

IA159 Formal Verification Methods

Verification via Automata, Symbolic Execution,
and Interpolation

Jan Strejček

Faculty of Informatics
Masaryk University

Focus

- basic principle
- interpolation
- example

ULTIMATE



Sources

- M. Heizmann, J. Hoenicke, A. Podelski: [Software Model Checking for People Who Love Automata](#), CAV 2013.
- M. Heizmann, J. Hoenicke, A. Podelski: [Termination Analysis by Learning Terminating Programs](#), CAV 2014.

Special thanks to M. Heizmann for providing me his slides.

New view on programs

A program defines a language over program statements.

- **alphabet** = the set of program statements
- **finite automaton** = control flow graph
- **accepting states** = error locations

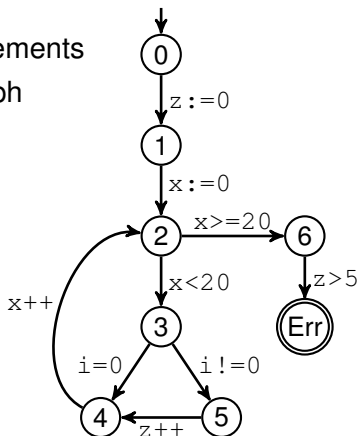
Programs as languages

New view on programs

A program defines a language over program statements.

- **alphabet** = the set of program statements
- **finite automaton** = control flow graph
- **accepting states** = error locations

```
z:=0
x:=0
while x<20 {
  if i!=0 then z++
  x++
}
assert (z<=5)
```



The goal

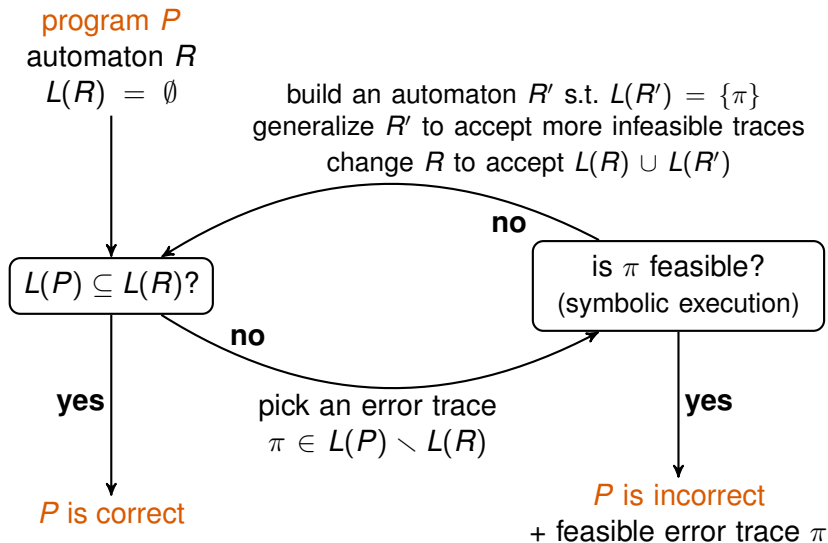
- such an automaton accepts **error traces**
- not all error traces are feasible

The goal

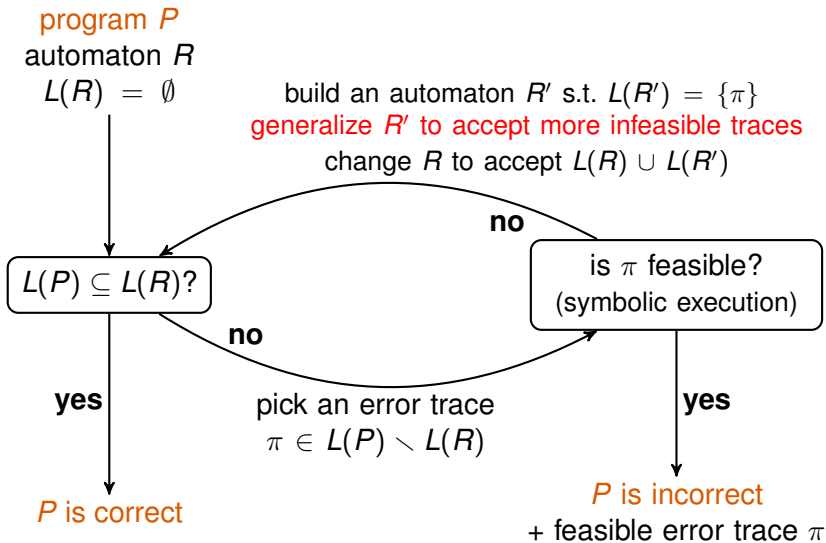
To decide whether there exists a feasible error trace accepted by the automaton.

The program is correct iff all error traces accepted by the automaton are infeasible.

