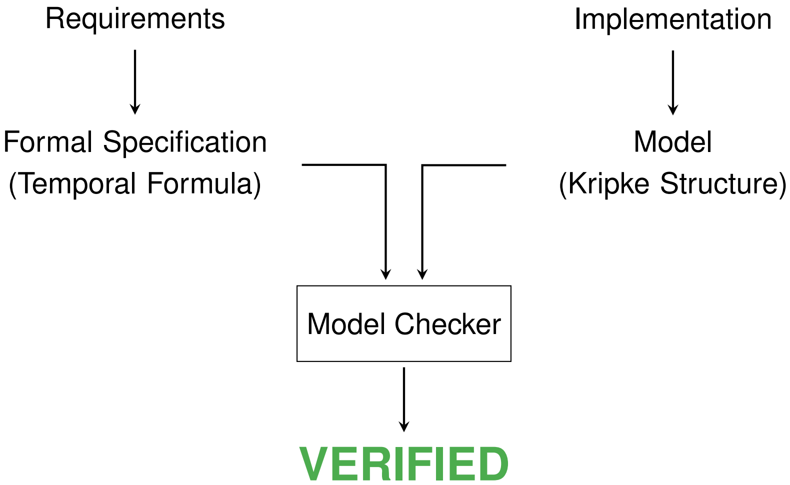
SAT-BASED BOUNDED MODEL CHECKING

Formal Models SS19

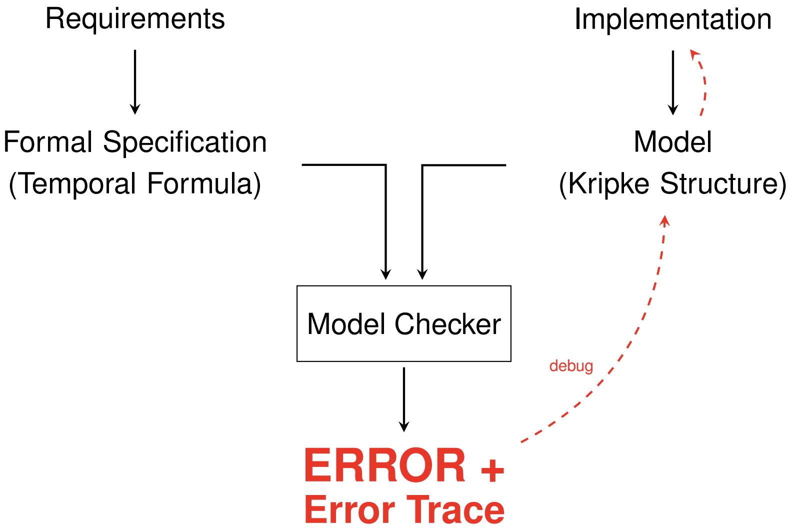
Martina Seidl
Institute for Formal Models and Verification

JOHANNES KEPLER UNIVERSITY LINZ

Model Checking



Model Checking



Types of Model Checking

General question: Given a system K and a property p, does p hold for K (i.e., for all initial states of K) ?

- Explicit state model checking
  - enumeration of the state space
  - state explosion problem
- Symbolic model checking
  - representation of model checking problem as logical formula (e.g., in propositional logic (SAT) or QBF)

Some Properties

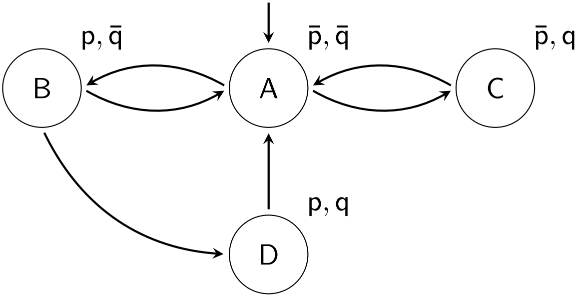
- **Reachability**: property p holds in one reachable state
- **Invariant**: property p holds in all reachable states
- **Safety**: some bad property p never holds
  "something bad will never happen"
- **Liveness**: something good will eventually happen
- **Fairness**: under certain conditions, some property holds repeatedly

Example: Mutual Exclusion

Given two processes P and Q which share a resource R.

- If R is accessed by P, then property p is true.
- If R is accessed by Q, then property q is true.

The behavior of P and Q is modeled by this Kripke structure:



Question: Does $F(p \wedge q)$ hold?

