Core Course Preview

Introduction to Model Checking

https://clegra.github.io/mc/mc.html

Clemens Grabmayer
https://clegra.github.io

Emilio Tuosto
https://cs.gssi.it/emilio.tuosto/

Department of Computer Science

GRAN SASSO SCIENCE INSTITUTE
SCHOOL OF ADVANCED STUDIES
Scuola Universitaria Superiore

December 11, 2025

Model Checking

. . . is an effective automatable technique:

- *to expose potential software design errors;*
- *that, given a finite-state model of a system and a formal property, systematically checks whether this property holds for that model.*

Model Checking

. . . is an effective automatable technique:

- *to expose potential software design errors;*
- *that, given a finite-state model of a system and a formal property, systematically checks whether this property holds for that model.*

- widely applied in industry
  - embedded systems, software engineering, hardware design, symbolic AI
- supports partial verification (of system parts)
- provides diagnostic information for debugging
- has sound mathematical underpinning (logic and process theory)

Model Checking

. . . is an effective automatable technique:

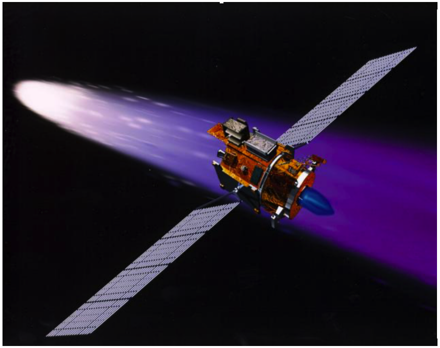
- *to expose potential software design errors;*
- *that, given a finite-state model of a system and a formal property, systematically checks whether this property holds for that model.*

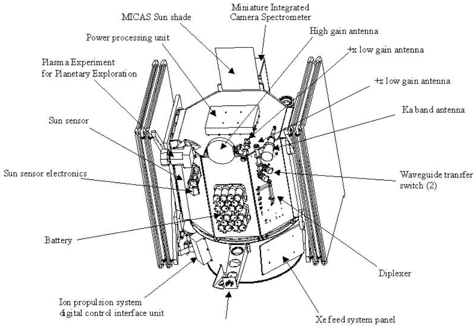
- widely applied in industry
    - embedded systems, software engineering, hardware design, symbolic AI
- supports partial verification (of system parts)
- provides diagnostic information for debugging
- has sound mathematical underpinning (logic and process theory)

Course Goals are introduction to:

- Theory:
    - modeling systems by labeled transition systems,
    - expressing properties by temporal-logic formulas
    - model-checking algorithms
- Practice: use the Maude system for examples

Deep Space 1 (NASA)





- Flyby of asteroid 9969 Braille (1999)
- Entered the coma of Comet Borrelly (2001)
- Model checking discovered 5 concurrency errors

Example (program concurrency/non-determinism)

Programs Inc, Dec, and Reset cooperate, and use a shared variable x:

```
proc Inc
  while true
    do
      if x < 200
        then x := x + 1
      fi
    od
```

```
proc Dec
  while true
    do
      if x > 0
        then x := x - 1
      fi
    od
```

```
proc Reset
  while true
    do
      if x = 200
        then x := 0
      fi
    od
```

Example (program concurrency/non-determinism)

Programs Inc, Dec, and Reset cooperate, and use a shared variable x:

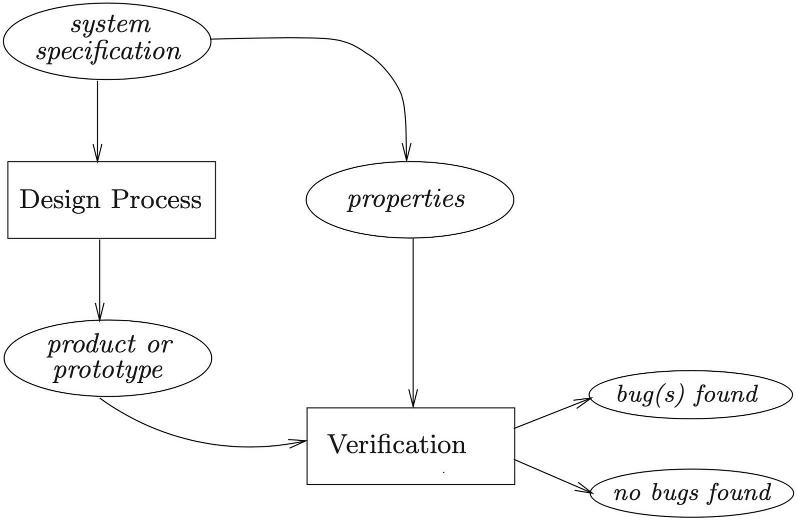
```
proc Inc
  while true
    do
      if x < 200
        then x := x + 1
      fi
    od
```

```
proc Dec
  while true
    do
      if x > 0
        then x := x - 1
      fi
    od
```

