Lecture 1: Introduction to model checking

B. Srivathsan

Chennai Mathematical Institute

Model Checking and Systems Verification

January - April 2015

What are we **interested** in?

What are we **interested** in?

Software Controllers

Code that controls the working of

safety critical systems

Safety-critical systems

Controlled by software

- Aircrafts
- Medical devices
- Cars
- Nuclear power plants
- Space missions
- Railway signalling systems
- *and many more ...*

Controlled by software

- Aircrafts
- Medical devices
- Cars
- Nuclear power plants
- Space missions
- Railway signalling systems
- *and many more ...*

Correctness of these software is very important

Accidents due to software bugs

- *Igor Walukiewicz's slides (4 - 7)*
- *Yogananda Jeppu's slides (22 - 38)*

Errors that are hard to detect

Concurrent programs

```
while x < 200
    x := x+1
```

```
while x>0
    x := x-1
```

```
while x=200
    x := 0
```

Concurrent programs

```
while x < 200
    x := x+1
```

```
while x>0
    x := x-1
```

```
while x=200
    x := 0
```

Is the value of x **always between** 0 and 200?

Concurrent programs

```
while x < 200
    x := x+1
```

```
while x>0
    x := x-1
```

```
while x=200
    x := 0
```

Is the value of x **always between** 0 and 200? **No!** (why?)

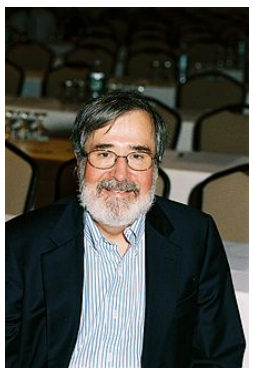
Goal: Make **low-defect** software controllers

Traditional testing **insufficient** for safety-critical systems

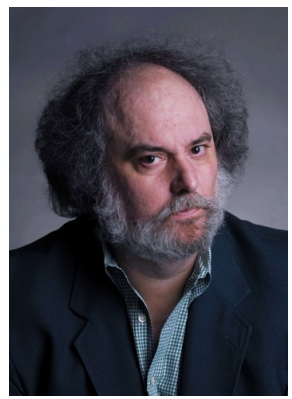
Goal: Make **low-defect** software controllers

Traditional testing **insufficient** for safety-critical systems

→ A new **verification technology** called **Model-checking**



Edmund Clarke

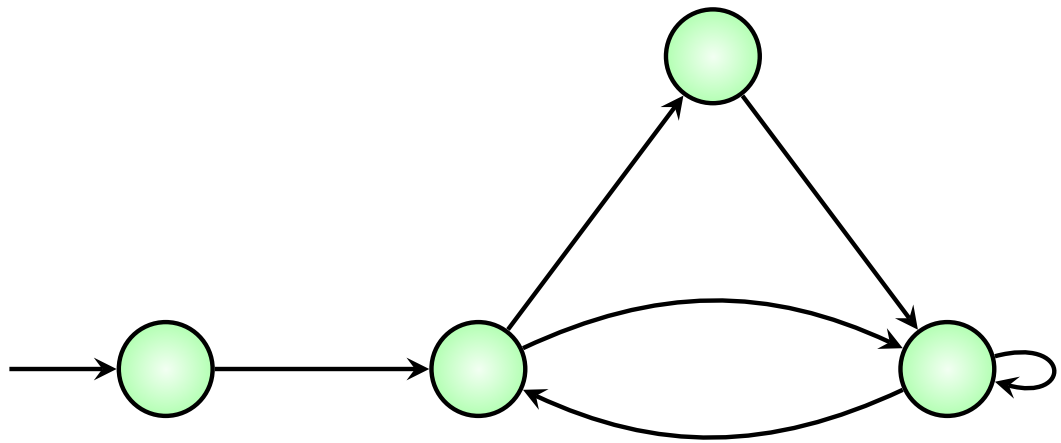


Allen Emerson

Joseph Sifakis

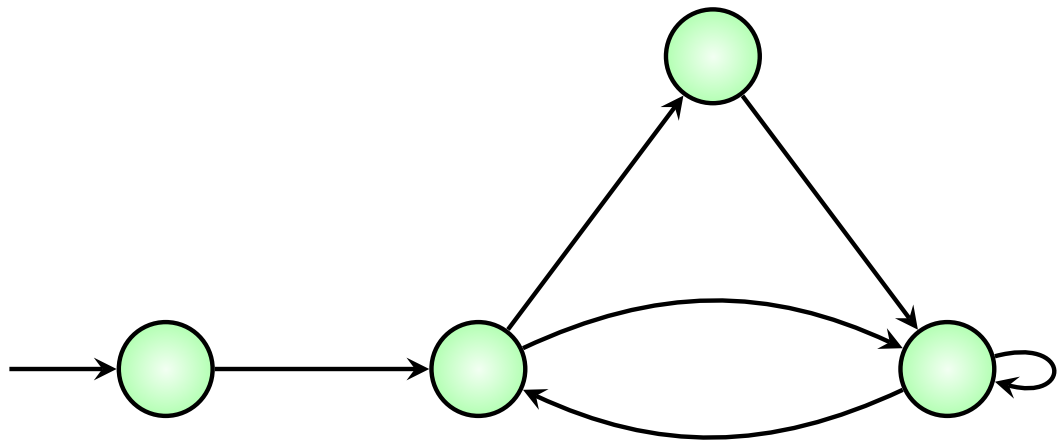
Model Checking

Think of controllers as **finite state machines**



Model Checking

Think of controllers as **finite state machines**



Philosophy: Computations as sequences of states - Igor's slides (55 - 57)

```
while x < 200
    x := x+1
```

```
while x>0
    x := x-1
```

```
while x=200
    x := 0
```

```
while x < 200
    x := x+1
```

```
while x>0
    x := x-1
```