The algorithm



The algorithm

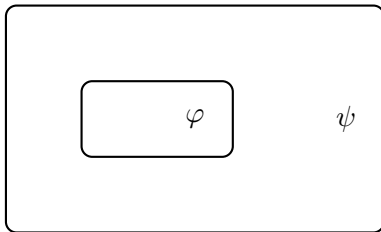


Craig interpolation

Theorem (William Craig, 1957)

Let φ, ψ be two first-order formulae such that $\varphi \implies \psi$. Then there exists a first order-formula θ called *interpolant* such that

- all non-logical symbols in θ occur in both φ and ψ ,
- $\varphi \implies \theta \implies \psi$.

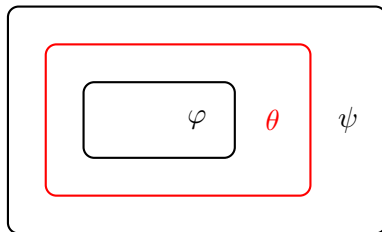


Craig interpolation

Theorem (William Craig, 1957)

Let φ, ψ be two first-order formulae such that $\varphi \implies \psi$. Then there exists a first order-formula θ called *interpolant* such that

- all non-logical symbols in θ occur in both φ and ψ ,
- $\varphi \implies \theta \implies \psi$.

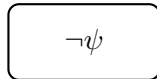


Craig interpolation

Theorem (William Craig, 1957)

Let φ, ψ be two first-order formulae such that $\varphi \implies \psi$. Then there exists a first order-formula θ called *interpolant* such that

- all non-logical symbols in θ occur in both φ and ψ ,
- $\varphi \implies \theta \implies \psi$.

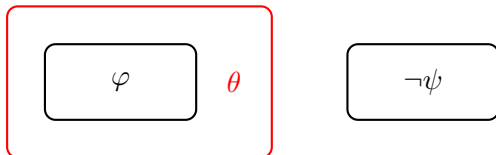


Craig interpolation

Theorem (William Craig, 1957)

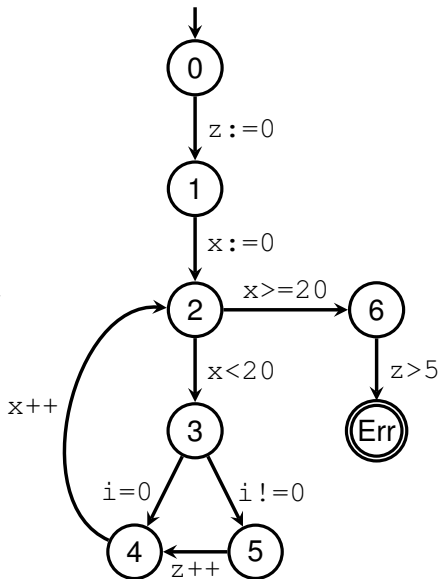
Let φ, ψ be two first-order formulae such that $\varphi \implies \psi$. Then there exists a first order-formula θ called *interpolant* such that

- all non-logical symbols in θ occur in both φ and ψ ,
- $\varphi \implies \theta \implies \psi$.



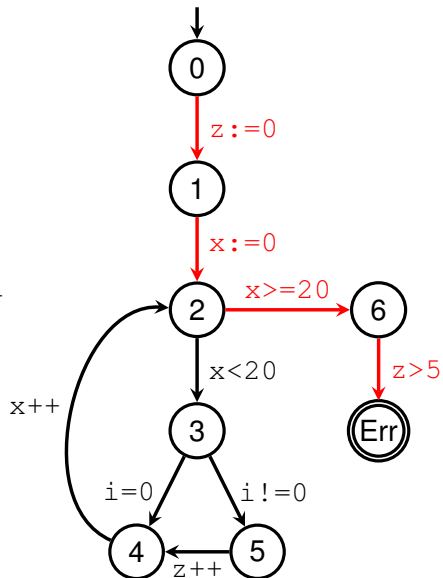
Example: program

```
z:=0
x:=0
while x<20 {
  if i!=0 then z++
  x++
}
assert (z<=5)
```

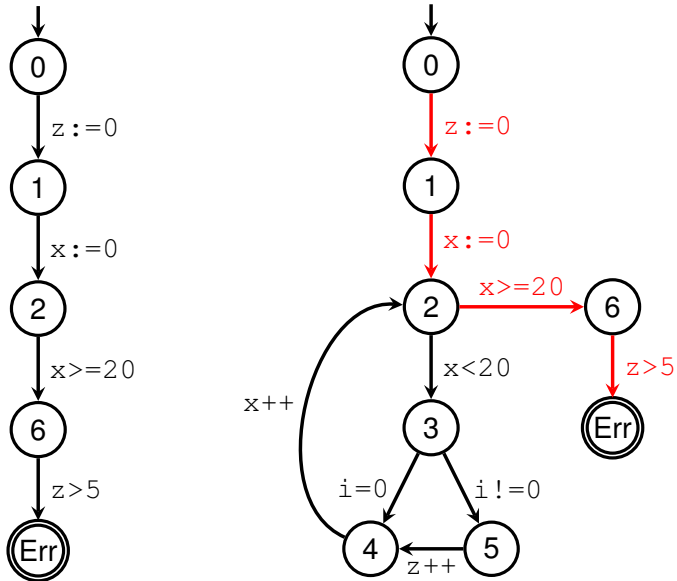


Example: error path

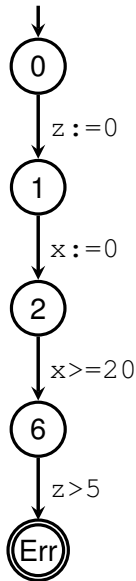
```
z:=0
x:=0
while x<20 {
  if i!=0 then z++
  x++
}
assert (z<=5)
```



