p \equiv x \land y$$