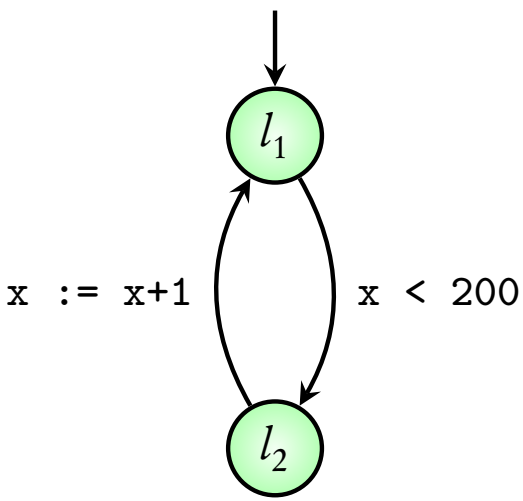
```
while x=200
    x := 0
```

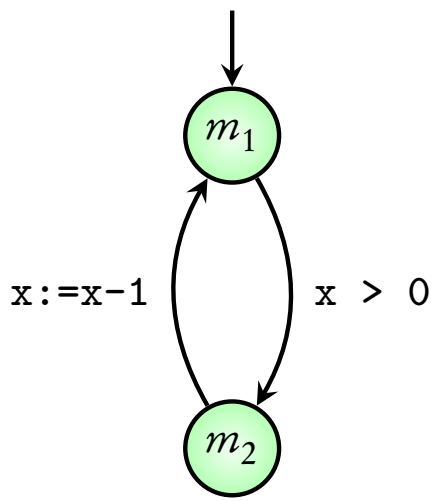
Is the value of x **always between** 0 and 200?

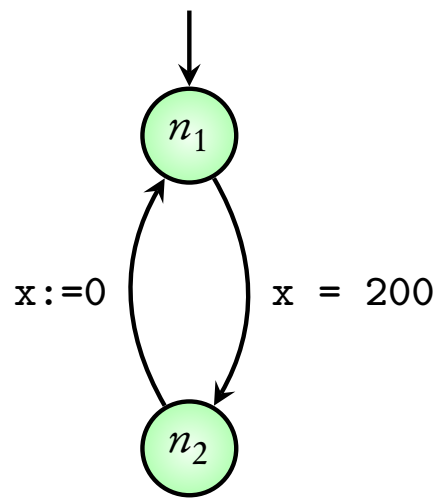
```
while x < 200
    x := x+1
```

```
while x>0
    x := x-1
```