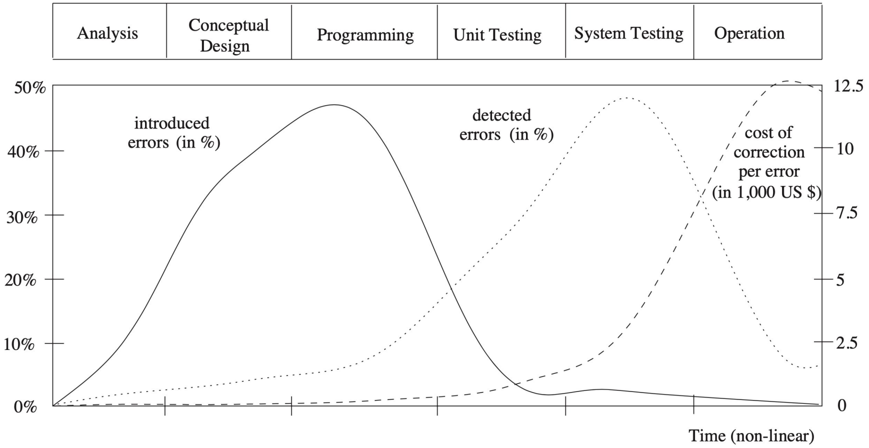
```
proc Reset
  while true
    do
      if x = 200
        then x := 0
      fi
    od
```

Question: Is $0 \leq x \leq 200$ always guaranteed?

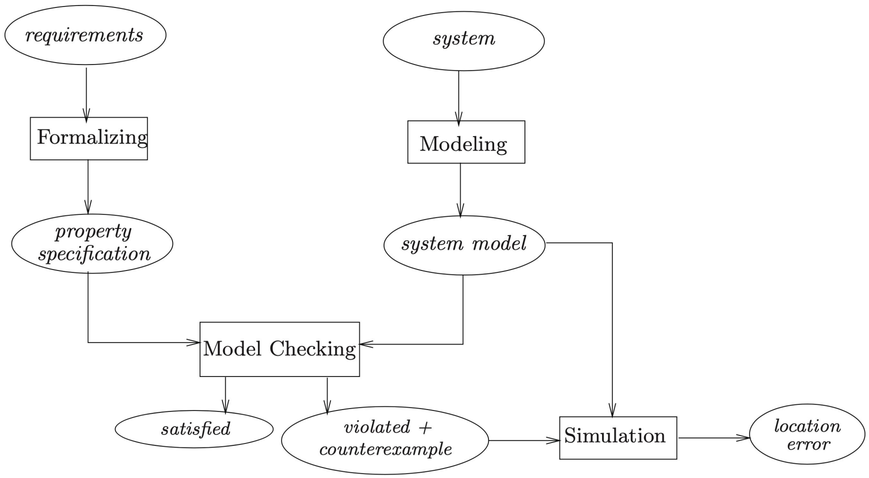
Hard-/Software Verification (traditionally)



Error introduction, detection, and repair costs



Model checking



Modeling (by program graphs)

```
proc Inc
  while true
    do
      if x < 200
        then x := x + 1
      fi
    od
```

```
proc Dec
  while true
    do
      if x > 0
        then x := x - 1
      fi
    od
```

```
proc Reset
  while true
    do
      if x = 200
        then x := 0
      fi
    od
```

Modeling (by program graphs)

```
proc Inc
  while true
    do
1:    if x < 200
2:      then x := x + 1
      fi
    od
```

```
proc Dec
  while true
    do
1:    if x > 0
2:      then x := x - 1
      fi
    od
```

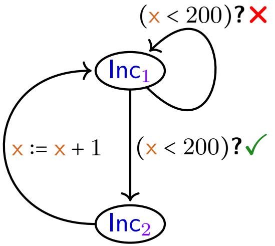
```
proc Reset
  while true
    do
1:    if x = 200
2:      then x := 0
      fi
    od
```

Modeling (by program graphs)

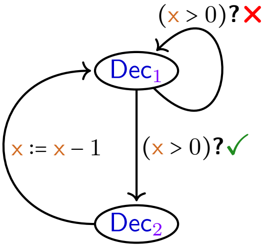
```
proc Inc
  while true
    do
1:    if x < 200
2:      then x := x + 1
      fi
    od
```

```
proc Dec
  while true
    do
1:    if x > 0
2:      then x := x - 1
      fi
    od
```