Limboole

- SAT-solver for formulas in non-CNF
- available at `http://fmv.jku.at/limboole/`
- input format in BNF:

$$
\begin{aligned}
\langle expr \rangle &::= \langle iff \rangle \\
\langle iff \rangle &::= \langle implies \rangle \mid \langle implies \rangle \text{ “}\texttt{<->}\text{” } \langle implies \rangle \\
\langle implies \rangle &::= \langle or \rangle \mid \langle or \rangle \text{ “}\texttt{->}\text{” } \langle or \rangle \mid \langle or \rangle \text{ “}\texttt{<-}\text{” } \langle or \rangle \\
\langle or \rangle &::= \langle and \rangle \mid \langle and \rangle \text{ “}\texttt{|}\text{” } \langle and \rangle \\
\langle and \rangle &::= \langle not \rangle \mid \langle not \rangle \text{ “}\texttt{\&}\text{” } \langle not \rangle \\
\langle not \rangle &::= \langle basic \rangle \mid \text{“}\texttt{!}\text{” } \langle not \rangle \\
\langle basic \rangle &::= \langle var \rangle \mid \text{“}\texttt{(}\text{” } \langle expr \rangle \text{ “}\texttt{)}\text{”}
\end{aligned}
$$

where 'var' is a string over letters, digits, and

`- _ . [ ] $ @`

Symbolic Encoding of Kripke Structures

Given Kripke structure $K = (S, I, T, L)$ over $\mathcal{A} = \{a_1, \ldots, a_n\}$.

1. Introduce sets $\mathcal{A}' = \{a'_1, \ldots, a'_n\}$ and $\mathcal{A}'' = \{a''_1, \ldots, a''_n\}$ for the **definition of one transition step $\mathcal{T}$ over $\mathcal{A}'$ and $\mathcal{A}''$**.
2. Associate each state $s \in S$ with two conjunctions of literals $current(s)$ and $next(s)$:[1]
   - $current(s) := (l_1 \wedge \ldots \wedge l_n)$
     such that $l_i = a'_i$ if $a_i \in L(s)$ else $l_i = \bar{a}'_i$;
   - $next(s) := (k_1 \wedge \ldots \wedge k_n)$
     such that $k_i = a''_i$ if $a_i \in L(s)$ else $k_i = \bar{a}''_i$.
3. Define prop. formula $\mathcal{T}$ over $\mathcal{A}', \mathcal{A}''$ such that $\forall s_i, s_j \in S$ $(\mathcal{T} \wedge current(s_i) \wedge next(s_j))$ **is satisfiable iff** $(s_i, s_j) \in T$.

[1] note: the mapping “state to conjunction” has to be bijective

Naive Encoding of Kripke Structures in SAT

Let $K = (S, I, T, L)$ be a Kripke structure over $\mathcal{A}$.

$\mathcal{T} := \top$

while $S \neq \emptyset$ **do**

  select $s \in S$

  $S := S \backslash \{s\}$

  $N := \bot$

  for all $(s, t) \in T$ **do**

    $N := N \vee next(t)$

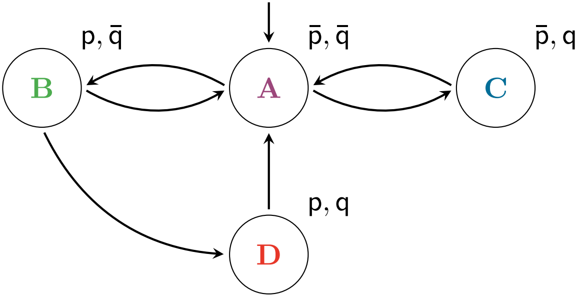
  end for

  $\mathcal{T} := \mathcal{T} \wedge (current(s) \rightarrow N)$

end while

return $\mathcal{T}$

Naive Encoding of Kripke Structures in SAT



$$
\begin{aligned}
\mathcal{T} \quad := \quad & \top & \wedge \\
& (\bar{p} \wedge \bar{q}) \rightarrow (\bot \vee (\bar{p}' \wedge q') \vee (p' \wedge \bar{q}')) & \wedge \\
& (p \wedge \bar{q}) \rightarrow (\bot \vee (\bar{p}' \wedge \bar{q}') \vee (p' \wedge q')) & \wedge \\
& (\bar{p} \wedge q) \rightarrow (\bot \vee (\bar{p}' \wedge \bar{q}')) & \wedge \\
& (p \wedge q) \rightarrow (\bot \vee (\bar{p}' \wedge \bar{q}')) &
\end{aligned}
$$