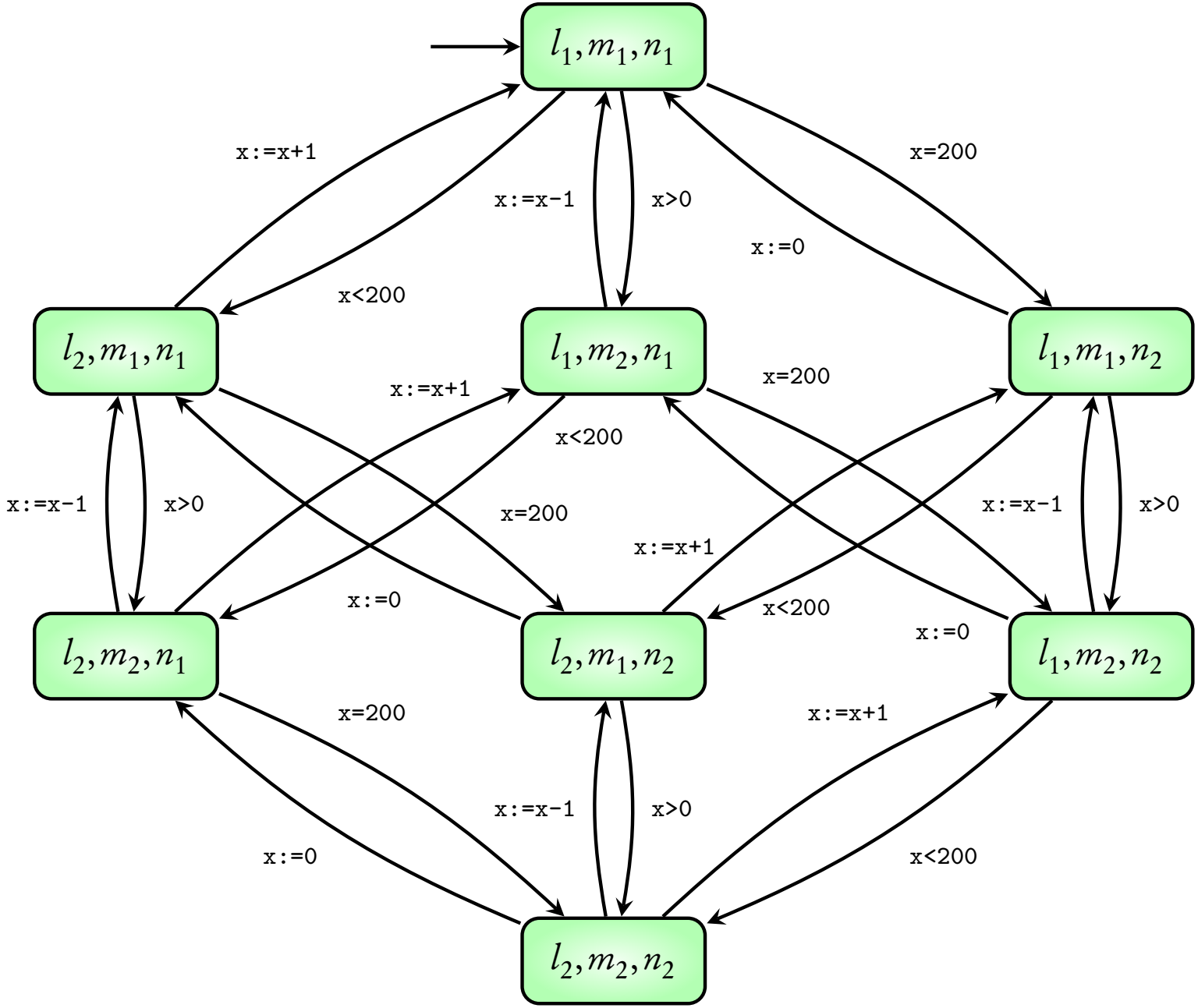
```
while x=200
    x := 0
```

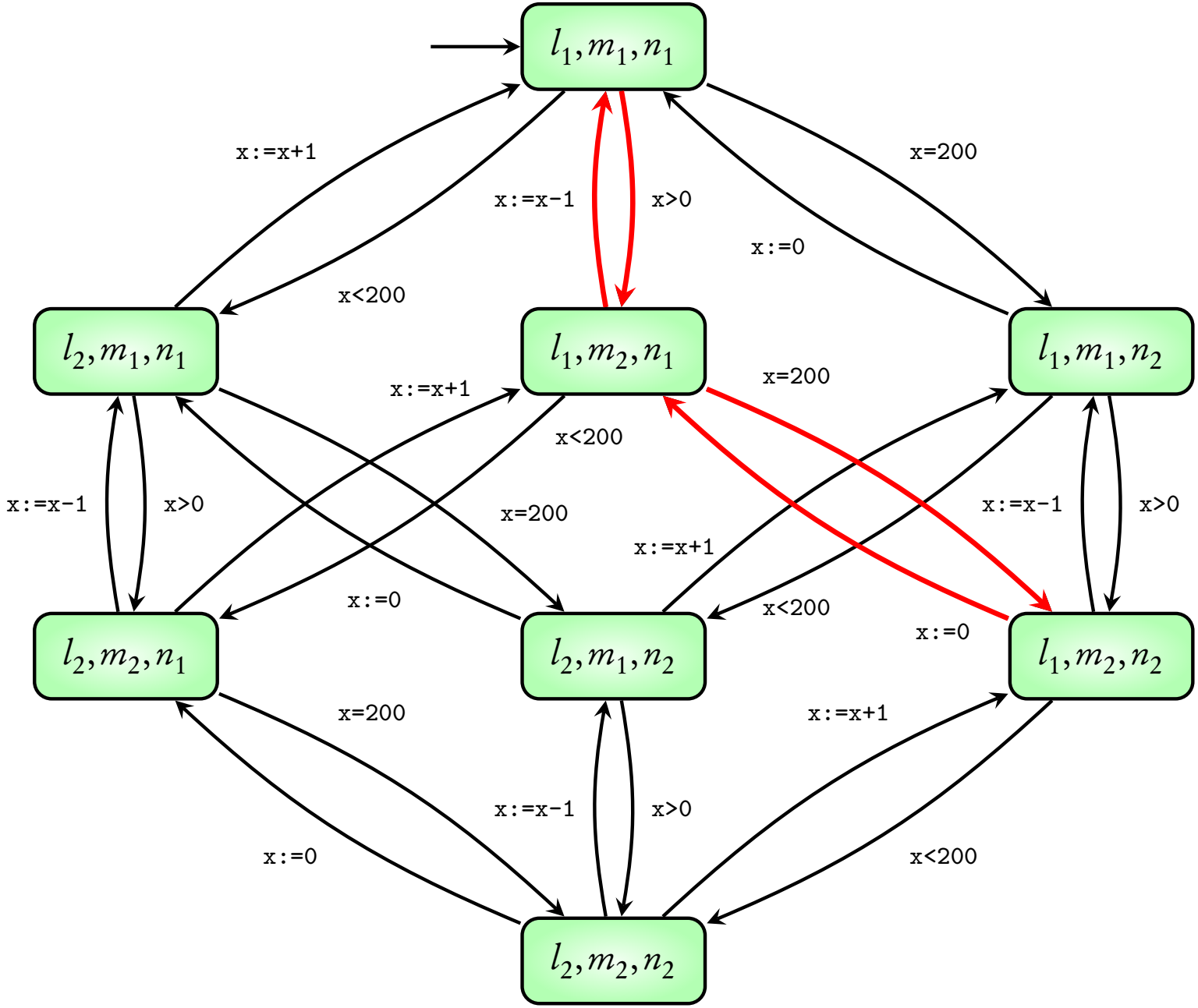


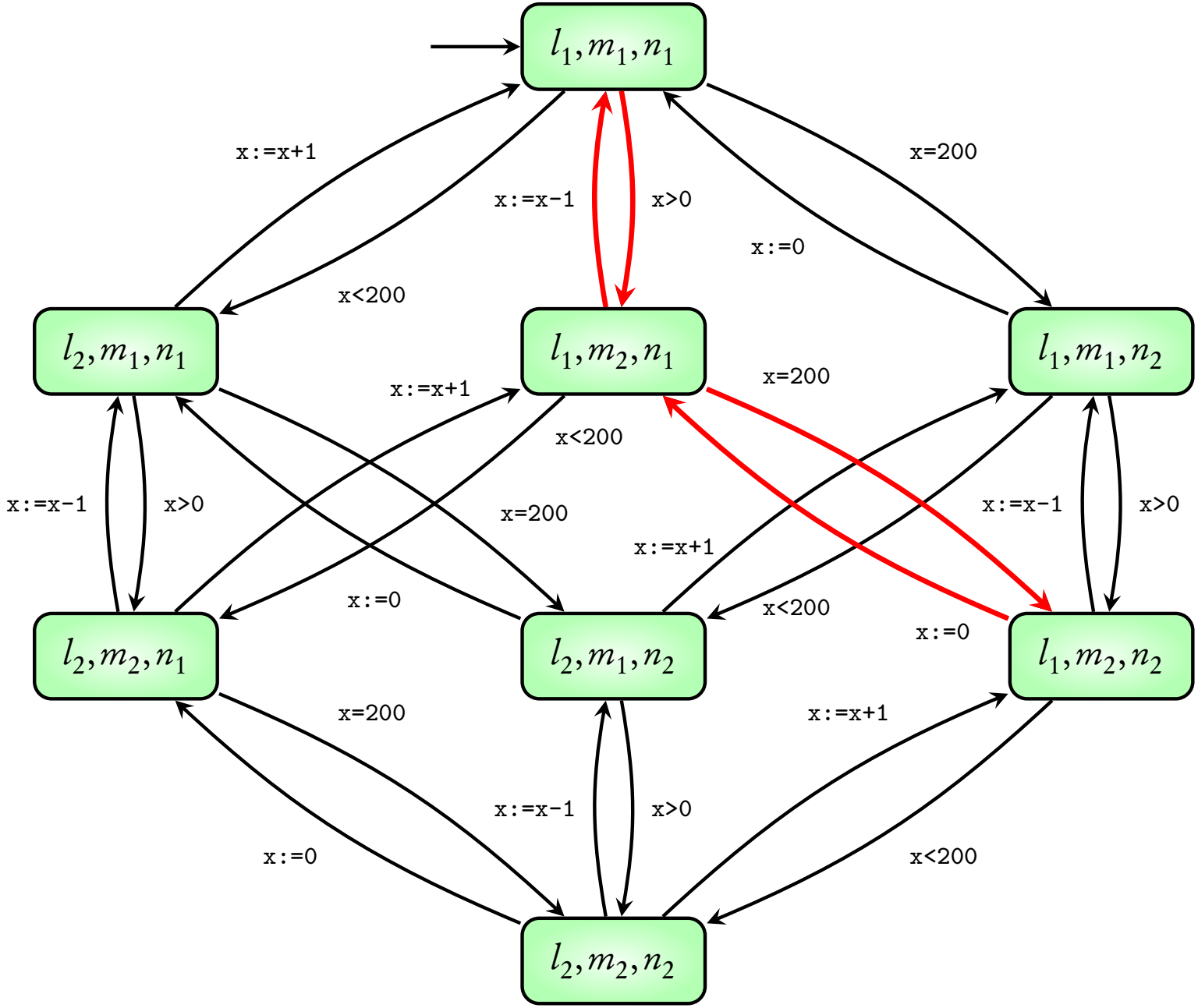




Is the value of x **always between** 0 and 200?


l_1, m_1, n_1
l_2, m_1, n_1
l_1, m_2, n_1
l_1, m_1, n_2
l_2, m_2, n_1
l_2, m_1, n_2
l_1, m_2, n_2
l_2, m_2, n_2
x:=x+1
x:=x-1
x>0
x=200
x:=0
x<200
x:=x+1
x=200
x<200
x:=x-1
x>0
x=200
x:=x+1
x:=0
x<200
x:=0
x=200
x:=x-1
x>0
x:=x+1
x:=0
x<200


l_1, m_1, n_1
l_2, m_1, n_1
l_1, m_2, n_1
l_1, m_1, n_2
l_2, m_2, n_1