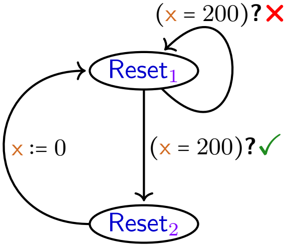
```
proc Reset
  while true
    do
1:    if x = 200
2:      then x := 0
      fi
    od
```



$$\begin{tikzcd} \mathrm{Inc}_1 \arrow[loop right, "(x<200)?\,✗"] \arrow[d, "(x<200)?\,✓"] \\ \mathrm{Inc}_2 \arrow[u, bend left=60, "x := x+1"] \end{tikzcd}$$



$$\begin{tikzcd} \mathrm{Dec}_1 \arrow[loop right, "(x>0)?\,✗"] \arrow[d, "(x>0)?\,✓"] \\ \mathrm{Dec}_2 \arrow[u, bend left=60, "x := x-1"] \end{tikzcd}$$



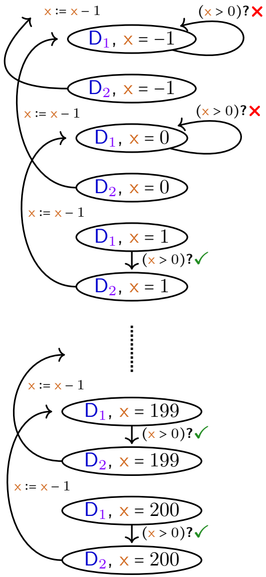
$$\begin{tikzcd} \mathrm{Reset}_1 \arrow[loop right, "(x=200)?\,✗"] \arrow[d, "(x=200)?\,✓"] \\ \mathrm{Reset}_2 \arrow[u, bend left=60, "x := 0"] \end{tikzcd}$$

Program graphs (PG)

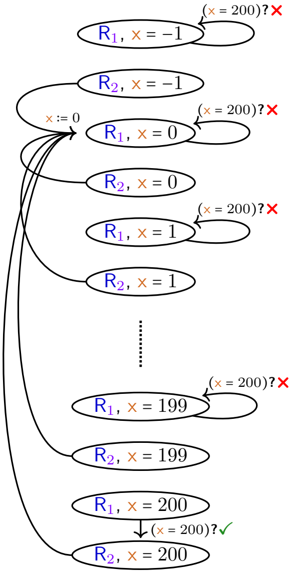
Modeling (by labeled transition systems, with state space explosion)

$$\begin{tikzcd}
{l_1,\ x = -1} \arrow[d, "(x < 200)?\checkmark"] \\
{l_2,\ x = -1} \arrow[d, "x := x + 1"] \\
{l_1,\ x = 0} \arrow[d, "(x < 200)?\checkmark"] \\
{l_2,\ x = 0} \arrow[d, "x := x + 1"] \\
{l_1,\ x = 1} \arrow[d, "(x < 200)?\checkmark"] \\
{l_2,\ x = 1} \arrow[d, "x := x + 1"] \\
\vdots \arrow[d, "x := x + 1"] \\
{l_1,\ x = 199} \arrow[d, "(x < 200)?\checkmark"] \\
{l_2,\ x = 199} \arrow[d, "x := x + 1"] \\
{l_1,\ x = 200} \arrow[loop right, "(x < 200)?\times"] \\
{l_2,\ x = 200} \arrow[d, "x := x + 1"] \\
{}
\end{tikzcd}$$

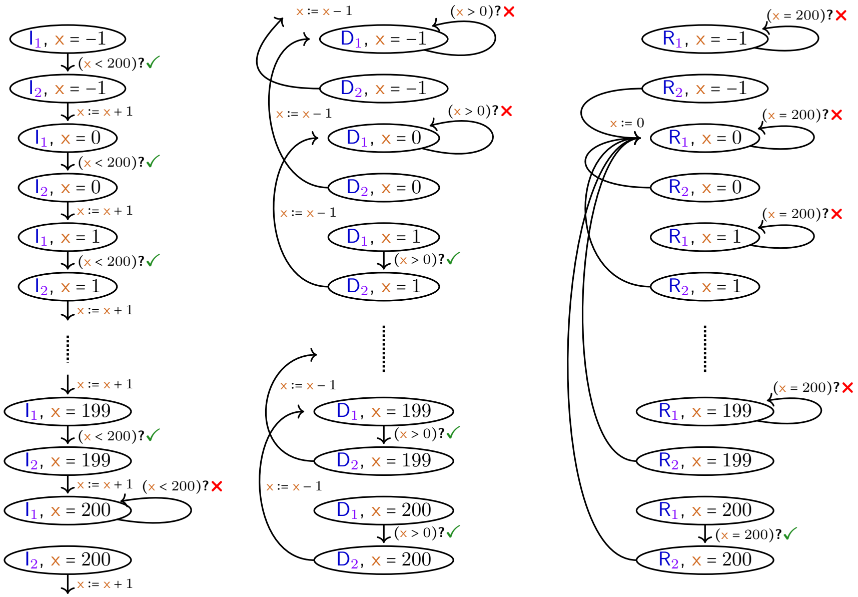
Modeling (by labeled transition systems, with state space explosion)