Example: error path

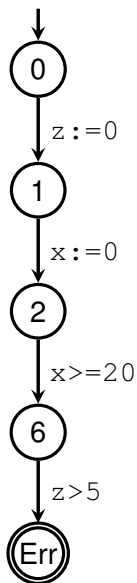


Example: feasibility analysis



symbolic execution produces
 $z = 0 \wedge x = 0 \wedge x \geq 20 \wedge z > 5$
 $\equiv \text{false}$
 \implies the error trace is **infeasible**

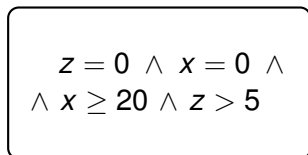
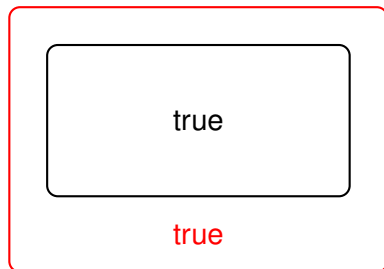
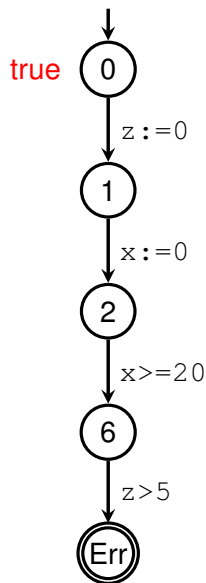
Example: generalization of infeasibility arguments



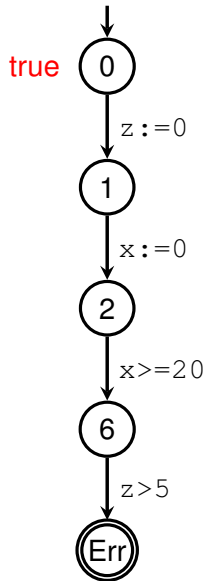
true

$$z = 0 \wedge x = 0 \wedge \\ \wedge x \geq 20 \wedge z > 5$$

Example: generalization of infeasibility arguments



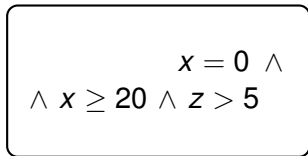
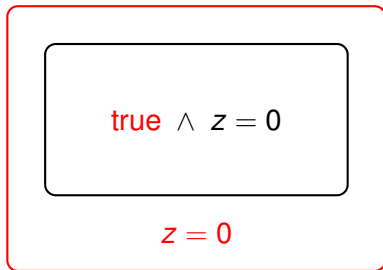
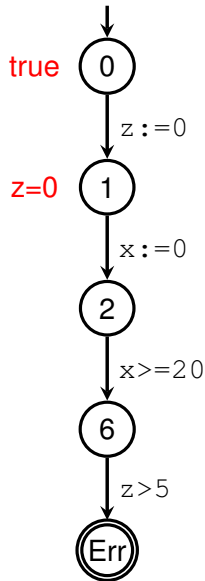
Example: generalization of infeasibility arguments



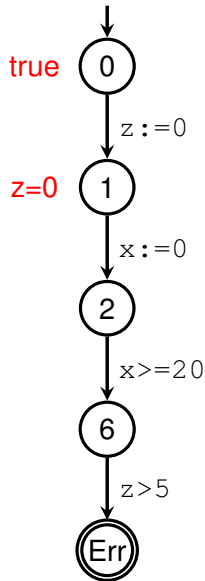
$\text{true} \wedge z = 0$

$x = 0 \wedge$
 $\wedge x \geq 20 \wedge z > 5$

Example: generalization of infeasibility arguments



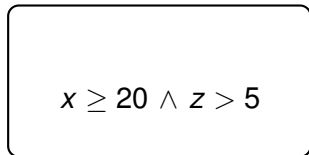
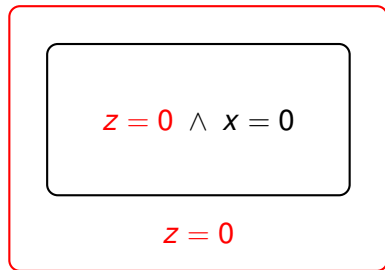
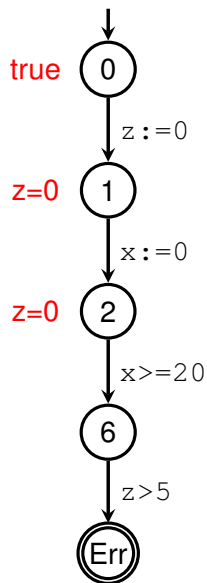
Example: generalization of infeasibility arguments