l_2, m_1, n_2
l_1, m_2, n_2
l_2, m_2, n_2
x:=x+1
x:=x-1
x>0
x=200
x:=0
x<200
x:=x+1
x=200
x<200
x:=x-1
x>0
x=200
x:=x+1
x:=0
x<200
x:=0
x=200
x:=x+1
x:=x-1
x>0
x:=0
x<200



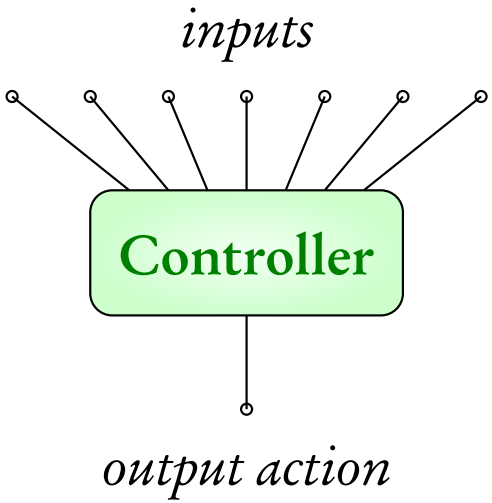
Is the value of x **always between** 0 and 200? **No**

Instead of writing the code directly, the functionality is **specified as a suitable mathematical model** (extensions of finite state machines)

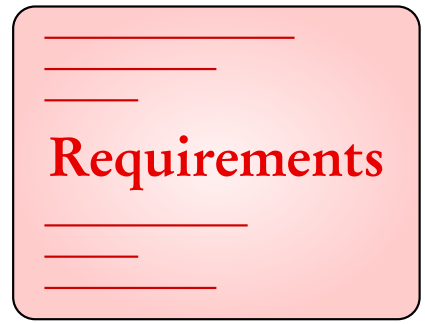
This **mathematical object** can then be **analyzed**.

