Introduction to Model Checking

(Preview of Core Module)

Clemens Grabmayer

Emilio Tuosto

https://clegra.github.io

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

Department of Computer Science



GRAN SASSO SCIENCE INSTITUTE



SCHOOL OF ADVANCED STUDIES Scuola Universitaria Superiore

December 2, 2024

otivation mc-schema example counterex-trace logical properties topics book module organizatio

Model Checking

... is an effective automatable technique:

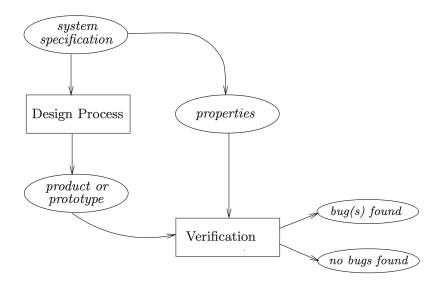
- to expose potential software design errors;
- that, given a finite-state model of a system and a formal property, systematically checks whether this property holds for that model.

Strengths:

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

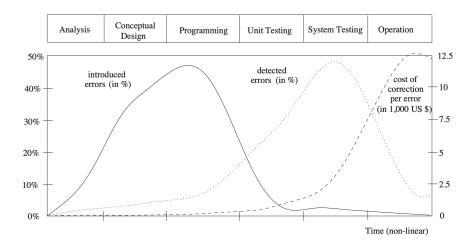
tivation mc-schema example counterex-trace logical properties topics book module organization

Hard-/Software Verification (traditionally)



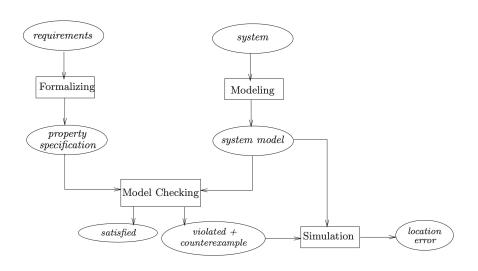
otivation mc-schema example counterex-trace logical properties topics book module organizatio

Error introduction, detection, and repair costs



ptivation mc-schema example counterex-trace logical properties topics book module organizati

Model checking



Example (program concurrency/non-determinism)

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

```
proc Inc
                            proc Dec
                                                        proc Reset
 while true
                              while true
                                                         while true
   do
                                ob
                                                           ob
    if \times < 200
                                 if x > 0
                                                            if x = 200
      then x := x + 1
                                 then \times := \times - 1
                                                             then x := 0
    fi
                                 fi
                                                             fi
   od
                                od
                                                           od
```

Example (program concurrency/non-determinism)

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

```
proc Inc
                             proc Dec
                                                          proc Reset
 while true
                               while true
                                                            while true
   do
                                 dΩ
                                                              ob
     if \times < 200
                                   if x > 0
                                                               if x = 200
       then x := x + 1
                                   then \times := \times - 1
                                                                then \mathbf{x} := 0
     fi
                                   fi
                                                                fi
   od
                                 od
                                                              od
```

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

Modeling (by labeled transition systems)

```
\begin{array}{c} \textbf{proc Inc} \\ \textbf{while true} \\ \textbf{do} \\ \textbf{if } \times < 200 \\ \textbf{then } \times := \times + 1 \\ \textbf{fi} \\ \textbf{od} \end{array}
```

```
\begin{array}{c} \textbf{proc Dec} \\ \textbf{while true} \\ \textbf{do} \\ \textbf{if } \times > 0 \\ \textbf{then } \times := \times -1 \\ \textbf{fi} \\ \textbf{od} \end{array}
```

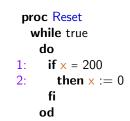
```
proc Reset
while true
do
if \times = 200
then \times := 0
fi
od
```

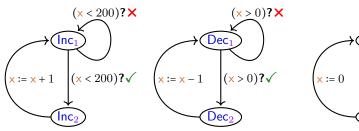
Modeling (by labeled transition systems)

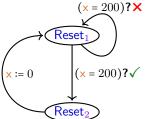
```
proc Inc
                       proc Dec
                                              proc Reset
 while true
                        while true
                                               while true
                          do
                                                do
  do
  if \times < 200 1: if \times > 0
                                          1: if \times = 200
 then x := x + 1 2: then x := x - 1
                                             2: then x := 0
    fi
                           fi
                                                  fi
  od
                          od
                                                od
```

Modeling (by labeled transition systems)

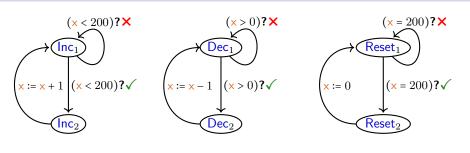
```
proc Dec
    while true
    do
1:         if x > 0
2:         then x := x - 1
         fi
         od
```