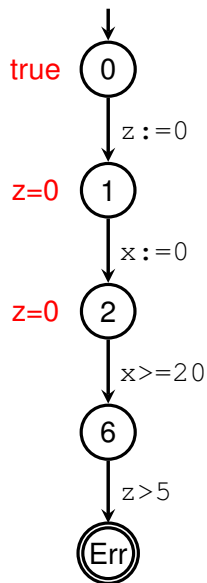
$$z = 0 \wedge x = 0$$

$$x \geq 20 \wedge z > 5$$

Example: generalization of infeasibility arguments



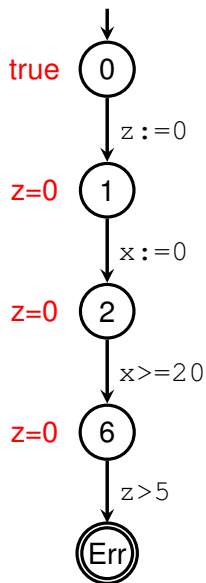
Example: generalization of infeasibility arguments



$$z = 0 \wedge x \geq 20$$

$$z > 5$$

Example: generalization of infeasibility arguments

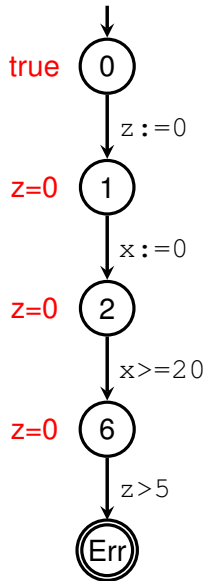


$$z = 0 \wedge x \geq 20$$

$$z = 0$$

$$z > 5$$

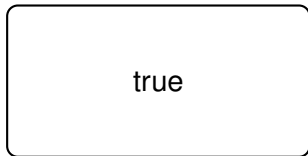
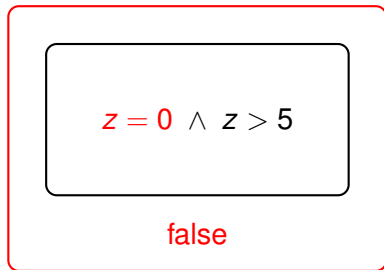
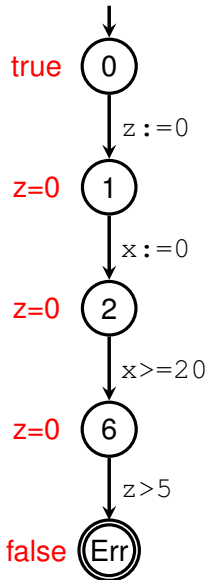
Example: generalization of infeasibility arguments



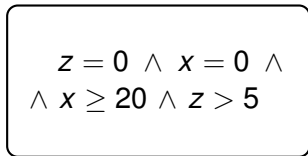
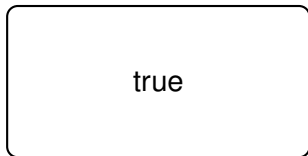
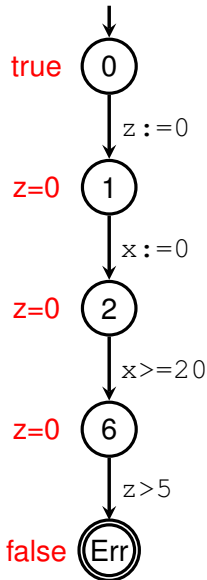
$$z = 0 \wedge z > 5$$

true

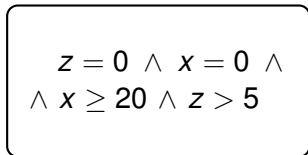
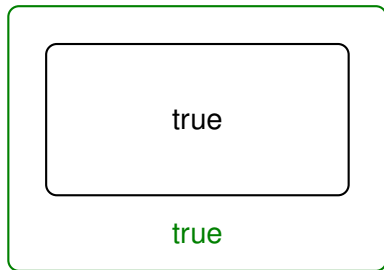
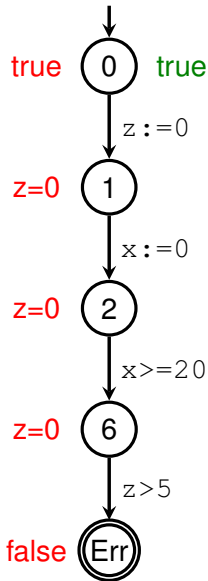
Example: generalization of infeasibility arguments



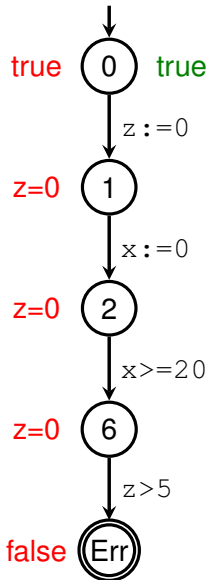
Example: generalization of infeasibility arguments