Naive Encoding of Kripke Structures in SAT

Encoding in Limboole syntax:

```
((!p & !q)  -> (!p-next & q-next) | (p-next & !q-next)) &
((p & !q)   -> (!p-next & !q-next) | (p-next & q-next)) &
((!p & q)   -> (!p-next & !q-next)) &
((p & q)   -> (!p-next & !q-next))
```

```
> limboole limboole/mutual.boole -s
% SATISFIABLE formula (satisfying assignment follows)

p = 0
q = 0
p-next = 0
q-next = 1
```

Symbolic Encoding of Kripke Structures

Alternative encoding of transition function:

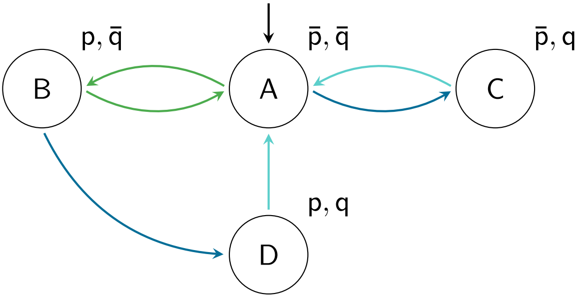
Successor states p', q':

$(p' \leftrightarrow (\bar{p} \wedge \bar{q}) \wedge q' \leftrightarrow 0)$

$\vee$

$(p' \leftrightarrow (p \wedge \bar{q}) \wedge q' \leftrightarrow \bar{q})$

p	q	p'	q'	or	p'	q'
0	0	1	0		0	1
0	1	0	0		0	0
1	0	0	0		1	1
1	1	0	0		0	0



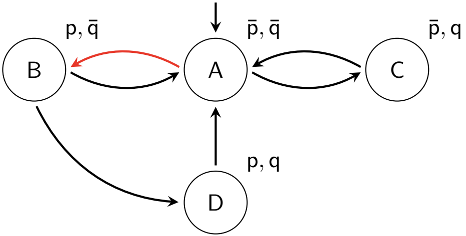
Example: One Step

Encoding in Limboole syntax:

```
(((p-next <-> (!p & !q)) & (!q-next)) |
((p-next <-> (p & !q)) & (q-next <-> !q)))

> limboole -s mutual2.boole
% SATISFIABLE formula (satisfying assignment follows)

p = 0
q = 0
p-next = 1
q-next = 0
```



Multiple Transition Steps

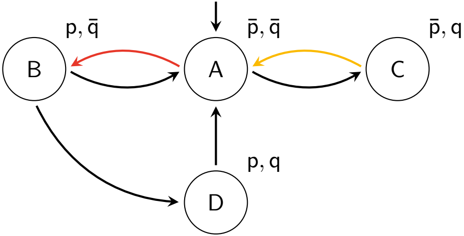
- $\mathcal{T}$ over $\mathcal{A}'$ and $\mathcal{A}''$ defines one transition step
  - we also write $\mathcal{T}(s_0, s_1)$ indicating that we can go from state s_0 to a state s_1
- $\mathcal{T}$ over $\mathcal{A}''$ and $\mathcal{A}'''$ defines one transition step
  - we also write $\mathcal{T}(s_1, s_2)$ indicating that we can go from state a s_1 to a state s_2
- $\mathcal{T}(s_0, s_1) \wedge \mathcal{T}(s_1, s_2)$ defines two transition steps from a state s_0 to a state s_1
- Example (previous slides):
  $(((\mathsf{p}' \leftrightarrow (\bar{\mathsf{p}} \wedge \bar{\mathsf{q}})) \wedge (\mathsf{q}' \leftrightarrow 0)) \vee ((\mathsf{p}' \leftrightarrow (\mathsf{p} \wedge \bar{\mathsf{q}})) \wedge (\mathsf{q}' \leftrightarrow \bar{\mathsf{q}}))) \quad \wedge$
  $(((\mathsf{p}'' \leftrightarrow (\bar{\mathsf{p}}' \wedge \bar{\mathsf{q}}')) \wedge (\mathsf{q}'' \leftrightarrow 0)) \vee ((\mathsf{p}'' \leftrightarrow (\mathsf{p}' \wedge \bar{\mathsf{q}}')) \wedge (\mathsf{q}'' \leftrightarrow \bar{\mathsf{q}}')))$