The final **code can be generated directly** from the model.

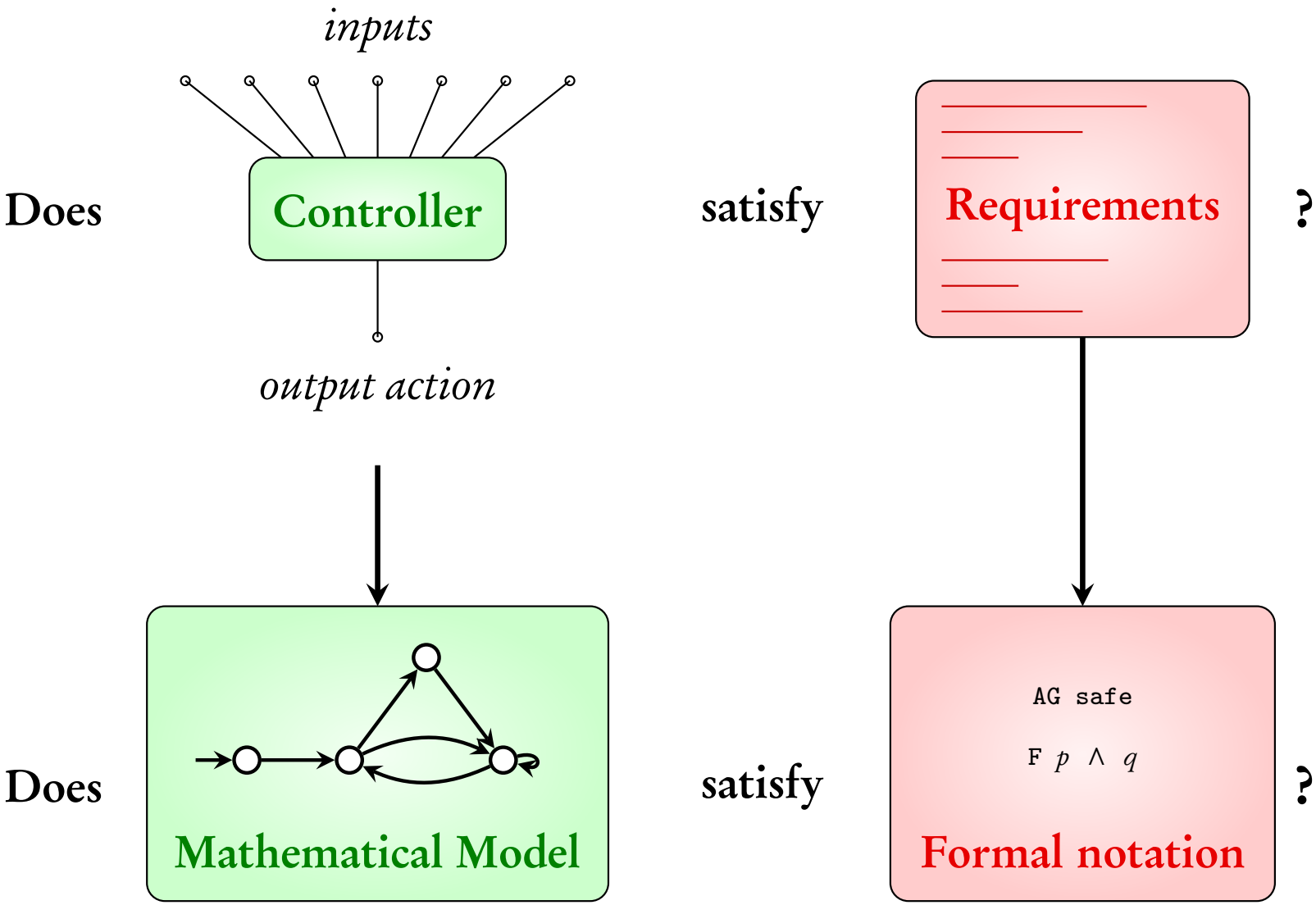