Example: generalization of infeasibility arguments



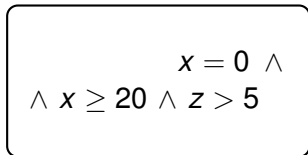
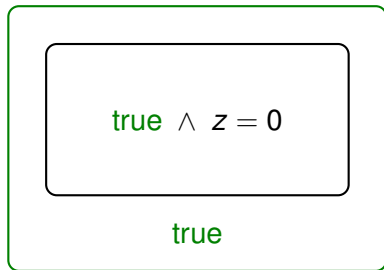
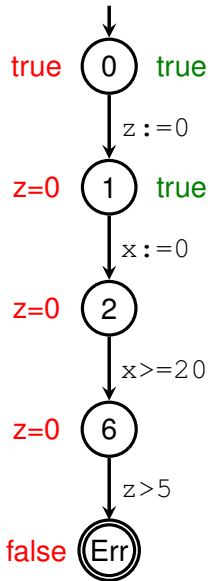
Example: generalization of infeasibility arguments



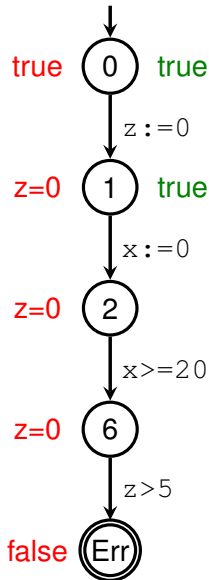
$\text{true} \wedge z = 0$

$x = 0 \wedge$
 $\wedge x \geq 20 \wedge z > 5$

Example: generalization of infeasibility arguments



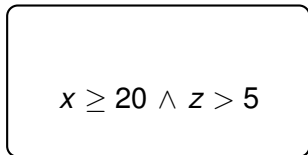
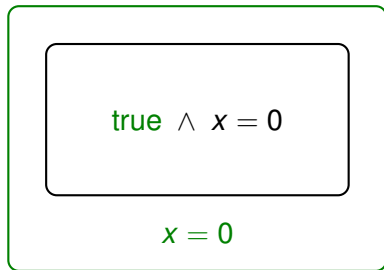
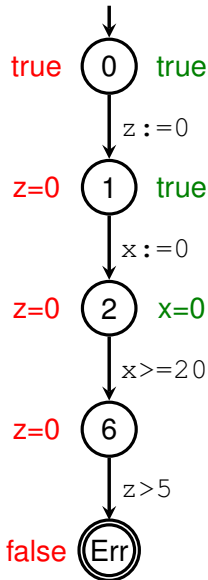
Example: generalization of infeasibility arguments



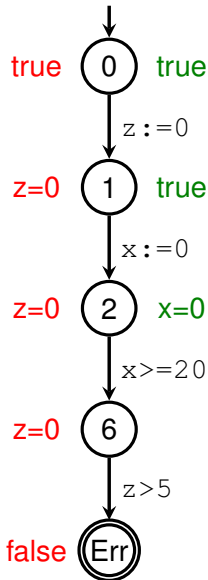
$\text{true} \wedge x = 0$

$x \geq 20 \wedge z > 5$

Example: generalization of infeasibility arguments



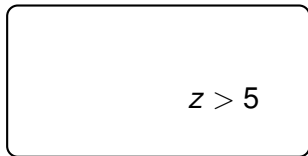
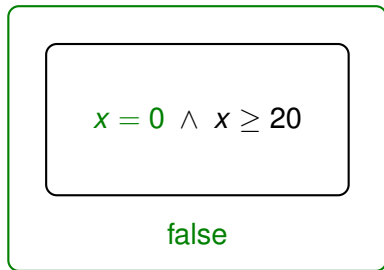
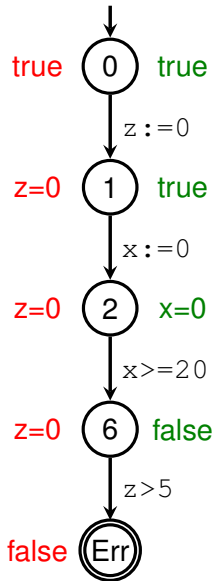
Example: generalization of infeasibility arguments