Example: Two Steps

Encoding in Limboole syntax:

```
((((p-next <-> (!p & !q)) & (!q-next))  |
((p-next <-> (p & !q)) & (q-next <-> !q)))) &
((((p-next2 <-> (!p-next & !q-next)) & (!q-next2)) |
((p-next2 <-> (p-next & !q-next)) & (q-next2 <-> !q-next))))
```

```
> limboole -s mutual2-twoSteps.boole
% SATISFIABLE formula (satisfying assignment follows)
p = 0
q = 1
p-next = 0
q-next = 0
p-next2 = 1
q-next2 = 0
```

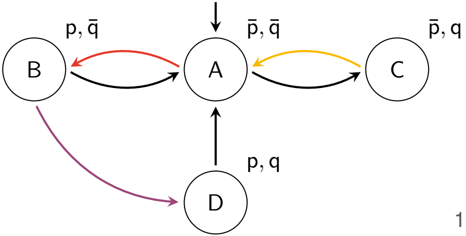


Example: Three Steps

Encoding in Limboole syntax:

```
((((p-next <-> (!p & !q)) & (!q-next))  |
((p-next <-> (p & !q)) & (q-next <-> !q)))) &
((((p-next2 <-> (!p-next & !q-next)) & (!q-next2)) |
((p-next2 <-> (p-next & !q-next)) & (q-next2 <-> !q-next)))) &
((((p-next3 <-> (!p-next2 & !q-next2)) & (!q-next3)) |
((p-next3 <-> (p-next2 & !q-next2)) & (q-next3 <-> !q-next2))))
```

```
limboole -s mutual2-threeSteps.boole
% SATISFIABLE formula (satisfying assignment follows)
p = 0
q = 1
p-next = 0
q-next = 0
p-next2 = 1
q-next2 = 0
p-next3 = 1
q-next3 = 1
```



Bounded Model Checking (Safety)

- Given a Kripke structure K. Is there a path of length k to a **bad state** s, i.e., a certain property p is violated in s?
- In other words: there is a path where $\mathsf{G}p$ does not hold in K
- Observation: if $\mathsf{G}p$ does not hold in K, there is a **finite counter-example.**
- Idea: consider paths of fixed length k
  - encode problem to propositional formula ϕ
  - pass problem to SAT solver
  - ϕ is true $\Leftrightarrow$ model of ϕ is counter-example
  - if ϕ is false, then increase k

Bounded Model Checking (Safety)

A bounded model checking (BMC) problem for Kripke structure K and safety property $\mathsf{G}p$ is encoded by

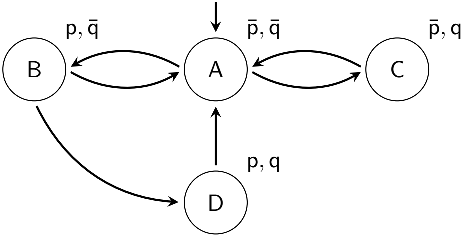
$$I(s_0) \wedge \mathcal{T}(s_0, s_1) \wedge \mathcal{T}(s_1, s_2) \wedge \ldots \wedge \mathcal{T}(s_{k-1}, s_k) \wedge B(s_k)$$

where

- $I(s_0)$ is true $\Leftrightarrow$ s_0 is an initial state
- $\mathcal{T}$ is the transition function of K
- $B(s_k)$ is true $\Leftrightarrow$ s_k is a bad state, i.e., $\neg p$ holds in s_k

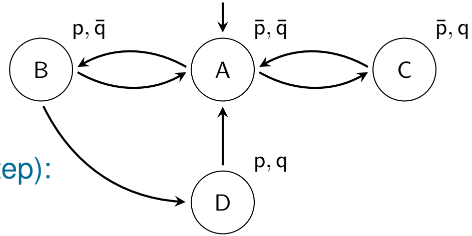
BMC Example

We want to know if $\mathsf{G}(\bar{\mathsf{p}} \vee \bar{\mathsf{q}})$ holds for Kripke structure K:



BMC Example

We want to know if $\mathsf{G}(\bar{p} \vee \bar{q})$ holds for Kripke structure K:



$k = 1$ (one step):

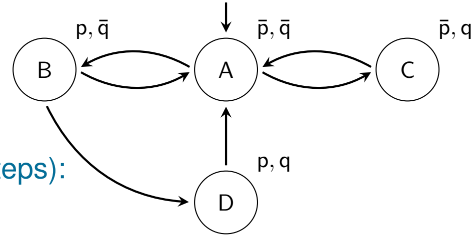
$(\bar{p} \wedge \bar{q}) \quad \wedge$

$((p' \leftrightarrow (\bar{p} \wedge \bar{q})) \wedge (q' \leftrightarrow 0)) \vee ((p' \leftrightarrow (p \wedge \bar{q})) \wedge (q' \leftrightarrow \bar{q})) \quad \wedge$