Does

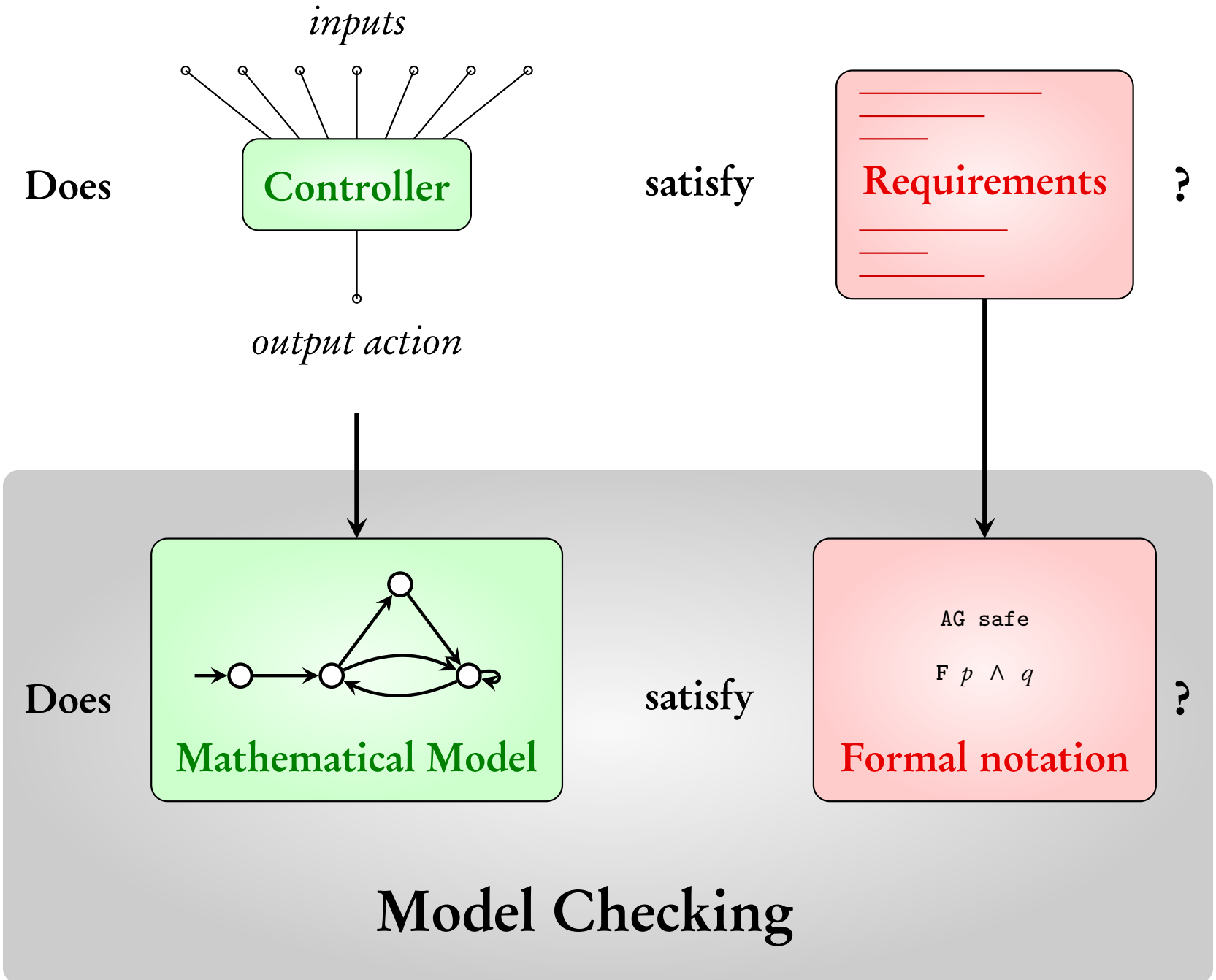


satisfy

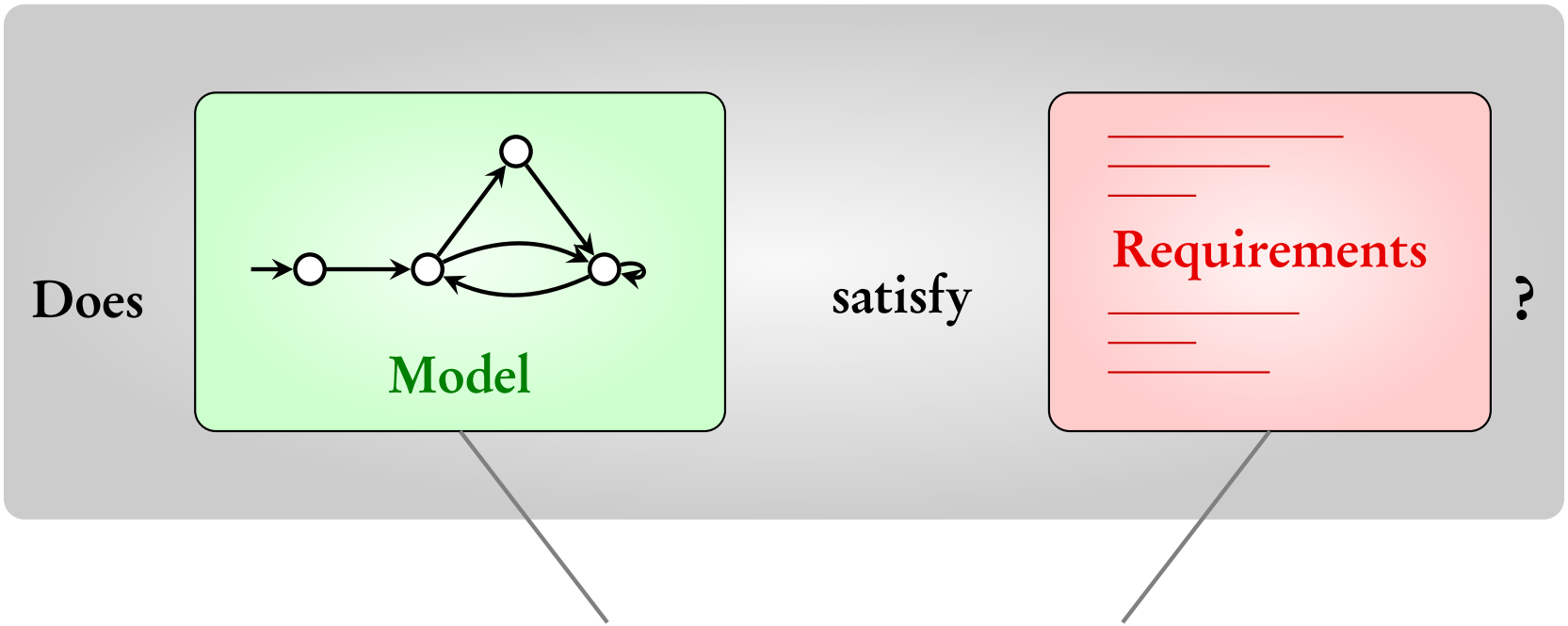


?