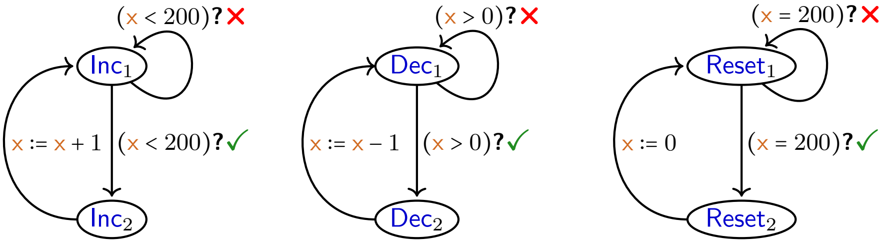
Modeling (by labeled transition systems, with state space explosion)



Modeling (by labeled transition systems, with state space explosion)



Formalizing properties (in temporal logic)



$\mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \overset{?}{\models} \Box(0 \leq x \wedge x \leq 200)$ (Linear-TL formula)

Counterexample (offending execution trace)

$$\boxed{\left\langle x = 199\,;\ \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle}$$

Counterexample (offending execution trace)

$$\boxed{\left\langle x = 199\,;\ \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle}$$

Counterexample (offending execution trace)

$$\begin{tikzcd}
\boxed{\left\langle x = 199\,;\ \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle} \arrow[d, "(x < 200)\textbf{?}\checkmark"] \\
\boxed{\left\langle x = 199\,;\ \mathsf{Inc}_2 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle}
\end{tikzcd}$$

Counterexample (offending execution trace)

$$\begin{tikzcd}
\boxed{\langle x = 199 \,;\, \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \rangle} \arrow[d, "(\mathsf{x} < 200)\textbf{?}\checkmark"] \\
\boxed{\langle x = 199 \,;\, \mathsf{Inc}_2 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \rangle}
\end{tikzcd}$$

Counterexample (offending execution trace)

$$\begin{tikzcd}
\boxed{\left\langle x = 199\,;\ \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle} \arrow[d, "(\mathsf{x} < 200)\textbf{?}\checkmark"] \\
\boxed{\left\langle x = 199\,;\ \mathsf{Inc}_2 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle} \arrow[d, "\mathsf{x} := \mathsf{x} + 1"] \\
\boxed{\left\langle x = 200\,;\ \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle}
\end{tikzcd}$$

Counterexample (offending execution trace)

$$\begin{tikzcd}
\boxed{\left\langle x = 199 \,;\, \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle} \arrow[d, "(\mathsf{x} < 200)\textbf{?}\checkmark"] \\
\boxed{\left\langle x = 199 \,;\, \mathsf{Inc}_2 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle} \arrow[d, "\mathsf{x} := \mathsf{x} + 1"] \\
\boxed{\left\langle x = 200 \,;\, \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle}
\end{tikzcd}$$

Counterexample (offending execution trace)

$$\begin{tikzcd}
\boxed{\langle x = 199 \,;\, \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \rangle} \arrow[d, "(\mathsf{x} < 200)\textbf{?}\checkmark"] \\
\boxed{\langle x = 199 \,;\, \mathsf{Inc}_2 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \rangle} \arrow[d, "\mathsf{x} := \mathsf{x} + 1"] \\
\boxed{\langle x = 200 \,;\, \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \rangle} \arrow[d, "(\mathsf{x} > 0)\textbf{?}\checkmark"] \\
\boxed{\langle x = 200 \,;\, \mathsf{Inc}_1 \parallel \mathsf{Dec}_2 \parallel \mathsf{Reset}_1 \rangle}
\end{tikzcd}$$

Counterexample (offending execution trace)

$$\begin{tikzcd}
\boxed{\left\langle x = 199 \,;\, \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle} \arrow[d, "(\mathsf{x} < 200)\textbf{?}\checkmark"] \\
\boxed{\left\langle x = 199 \,;\, \mathsf{Inc}_2 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle} \arrow[d, "\mathsf{x} := \mathsf{x} + 1"] \\
\boxed{\left\langle x = 200 \,;\, \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle} \arrow[d, "(\mathsf{x} > 0)\textbf{?}\checkmark"] \\
\boxed{\left\langle x = 200 \,;\, \mathsf{Inc}_1 \parallel \mathsf{Dec}_2 \parallel \mathsf{Reset}_1 \right\rangle}
\end{tikzcd}$$

