IKOS

Static Analysis using Abstract Interpretation

Maxime Arthaud

Ames Research Center

NASA

Table of contents

Table of contents

Software development

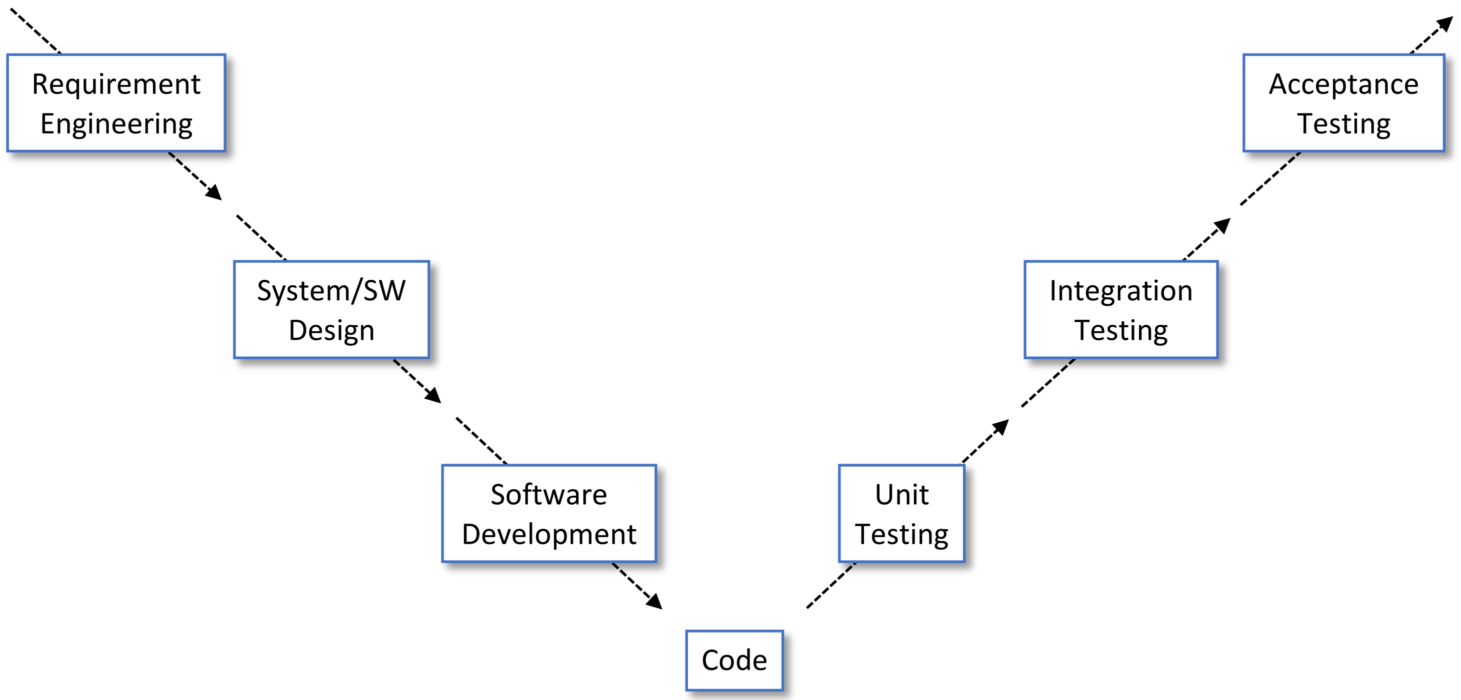
- Software represent **more than half** of the development **cost** of an aircraft