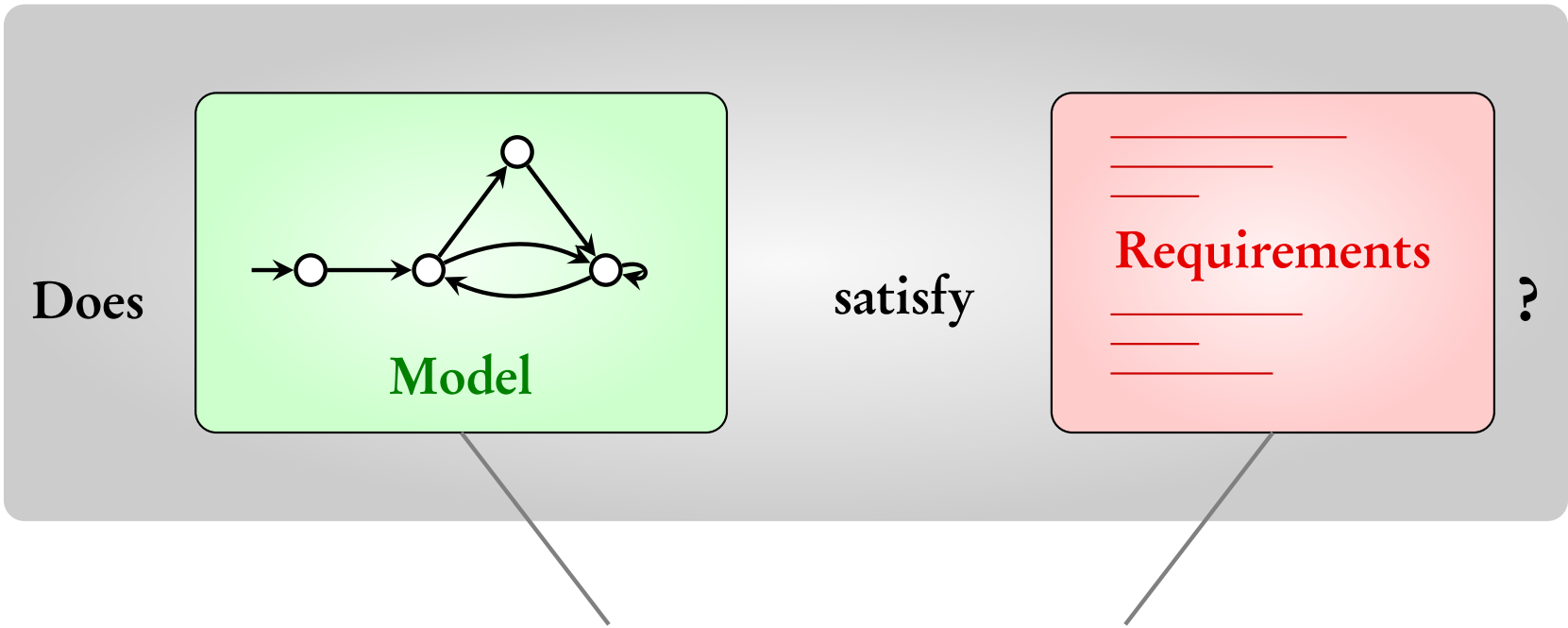

inputs
Does
Controller
satisfy
Requirements
?
output action
Does
Mathematical Model
satisfy
AG safe
F p ∧ q
Formal notation
?


inputs
Does
Controller
satisfy
Requirements
?
output action
Does
Mathematical Model
satisfy
AG safe
F p ∧ q
Formal notation
?
Model Checking