Counterexample (offending execution trace)

$$\begin{tikzcd}
\boxed{\left\langle x = 199 \,;\, \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle} \arrow[d, "(\mathsf{x} < 200)\mathbf{?}\checkmark"] \\
\boxed{\left\langle x = 199 \,;\, \mathsf{Inc}_2 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle} \arrow[d, "\mathsf{x} := \mathsf{x} + 1"] \\
\boxed{\left\langle x = 200 \,;\, \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle} \arrow[d, "(\mathsf{x} > 0)\mathbf{?}\checkmark"] \\
\boxed{\left\langle x = 200 \,;\, \mathsf{Inc}_1 \parallel \mathsf{Dec}_2 \parallel \mathsf{Reset}_1 \right\rangle} \arrow[d, "(\mathsf{x} = 200)\mathbf{?}\checkmark"] \\
\boxed{\left\langle x = 200 \,;\, \mathsf{Inc}_1 \parallel \mathsf{Dec}_2 \parallel \mathsf{Reset}_2 \right\rangle}
\end{tikzcd}$$

Counterexample (offending execution trace)

$$\begin{tikzcd}
\boxed{\left\langle x = 199 \,;\, \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle} \arrow[d, "(\mathsf{x} < 200)\mathbf{?}\checkmark"] \\
\boxed{\left\langle x = 199 \,;\, \mathsf{Inc}_2 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle} \arrow[d, "\mathsf{x} := \mathsf{x} + 1"] \\
\boxed{\left\langle x = 200 \,;\, \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle} \arrow[d, "(\mathsf{x} > 0)\mathbf{?}\checkmark"] \\
\boxed{\left\langle x = 200 \,;\, \mathsf{Inc}_1 \parallel \mathsf{Dec}_2 \parallel \mathsf{Reset}_1 \right\rangle} \arrow[d, "(\mathsf{x} = 200)\mathbf{?}\checkmark"] \\
\boxed{\left\langle x = 200 \,;\, \mathsf{Inc}_1 \parallel \mathsf{Dec}_2 \parallel \mathsf{Reset}_2 \right\rangle}
\end{tikzcd}$$

Counterexample (offending execution trace)

$$\begin{tikzcd}
\boxed{\left\langle x = 199 \,;\, \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle} \arrow[d, "(\mathsf{x} < 200)\mathbf{?}\checkmark"] \\
\boxed{\left\langle x = 199 \,;\, \mathsf{Inc}_2 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle} \arrow[d, "\mathsf{x} := \mathsf{x} + 1"] \\
\boxed{\left\langle x = 200 \,;\, \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle} \arrow[d, "(\mathsf{x} > 0)\mathbf{?}\checkmark"] \\
\boxed{\left\langle x = 200 \,;\, \mathsf{Inc}_1 \parallel \mathsf{Dec}_2 \parallel \mathsf{Reset}_1 \right\rangle} \arrow[d, "(\mathsf{x} = 200)\mathbf{?}\checkmark"] \\
\boxed{\left\langle x = 200 \,;\, \mathsf{Inc}_1 \parallel \mathsf{Dec}_2 \parallel \mathsf{Reset}_2 \right\rangle} \arrow[d, "\mathsf{x} := 0"] \\
\boxed{\left\langle x = 0 \,;\, \mathsf{Inc}_1 \parallel \mathsf{Dec}_2 \parallel \mathsf{Reset}_1 \right\rangle}
\end{tikzcd}$$

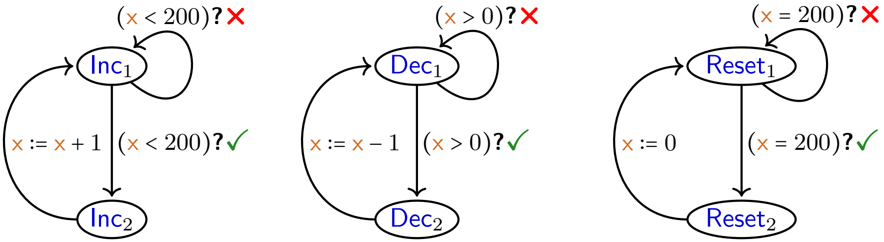
Counterexample (offending execution trace)