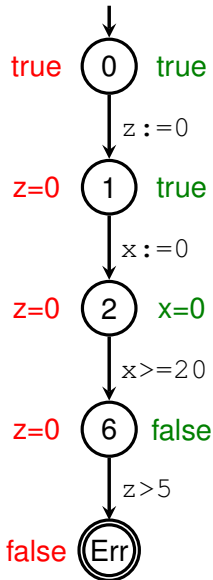
$$x = 0 \wedge x \geq 20$$

$$z > 5$$

Example: generalization of infeasibility arguments



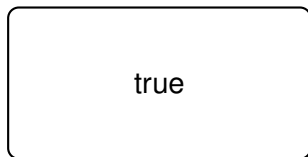
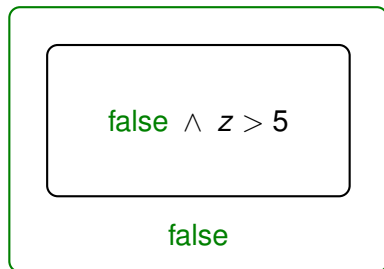
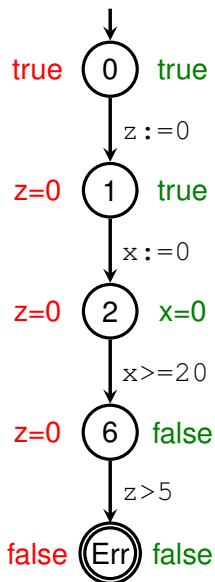
Example: generalization of infeasibility arguments



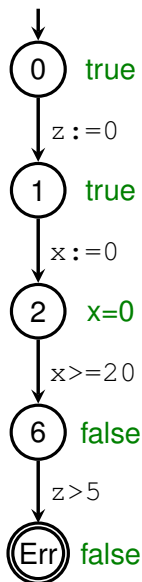
$\text{false} \wedge z > 5$

true

Example: generalization of infeasibility arguments



Example: generalization of infeasibility arguments



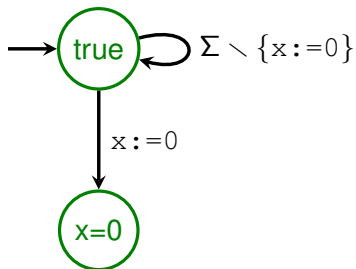
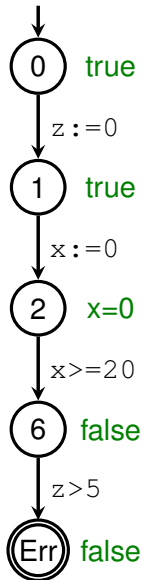
true

x=0

false

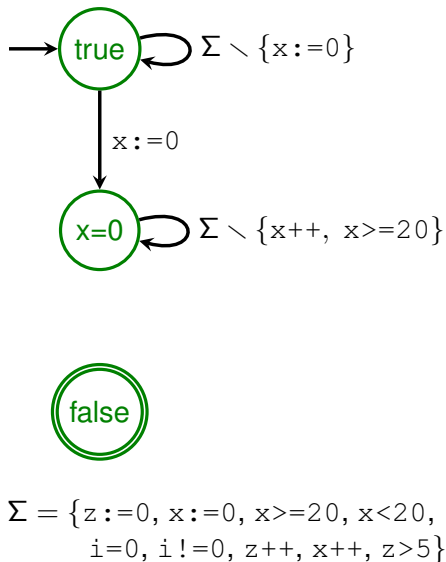
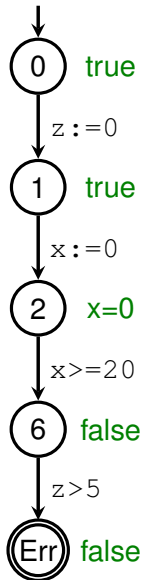
$$\Sigma = \{z := 0, x := 0, x \geq 20, x < 20, i = 0, i \neq 0, z ++, x ++, z > 5\}$$

Example: generalization of infeasibility arguments

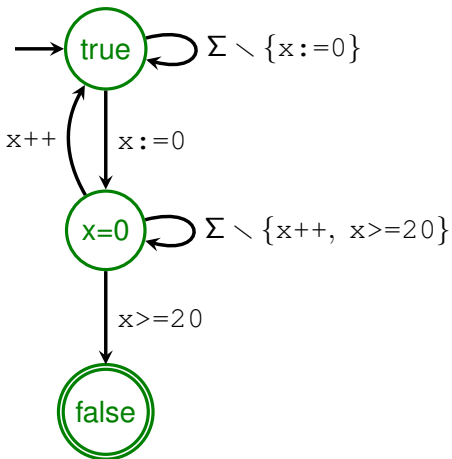
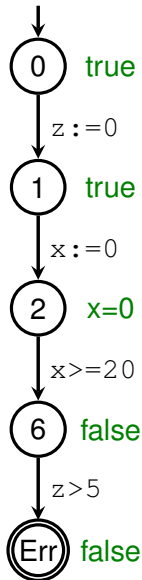


$$\Sigma = \{z := 0, x := 0, x \geq 20, x < 20, i = 0, i \neq 0, z ++, x ++, z > 5\}$$