- **Safety Critical** : Failure is not an option

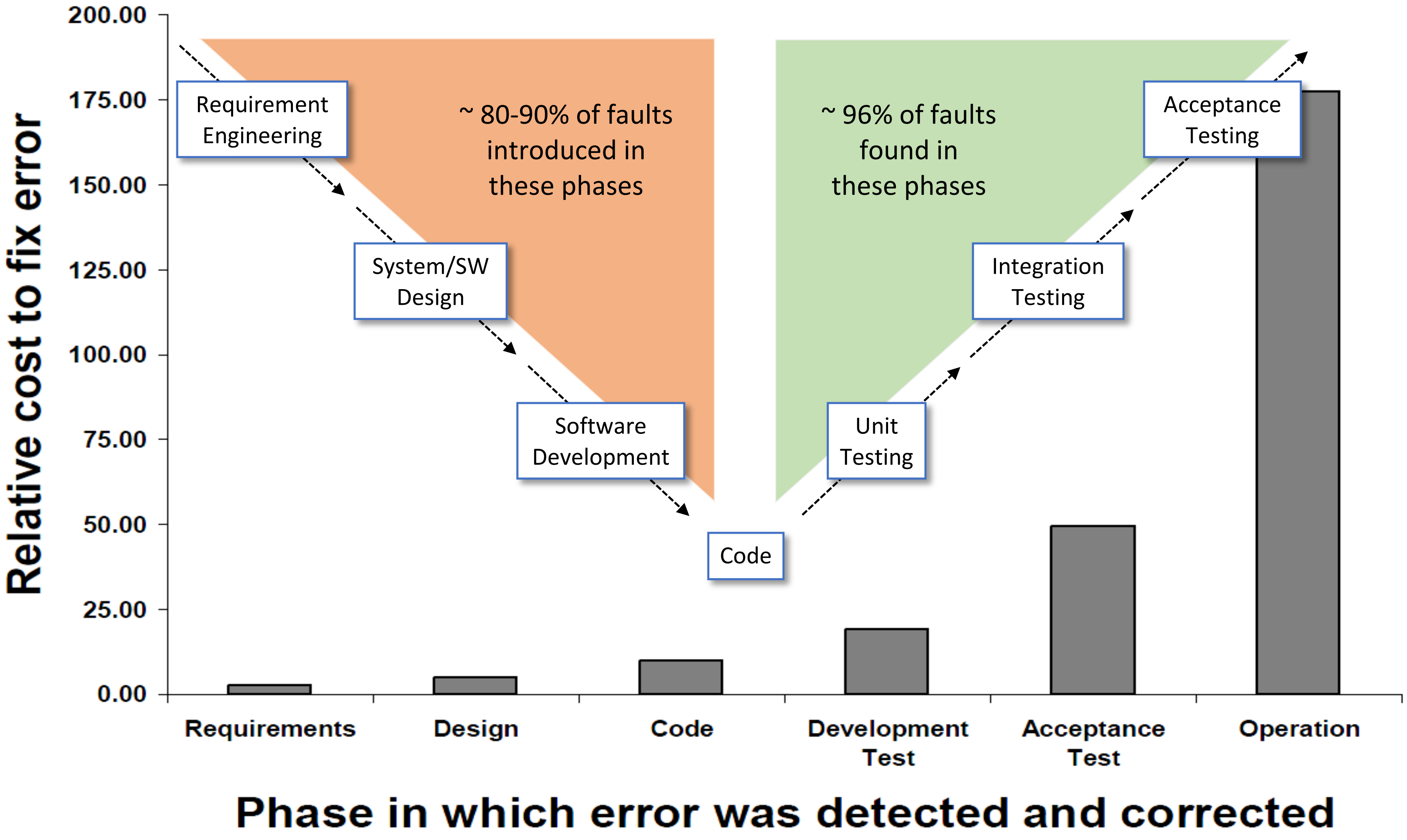
- Regulated by international standards : **DO-178 rev. B/C**

- Robust Software Engineering

V Model


Requirement Engineering
System/SW Design
Software Development
Code
Unit Testing
Integration Testing
Acceptance Testing

Cost Analysis



Validation and Verification

- **Validation:**

  The software **meets** the **needs** of the user

  → Are we building the right thing?

- **Verification:**

  The software **meets** the **specification**

  → Are we building it right?

Formal Verification

- **Formal Verification**:

  Prove the correctness of a program with respect to a **formal specification**

- **Formal Specification:**

  Mathematical description of a software

Properties

- **Safety property:**

  Something **bad** will **never** happen

- **Liveness property:**

  Something **good** will **eventually** happen

Formal Methods

- **Abstract Interpretation:**
  Compute an over approximation of all the reachable states
- **Model Checking:**
  Explore all the reachable states
- **Symbolic Execution:**
  Explore interesting paths with symbolic inputs
- **Theorem Proving:**
  Prove properties manually or with heuristic algorithms
- Etc...