Model-checkers

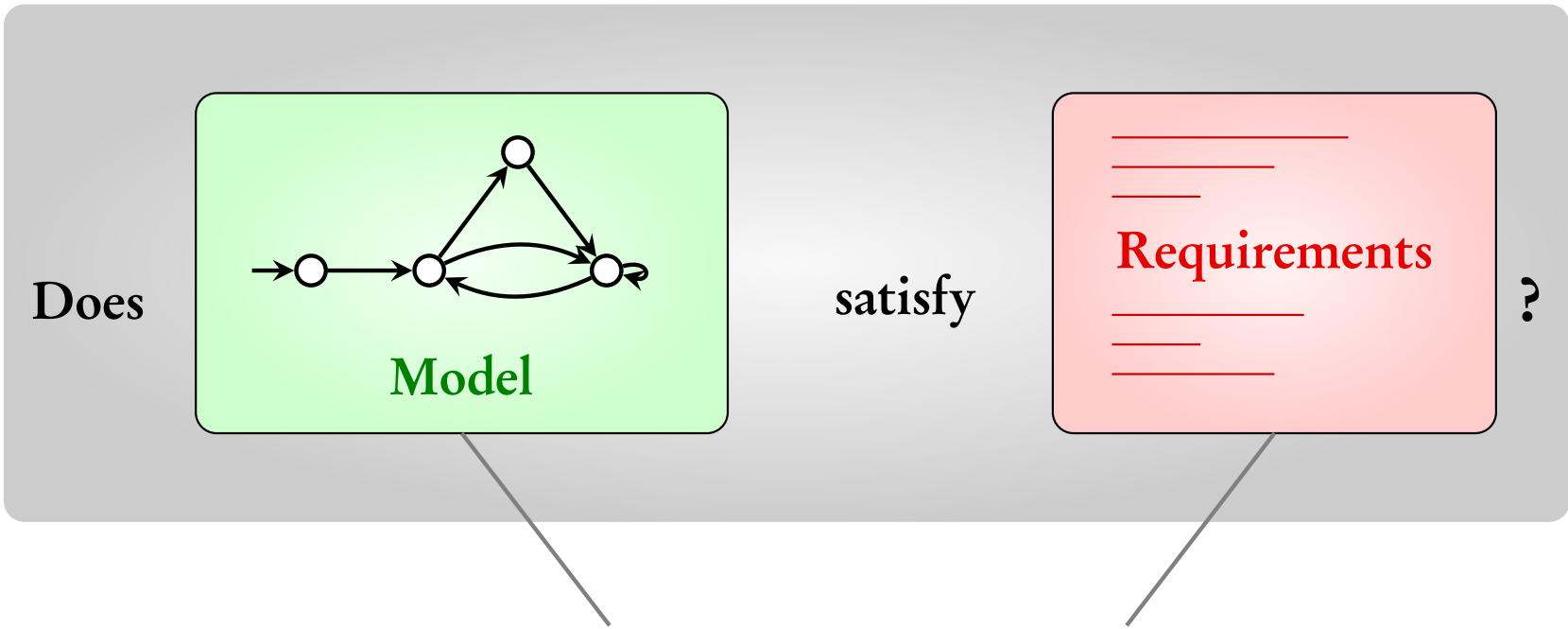


Model-checkers



Model-checkers **automatically** solve the above question

Model-checkers



Model-checkers **automatically** solve the above question

Some model-checkers: NuSMV, SPIN, TLA+, SCADE Suite

Success of Model-checking

Airbus

- Uses SCADE Suite (developed by Esterel Technologies) to **develop critical on board software** for A340-500/600, A380 series aircrafts
- **Significant decrease of coding errors** due to extensive use of automatic code generation. For Airbus A340, up to 70% of the code has been automatically generated
- **Major productivity improvement**, which is particularly significant considering that the size of the software doubles with each new Airbus program

Source: Website of Esterel Technologies

Hardware verification

- Many companies, including industry leaders such as **AT&T, Cadence, Hewlett-Packard, IBM, Intel, LSI Logic, Motorola, Rockwell, Texas Instruments, and Silicon Graphics** have created formal verification groups to help with ongoing designs.

- In many cases, these groups began by demonstrating the effectiveness of formal verification by **finding subtle design errors** that were overlooked by months of simulation.

Source: Acceptance of formal methods: Lessons from hardware design, by D. Dill and J. Rushby

Amazon

- Since 2011, engineers at Amazon Web Services (AWS) have used formal specification and model checking to help **solve difficult design problems in critical systems**

Source: How Amazon Web Services Uses Formal Methods, by C. Newcombe et al.

Some other places where **Model Checking** technology is **used**

- *Avionics:* Rockwell Collins, Honeywell
- *Automobiles:* Toyota
- *Space:* NASA, European Space Agency
- *Others:* Microsoft Research, Tata

- *Model-checking solutions:*

  Esterel technologies, BTC embedded systems, Mathworks, Prover technology

Some other places where **Model Checking** technology is **used**

- *Avionics:* Rockwell Collins, Honeywell
- *Automobiles:* Toyota
- *Space:* NASA, European Space Agency
- *Others:* Microsoft Research, Tata
- *Model-checking solutions:*
  Esterel technologies, BTC embedded systems, Mathworks, Prover technology

Backed by many **university groups** from all over the world!

Turing Award 2007

Clarke, Emerson and Sifakis for Model-checking

Some other Turing award winners:

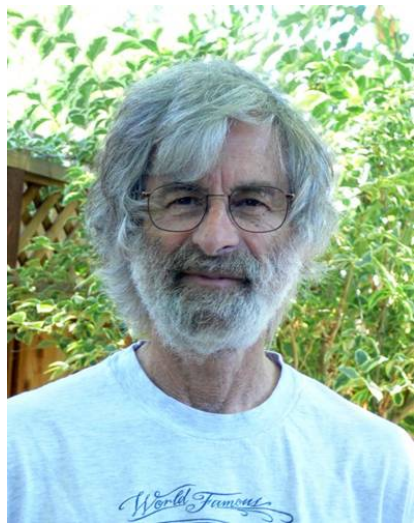
- Edsger Dijkstra (1972)
- Donald Knuth (1974)
- Rabin and Scott (1976)
- Tony Hoare (1980)
- Ritchie and Thompson (1983)
- Hopcroft and Tarjan (1986)
- Rivest, Shamir, Adleman (2002)

Turing Award 1996

Amir Pnueli

Pnueli received the Turing Award for seminal work **introducing temporal logic** into computing science and for outstanding contributions to **program and systems verification**

Turing Award 2013



Leslie Lamport

He devised important algorithms and developed **formal modeling and verification protocols** that improve the quality of **real distributed systems**. These contributions have resulted in **improved correctness, performance, and reliability of computer systems**.

What we have seen?

- Software control many **safety-critical systems**
- **Accidents** do occur due to **software errors**
- Model-checking is an additional **verification method**
- Model-checking has been **successful**

In this course

Introduction to **techniques, tools and challenges** in model-checking

- **Part 1: (Srivathsan)** Basic concepts, Automata-theoretic methods
- **Part 2: (Srivas)** Advanced concepts, Symbolic model-checking

Book: Principles of Model Checking,
Christel Baier and Joost-Pieter Katoen, MIT Press (2008)