Example: generalization of infeasibility arguments

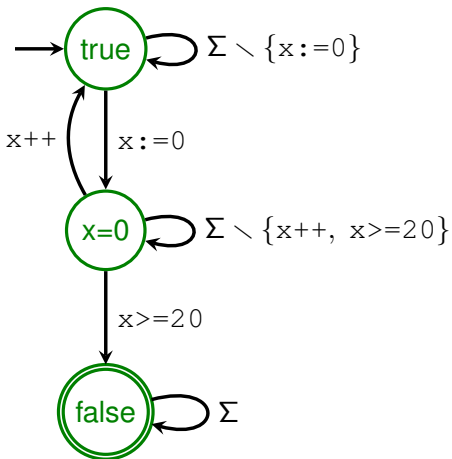
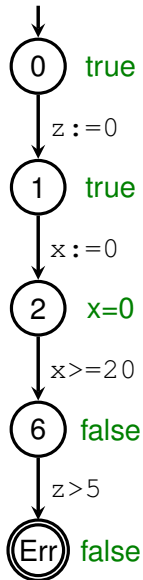


Example: generalization of infeasibility arguments



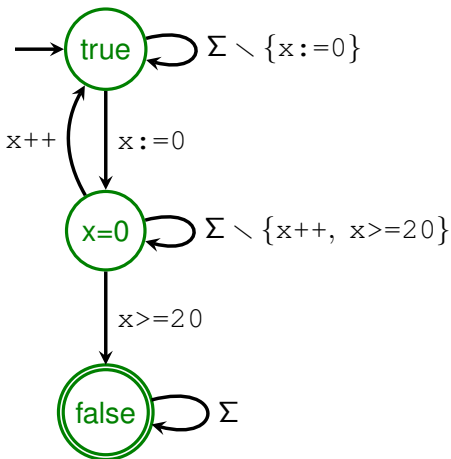
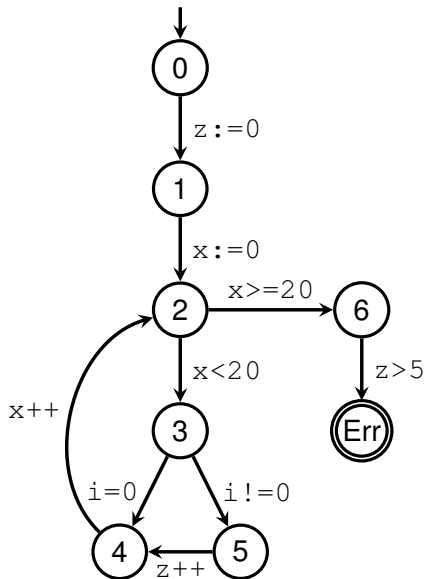
$$\Sigma = \{z := 0, x := 0, x \geq 20, x < 20, i = 0, i \neq 0, z++, x++, z > 5\}$$

Example: generalization of infeasibility arguments



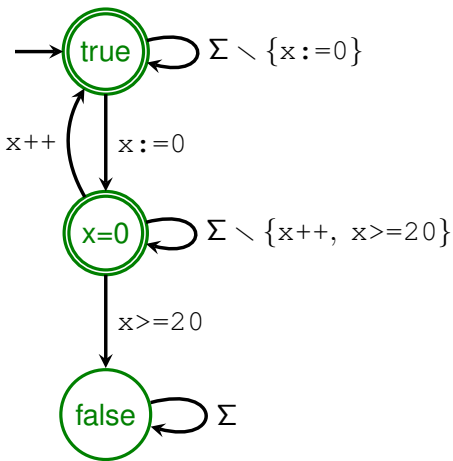
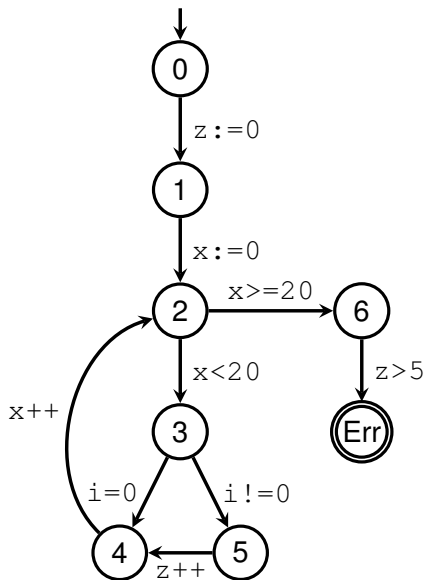
$$\Sigma = \{z:=0, x:=0, x>=20, x<20, i=0, i!=0, z++, x++, z>5\}$$

Example: $L(P) \subseteq L(R)$?



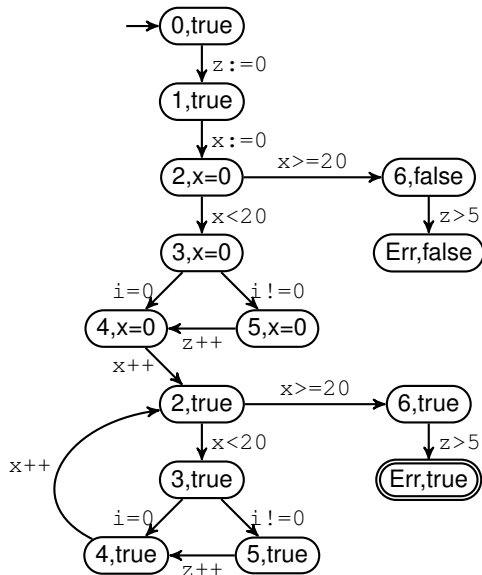
$L(P) \subseteq L(R)$ iff the language of the synchronous product of P and complemented R is empty.