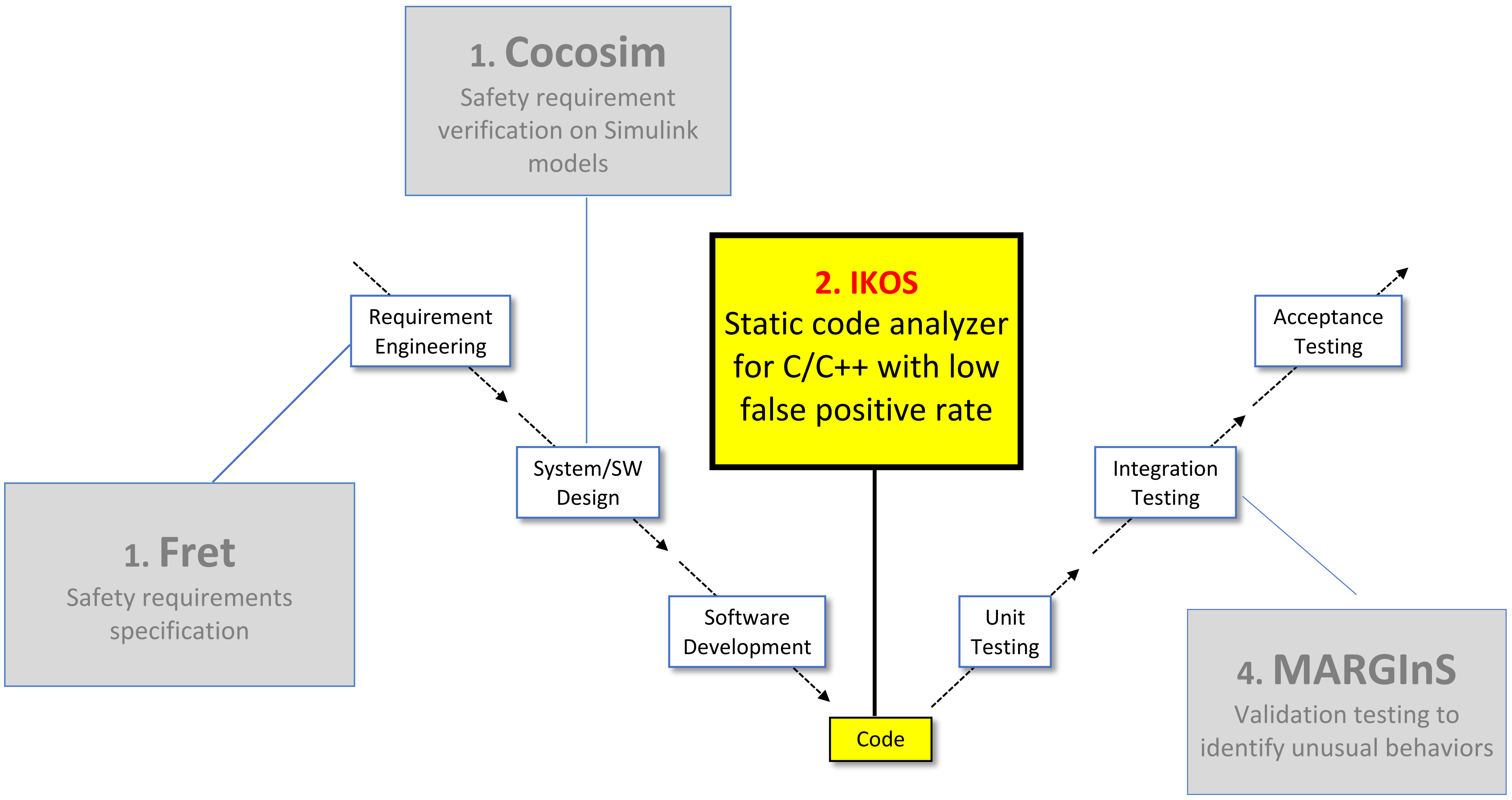
Static Analysis

- **Static Analysis:**

  Analysis of a software **without** actually **executing** it
- Opposite of Dynamic Analysis
- Can be performed on the **source code** or **binary code**
- Objectives:
  - Find **runtime errors**
  - Measure **metrics**
  - Reverse engineering
  - Etc...

Table of contents

IKOS


1. Cocosim
Safety requirement verification on Simulink models
1. Fret
Safety requirements specification
Requirement Engineering
System/SW Design
Software Development
Code
2. IKOS
Static code analyzer for C/C++ with low false positive rate
Unit Testing
Integration Testing
Acceptance Testing
4. MARGInS
Validation testing to identify unusual behaviors

IKOS

IKOS performs a **compile-time** analysis of a C/C++ source code.