Labeled transition systems (LTSs)



$$Inc_1 \parallel Dec_1 \parallel Reset_1 \stackrel{?}{\vDash} \square (0 \le x \land x \le 200)$$
 (Linear-TL formula)

otivation mc-schema example <mark>counterex-trace</mark> logical properties topics book module organization

Counterexample (offending execution trace)

 $(x = 199; Inc_1 \parallel Dec_1 \parallel Reset_1)$

otivation mc-schema example <mark>counterex-trace</mark> logical properties topics book module organization

Counterexample (offending execution trace)

 $(x = 199; | lnc_1 | | lnc_1 | | Reset_1)$

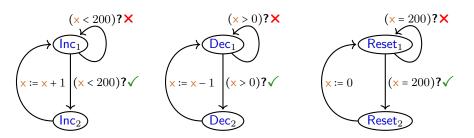
$$\begin{array}{c|c} \left(\left\langle x = 199 \; ; \; \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle \right) \\ & & \downarrow (\mathsf{x} < 200) ? \checkmark \\ \\ \left(\left\langle x = 199 \; ; \; \mathsf{Inc}_2 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle \right) \end{array}$$

$$\begin{array}{c|c} \left(\left\langle x = 199 \; ; \; \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle \right) \\ & & \downarrow (\mathsf{x} < 200) ? \checkmark \\ \hline \left(\left\langle x = 199 \; ; \; \mathsf{Inc}_2 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \right\rangle \right) \end{array}$$

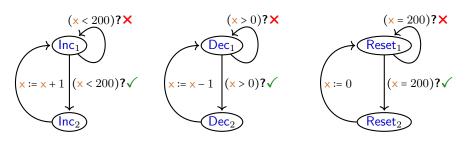
$$\begin{array}{c|c} \left(x = 199 \; ; \; \mathsf{Inc_1} \parallel \mathsf{Dec_1} \parallel \mathsf{Reset_1} \right) \\ & & & & & & & \\ & & & & & & \\ & & & & & & \\ \hline \left(x = 199 \; ; \; \mathsf{Inc_2} \parallel \mathsf{Dec_1} \parallel \mathsf{Reset_1} \right) \\ & & & & & & \\ \hline \left(x = 200 \; ; \; \mathsf{Inc_1} \parallel \mathsf{Dec_1} \parallel \mathsf{Reset_1} \right) \\ & & & & & \\ \hline \left(x = 200 \; ; \; \mathsf{Inc_1} \parallel \mathsf{Dec_2} \parallel \mathsf{Reset_1} \right) \\ \hline \end{array}$$

$$\begin{array}{c} \left(x = 199 \; ; \; \operatorname{Inc_1} \parallel \operatorname{Dec_1} \parallel \operatorname{Reset_1} \right) \\ & \downarrow (\mathsf{x} < 200) ? \checkmark \\ \hline \\ \left(x = 199 \; ; \; \operatorname{Inc_2} \parallel \operatorname{Dec_1} \parallel \operatorname{Reset_1} \right) \\ & \downarrow \mathsf{x} := \mathsf{x} + 1 \\ \hline \\ \left(x = 200 \; ; \; \operatorname{Inc_1} \parallel \operatorname{Dec_1} \parallel \operatorname{Reset_1} \right) \\ & \downarrow (\mathsf{x} > 0) ? \checkmark \\ \hline \\ \left(x = 200 \; ; \; \operatorname{Inc_1} \parallel \operatorname{Dec_2} \parallel \operatorname{Reset_1} \right) \\ & \downarrow (\mathsf{x} = 200) ? \checkmark \\ \hline \\ \left(x = 200 \; ; \; \operatorname{Inc_1} \parallel \operatorname{Dec_2} \parallel \operatorname{Reset_2} \right) \\ & \downarrow \mathsf{x} := 0 \\ \hline \\ \left(x = 0 \; ; \; \operatorname{Inc_1} \parallel \operatorname{Dec_2} \parallel \operatorname{Reset_1} \right) \\ \hline \end{array}$$