$$\begin{tikzcd}
\boxed{\left\langle x = 199 \,;\, \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle} \arrow[d, "(\mathsf{x} < 200)\textbf{?}\checkmark"] \\
\boxed{\left\langle x = 199 \,;\, \mathsf{Inc}_2 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle} \arrow[d, "\mathsf{x} := \mathsf{x} + 1"] \\
\boxed{\left\langle x = 200 \,;\, \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle} \arrow[d, "(\mathsf{x} > 0)\textbf{?}\checkmark"] \\
\boxed{\left\langle x = 200 \,;\, \mathsf{Inc}_1 \parallel \mathsf{Dec}_2 \parallel \mathsf{Reset}_1 \right\rangle} \arrow[d, "(\mathsf{x} = 200)\textbf{?}\checkmark"] \\
\boxed{\left\langle x = 200 \,;\, \mathsf{Inc}_1 \parallel \mathsf{Dec}_2 \parallel \mathsf{Reset}_2 \right\rangle} \arrow[d, "\mathsf{x} := 0"] \\
\boxed{\left\langle x = 0 \,;\, \mathsf{Inc}_1 \parallel \mathsf{Dec}_2 \parallel \mathsf{Reset}_1 \right\rangle}
\end{tikzcd}$$

Counterexample (offending execution trace)

$$\begin{tikzcd}
\boxed{\left\langle x = 199\ ;\ \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle} \arrow[d, "(\mathsf{x} < 200)\textbf{?}\checkmark"] \\
\boxed{\left\langle x = 199\ ;\ \mathsf{Inc}_2 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle} \arrow[d, "\mathsf{x} := \mathsf{x} + 1"] \\
\boxed{\left\langle x = 200\ ;\ \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle} \arrow[d, "(\mathsf{x} > 0)\textbf{?}\checkmark"] \\
\boxed{\left\langle x = 200\ ;\ \mathsf{Inc}_1 \parallel \mathsf{Dec}_2 \parallel \mathsf{Reset}_1 \right\rangle} \arrow[d, "(\mathsf{x} = 200)\textbf{?}\checkmark"] \\
\boxed{\left\langle x = 200\ ;\ \mathsf{Inc}_1 \parallel \mathsf{Dec}_2 \parallel \mathsf{Reset}_2 \right\rangle} \arrow[d, "\mathsf{x} := 0"] \\
\boxed{\left\langle x = 0\ ;\ \mathsf{Inc}_1 \parallel \mathsf{Dec}_2 \parallel \mathsf{Reset}_1 \right\rangle} \arrow[d, "\mathsf{x} := \mathsf{x} - 1"] \\
\boxed{\left\langle x = -1\ ;\ \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle}
\end{tikzcd}$$

Formalizing properties (in temporal logic)



$\mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \not\models \Box(0 \le x \wedge x \le 200)$ (Linear-TL formula)

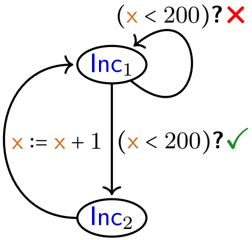
Formalizing properties (in temporal logic)

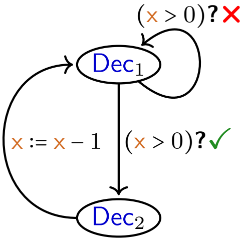
$$\begin{tikzcd}
\mathsf{Inc}_1 \arrow[loop right, "(x < 200)?\,\times"] \arrow[d, "(x < 200)?\,\checkmark"] \\
\mathsf{Inc}_2 \arrow[u, bend left=60, "x := x + 1"]
\end{tikzcd}
\qquad
\begin{tikzcd}
\mathsf{Dec}_1 \arrow[loop right, "(x > 0)?\,\times"] \arrow[d, "(x > 0)?\,\checkmark"] \\
\mathsf{Dec}_2 \arrow[u, bend left=60, "x := x - 1"]
\end{tikzcd}
\qquad
\begin{tikzcd}
\mathsf{Reset}_1 \arrow[loop right, "(x = 200)?\,\times"] \arrow[d, "(x = 200)?\,\checkmark"] \\
\mathsf{Reset}_2 \arrow[u, bend left=60, "x := 0"]
\end{tikzcd}$$

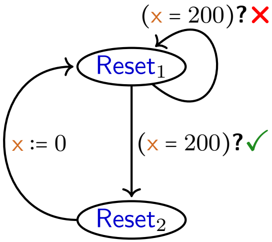
$\mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \not\models \Box(0 \leq x \wedge x \leq 200)$ (Linear-TL formula)

$\mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \models \Diamond(x < 0)$ (LTL formula)

Formalizing properties (in temporal logic)