Example: $L(P) \subseteq L(R)$?

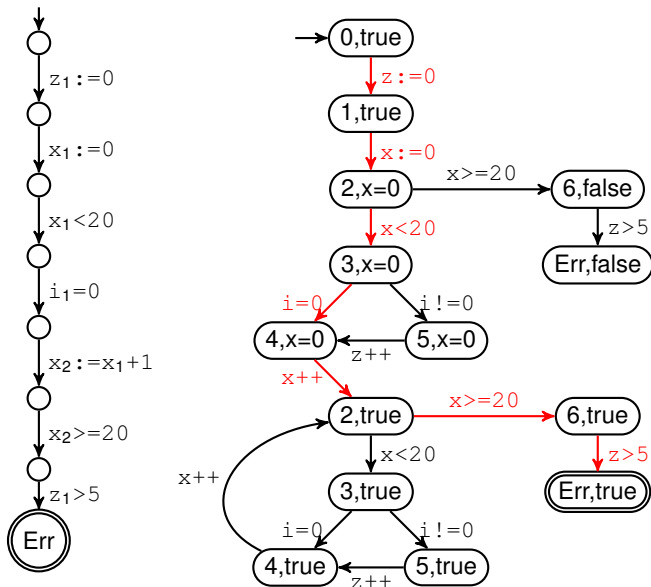


$L(P) \subseteq L(R)$ iff the language of the synchronous product of P and **complemented R** is empty.

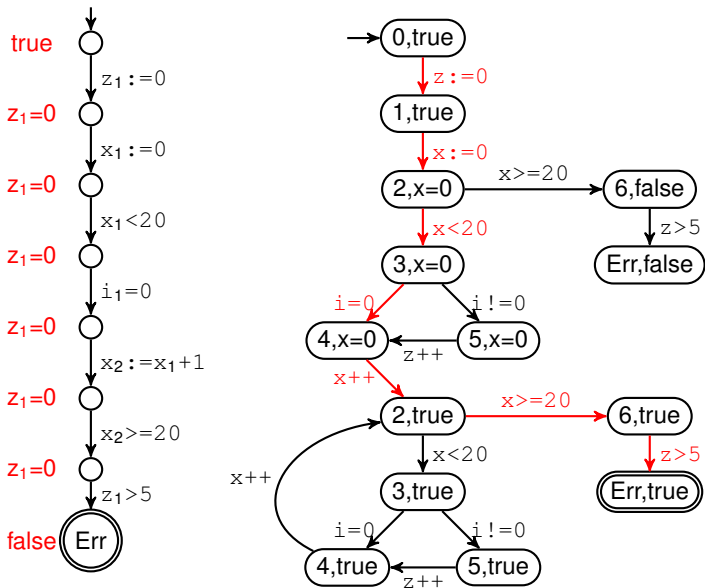
Example: the product



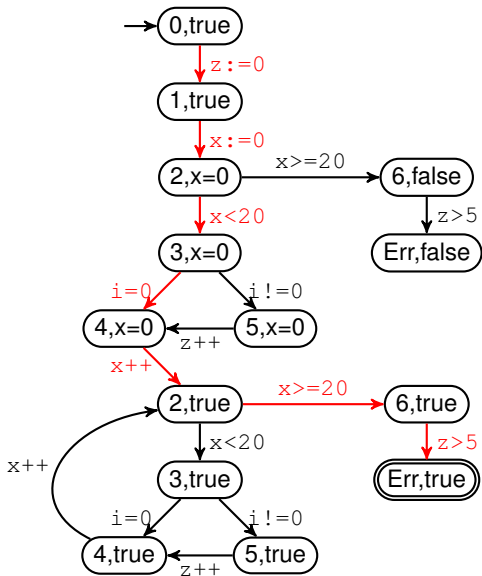
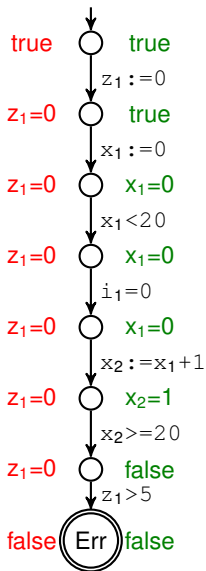
Example: the product



Example: the product



Example: the product



- interpolation algorithms exist only for some logics/theories
- to handle function calls and unbounded recursion, nested word automata and nested interpolants are needed
- implemented in **Ultimate Automizer** with numerous optimizations, e.g. on-the-fly complementation and emptiness check of the product
- extensions
 - for termination analysis: **Ultimate Büchi Automizer**
 - for LTL properties: **Ultimate LTL Automizer**

Try it out online:

<https://monteverdi.informatik.uni-freiburg.de/tomcat/Website/>

Property-Directed Reachability (PDR/IC3)

- Another approach based on interpolation.
- Very successful in hardware verification (IC3) and popular in software verification.