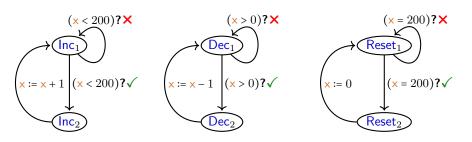
$$\begin{array}{c} \left(x = 199 \; ; \; \operatorname{Inc}_1 \parallel \operatorname{Dec}_1 \parallel \operatorname{Reset}_1 \right) \\ & \downarrow (\mathsf{x} < 200) ? \checkmark \\ \hline \\ \left(x = 199 \; ; \; \operatorname{Inc}_2 \parallel \operatorname{Dec}_1 \parallel \operatorname{Reset}_1 \right) \\ & \downarrow \mathsf{x} := \mathsf{x} + 1 \\ \hline \\ \left(x = 200 \; ; \; \operatorname{Inc}_1 \parallel \operatorname{Dec}_1 \parallel \operatorname{Reset}_1 \right) \\ & \downarrow (\mathsf{x} > 0) ? \checkmark \\ \hline \\ \left(x = 200 \; ; \; \operatorname{Inc}_1 \parallel \operatorname{Dec}_2 \parallel \operatorname{Reset}_1 \right) \\ & \downarrow \mathsf{x} := 0 \\ \hline \\ \left(x = 0 \; ; \; \operatorname{Inc}_1 \parallel \operatorname{Dec}_2 \parallel \operatorname{Reset}_1 \right) \\ & \downarrow \mathsf{x} := \mathsf{x} - 1 \\ \hline \\ \left(x = -1 \; ; \; \operatorname{Inc}_1 \parallel \operatorname{Dec}_1 \parallel \operatorname{Reset}_1 \right) \\ \hline \end{array}$$



$$Inc_1 \parallel Dec_1 \parallel Reset_1 \neq \Box (0 \le x \land x \le 200)$$
 (Linear-TL formula)



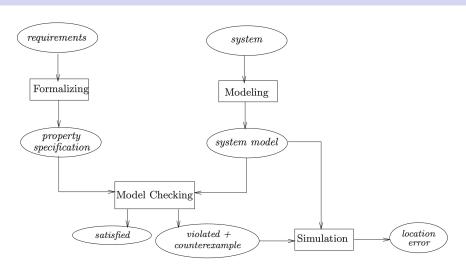
$$\begin{aligned} & \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \quad \not\models \quad \Box \big(0 \leq x \, \land \, x \leq 200 \big) & \quad \big(\mathsf{Linear}\text{-}\mathsf{TL} \; \mathsf{formula} \big) \\ & \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \; \models \; \diamondsuit (x < 0) & \quad \big(\mathsf{LTL} \; \mathsf{formula} \big) \end{aligned}$$



```
\begin{split} & | \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \; \not\models \; \Box (0 \leq x \, \land \, x \leq 200) \qquad \text{(Linear-TL formula)} \\ & | \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \; \models \; \diamondsuit (x < 0) \qquad \qquad \text{(LTL formula)} \\ & | \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \; \not\models \; \forall \Box (0 \leq x \, \land \, x \leq 200) \qquad \text{(Computation-Tree-L formula)} \\ & | \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \; \models \; \exists \Box (0 \leq x \, \land \, x \leq 200) \qquad \text{(CTL formula)} \\ & | \mathsf{Inc}_1 \parallel \mathsf{Dec}_1 \parallel \mathsf{Reset}_1 \; \models \; \forall \Box \exists \diamondsuit (x < 0) \qquad \qquad \text{(CTL formula)} \end{split}
```

otivation mc-schema example counterex-trace logical properties topics book module organization

Model checking



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

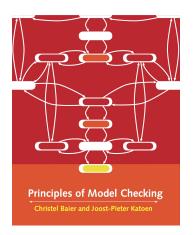
tivation mc-schema example counterex-trace logical properties <mark>topics</mark> book module organizatio

Topics of the module

- modeling systems by labeled transition systems (LTSs)
- fairness
- 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)
- ▶ analysing system behavior with the mCRL2 model-checker toolbox

tivation mc-schema example counterex-trace logical properties topics <mark>book</mark> module organization

Book



pdf available:

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

tivation mc-schema example counterex-trace logical properties topics book <mark>module organizati</mark>c

Organization

Lectures (Clemens 5/Emilio 2)

- presentations on blackboard
- notes after the lecture (notes 2024/25 available)
- February (first/second week)

tivation mc-schema example counterex-trace logical properties topics book module organization

Organization

Lectures (Clemens 5/Emilio 2)

- presentations on blackboard
- notes after the lecture (notes 2024/25 available)
- February (first/second week)

Exam

- options:
 - ▶ small verification project (of an algorithm, e.g. in mCRL2)
 - presentation about a paper
 - written exam?

otivation mc-schema example counterex-trace logical properties topics book <mark>module organizat</mark>i

Organization

Lectures (Clemens 5/Emilio 2)

- presentations on blackboard
- notes after the lecture (notes 2024/25 available)
- February (first/second week)

Exam

- options:
 - ▶ small verification project (of an algorithm, e.g. in mCRL2)
 - presentation about a paper
 - written exam?

Thank you – we are looking forward to the course!