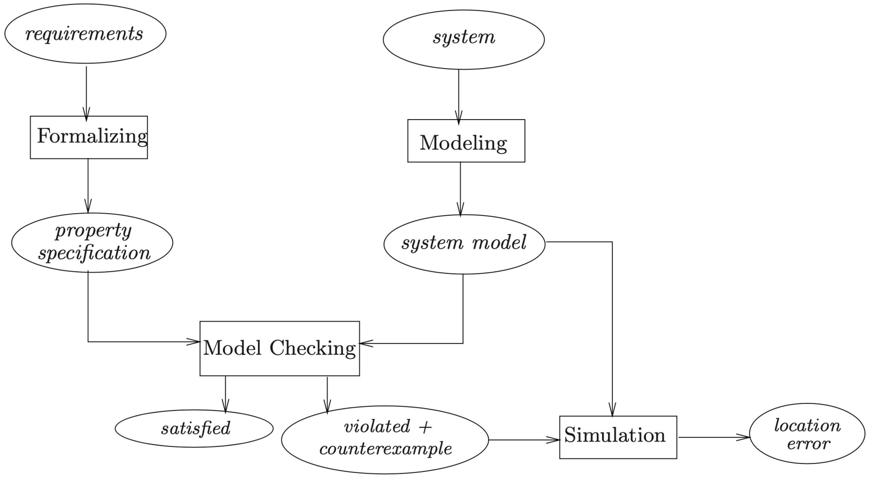






$\mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1$	$\not\models$	$\Box(0 \leq x \,\wedge\, x \leq 200)$	(Linear-TL formula)
$\mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1$	$\models$	$\Diamond(x < 0)$	(LTL formula)
$\mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1$	$\not\models$	$\forall\Box(0 \leq x \,\wedge\, x \leq 200)$	(Computation-Tree-L formula)
$\mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1$	$\models$	$\exists\Box(0 \leq x \,\wedge\, x \leq 200)$	(CTL formula)
$\mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1$	$\models$	$\forall\Box\exists\Diamond(x < 0)$	(CTL formula)

Model checking



Any [such] verification is only as good as the model of the system.

Possible Maude code (idea)

```
crl [Inc1a]   :  Inc1 x => Inc2 x  if x < 200
rl  [Inc2]    :  Inc2 x => Inc1 x + 1
crl [Inc1b]   :  Inc1 x => Inc1 x  if not(x < 200)
```

Possible Maude code (idea)

```
crl [Inc1a]   :  Inc1 x => Inc2 x  if x < 200
rl  [Inc2]    :  Inc2 x => Inc1 x + 1
crl [Inc1b]   :  Inc1 x => Inc1 x  if not(x < 200)

crl [Dec1a]   :  Dec1 x => Dec2 x  if x =/= 0
rl  [Dec2]    :  Dec2 x => Dec1 x - 1
crl [Dec1b]   :  Dec1 x => Dec1 x  if x = 0
```

Possible Maude code (idea)

```
crl [Inc1a]   :  Inc1 x => Inc2 x  if x < 200
rl  [Inc2]    :  Inc2 x => Inc1 x + 1
crl [Inc1b]   :  Inc1 x => Inc1 x  if not(x < 200)

crl [Dec1a]   :  Dec1 x => Dec2 x  if x =/= 0
rl  [Dec2]    :  Dec2 x => Dec1 x - 1
crl [Dec1b]   :  Dec1 x => Dec1 x  if x = 0

crl [Reset1a] :  Reset1 x => Reset2 x  if x = 200
rl  [Reset2]  :  Reset2 x => Reset1 0
crl [Reset1b] :  Reset1 x => Reset1 x  if not(x = 200)
```

Possible Maude code (idea)

```
crl [Inc1a]   :  Inc1 x => Inc2 x  if x < 200
rl  [Inc2]    :  Inc2 x => Inc1 x + 1
crl [Inc1b]   :  Inc1 x => Inc1 x  if not(x < 200)

crl [Dec1a]   :  Dec1 x => Dec2 x  if x =/= 0
rl  [Dec2]    :  Dec2 x => Dec1 x - 1
crl [Dec1b]   :  Dec1 x => Dec1 x  if x = 0

crl [Reset1a] :  Reset1 x => Reset2 x  if x = 200
rl  [Reset2]  :  Reset2 x => Reset1 0
crl [Reset1b] :  Reset1 x => Reset1 x  if not(x = 200)


eq initial = { Dec1 Inc1 Reset1 199 }
```

Possible Maude code (idea)

```
crl [Inc1a]   :  Inc1 x => Inc2 x  if x < 200
rl  [Inc2]    :  Inc2 x => Inc1 x + 1
crl [Inc1b]   :  Inc1 x => Inc1 x  if not(x < 200)

crl [Dec1a]   :  Dec1 x => Dec2 x  if x =/= 0
rl  [Dec2]    :  Dec2 x => Dec1 x - 1
crl [Dec1b]   :  Dec1 x => Dec1 x  if x = 0

crl [Reset1a] :  Reset1 x => Reset2 x  if x = 200
rl  [Reset2]  :  Reset2 x => Reset1 0
crl [Reset1b] :  Reset1 x => Reset1 x  if not(x = 200)

eq initial = { Dec1 Inc1 Reset1 199 }

ceq (S1 S2 S3 x |= counterge0) = true  if (0 < x \/ x = 0)
ceq (S1 S2 S3 x |= counterlt0) = true  if (x < 0)
ceq (S1 S2 S3 x |= counterle200) = true  if (x < 200 \/ x = 200)
```