$(p' \wedge q')$

BMC Example

We want to know if $\mathsf{G}(\bar{\mathsf{p}} \vee \bar{\mathsf{q}})$ holds for Kripke structure K:



$k = 2$ (two steps):

$(\bar{p} \wedge \bar{q}) \quad \wedge$

$(((p' \leftrightarrow (\bar{p} \wedge \bar{q})) \wedge (q' \leftrightarrow 0)) \vee ((p' \leftrightarrow (p \wedge \bar{q})) \wedge (q' \leftrightarrow \bar{q})) \quad \wedge$

$(((p'' \leftrightarrow (\bar{p}' \wedge \bar{q}')) \wedge (q'' \leftrightarrow 0)) \vee$

$((p'' \leftrightarrow (p' \wedge \bar{q}')) \wedge (q'' \leftrightarrow \bar{q}'))) \quad \wedge$

$(p'' \wedge q'')$

Bounded Model Checking (Fairness)

- Given a Kripke structure K. Is there a path such that a property $\neg p$ holds forever?

- In other words: there is a path such that Fp does not hold in K

- Observation 1: if Fp does not hold in K, there is an **infinite counter-example**.
- Observation 2: if the counter-example is infinite, then it has to be because of a cycle.

Bounded Model Checking (Fairness)

A bounded model checking (BMC) problem for Kripke structure K and fairness property Fp is encoded by

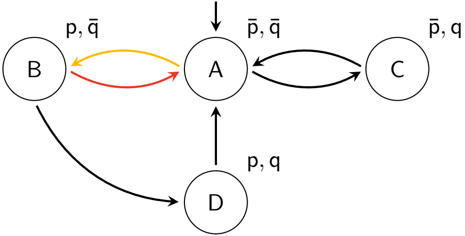
$$I(s_0) \wedge \bigwedge_{l=0}^{k-1} \mathcal{T}(s_l, s_{l+1}) \wedge \bigvee_{i=0}^{k} \mathcal{T}(s_k, s_i) \wedge \bigwedge_{j=0}^{k} F(s_j)$$

where

- $I(s_0)$ is true $\Leftrightarrow$ s_0 is an initial state
- $\mathcal{T}$ is the transition function of K
- $F(s_k)$ is true $\Leftrightarrow$ $\neg p$ holds in s_k

BMC Fairness

We want to know if $\mathsf{F}q$ holds for Kripke structure K:



Initial State:

$(\bar{p} \wedge \bar{q}) \quad \wedge$

One Step:

$(((p' \leftrightarrow (\bar{p} \wedge \bar{q})) \wedge (q' \leftrightarrow 0)) \vee ((p' \leftrightarrow (p \wedge \bar{q})) \wedge (q' \leftrightarrow \bar{q}))) \quad \wedge$

Cycle Check:

$((((p \leftrightarrow (\bar{p}' \wedge \bar{q}')) \wedge (q \leftrightarrow 0)) \vee ((p \leftrightarrow (p' \wedge \bar{q}')) \wedge (q \leftrightarrow \bar{q}'))) \quad \vee$

$(((p' \leftrightarrow (\bar{p}' \wedge \bar{q}')) \wedge (q' \leftrightarrow 0)) \vee ((p' \leftrightarrow (p' \wedge \bar{q}')) \wedge (q' \leftrightarrow \bar{q}')))) \quad \wedge$

Property Check:

$\bar{q} \wedge \bar{q}'$

BMC Summary

- BMC is incomplete ...
  - if all checked formulas are unsat, no insight
  - how to choose k? when to stop increasing k?

- ... very efficient (e.g., debugging)

- many tuning techniques
  - exploit similarities between two transition steps (structure sharing
  - simplification of formula by rewritings)

How to choose k for Safety?

Given Kripke structure K, the **diameter** is the smallest number d such that for every path $s_0, \ldots, s_{d+1}$ there exists a path $t_0, \ldots, t_l$ such that $l \leq d$ and $t_0 = s_0$ and $t_l = s_{d+1}$.

- If a state s is reachable from state t, then there is a path of length d or less where d is the diameter.
- The diameter is the longest shortest path.
- Computing the diameter is difficult (solve a QBF).