Maude output

```
Maude> red modelCheck(initial, <> counterlt0)
reduce in COUNTERS-CHECK : modelCheck(initial, <> counterlt0)
result ModelCheckResult:
result Bool :  true
```

Maude output

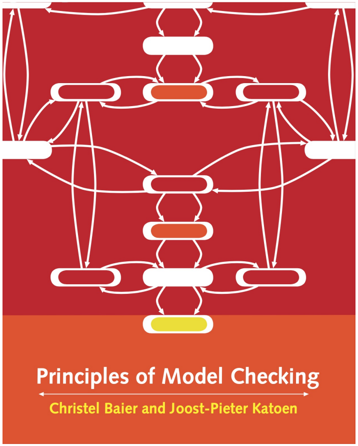
```
Maude> red modelCheck(initial, <> counterlt0)
reduce in COUNTERS-CHECK : modelCheck(initial, <> counterlt0)
result ModelCheckResult:
result Bool :  true


Maude> red modelCheck(initial, [](counterge0 /\counterle200)
reduce in COUNTERS-CHECK :
                modelCheck(initial, [](counterge0 /\counterle200)
result ModelCheckResult:
counterexample({Inc1 Dec1 Reset1 199}
                {Inc2 Dec1 Reset1 199}
                {Inc1 Dec1 Reset1 200}
                {Inc1 Dec2 Reset1 200}
                {Inc1 Dec2 Reset2 200}
                {Inc1 Dec2 Reset1 0}
                {Inc1 Dec1 Reset1 -1})
```

Topics of the course

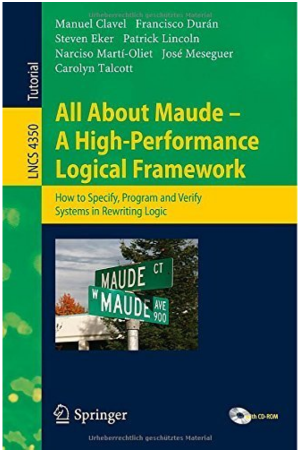
- modeling systems by labeled transition systems (LTSs)
- safety, liveness, and fairness properties
- Linear Temporal Logic (LTL)
  - model checking formulas
    - express properties by Büchi automata
    - model check LTSs and properties via product automata
- Computation Tree Logic (CTL)
- partial model checking
  - partially known systems (state properties/states/transitions)
- learning Maude and its model-checker

Book


Principles of Model Checking
Christel Baier and Joost-Pieter Katoen

- pdf available:
  https://is.ifmo.ru/books/_principles_of_model_checking.pdf

Book Maude


Manuel Clavel Francisco Durán
Steven Eker Patrick Lincoln
Narciso Martí-Oliet José Meseguer
Carolyn Talcott
Tutorial
LNCS 4350
All About Maude – A High-Performance Logical Framework
How to Specify, Program and Verify Systems in Rewriting Logic
MAUDE CT
W MAUDE AVE 900
Springer
with CD-ROM

- pdf available:
  https://maude.cs.illinois.edu/w/images/0/0d/Maude-book.pdf

Organization

Lectures (Emilio 3 / Clemens 4)

- January 19 – January 28 (7 meetings)
- blackboard presentations
- notes after the lecture (last year's notes available)
- Maude examples at the end of each lecture

Webpage

- https://clegra.github.io/mc/mc.html

Organization

Lectures (Emilio 3 / Clemens 4)

- January 19 – January 28 (7 meetings)
- blackboard presentations
- notes after the lecture (last year's notes available)
- Maude examples at the end of each lecture

Webpage

- https://clegra.github.io/mc/mc.html

Exam

- options:
  - verification project (of an algorithm, etc.) in Maude
  - presentation about a paper
  - written exam

Organization

Lectures (Emilio 3 / Clemens 4)

- January 19 – January 28 (7 meetings)
- blackboard presentations
- notes after the lecture (last year's notes available)
- Maude examples at the end of each lecture

Webpage

- https://clegra.github.io/mc/mc.html

Exam

- options:
  - verification project (of an algorithm, etc.) in Maude
  - presentation about a paper
  - written exam

Thank you – we are looking forward to the course!