It can **detect** or **prove the absence** of **runtime errors.**

```
int tab[10];

for (int i = 0; i < 10; i++) {
  tab[i] = i * i;
}
```

C/C++ code

IKOS Static Analyzer

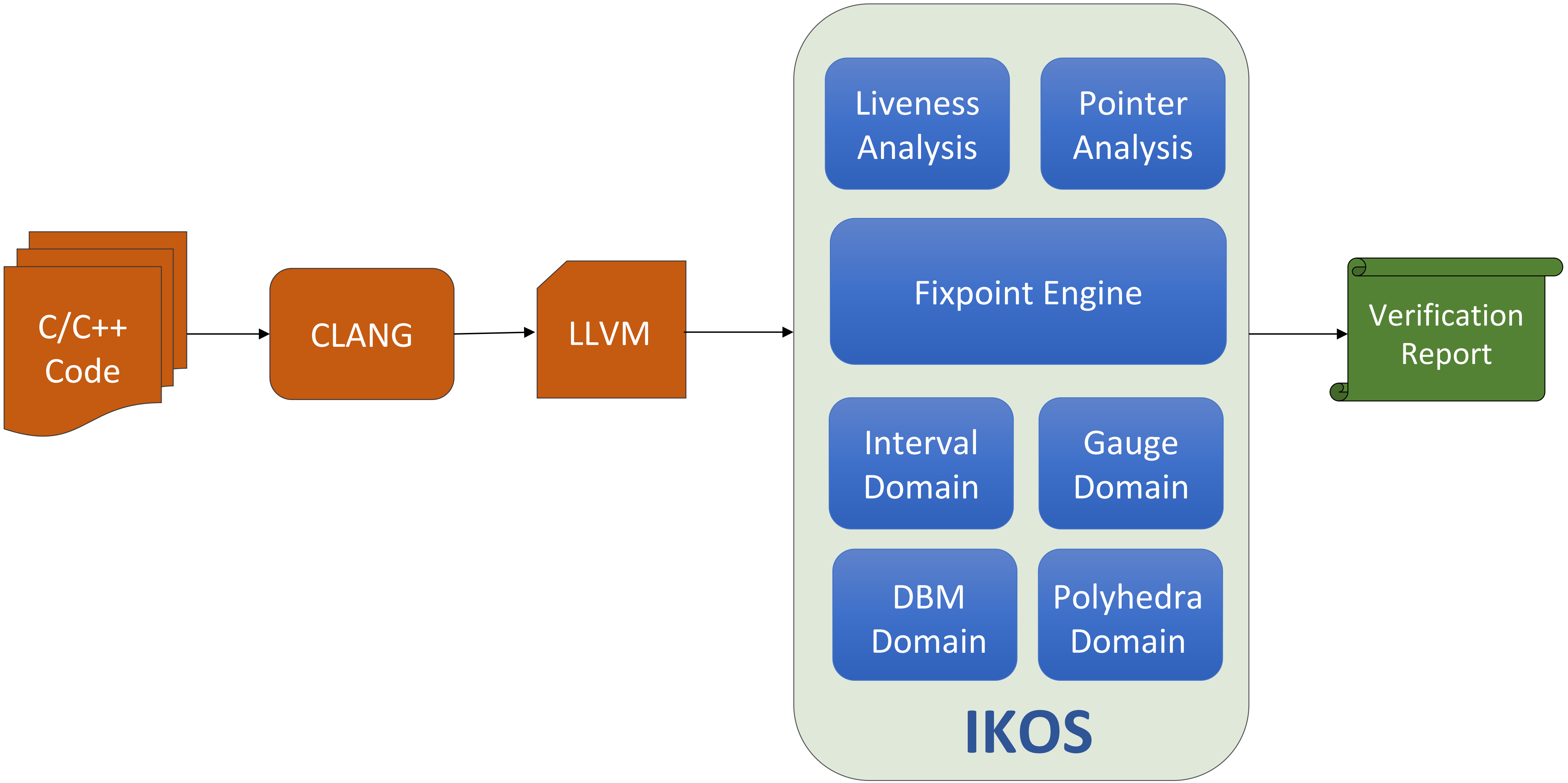
List of (possible) runtime errors:

- Buffer Overflows
- Null pointers
- Division by Zero
- Uninitialized Variables
- Assertion Prover
- Etc.

IKOS is **NOT** a code style checker

IKOS is **NOT** a compiler: It can detect errors that compilers cannot catch

IKOS Design



C/C++
Code
CLANG
LLVM
Liveness
Analysis
Pointer
Analysis
Fixpoint Engine
Interval
Domain
Gauge
Domain
DBM
Domain
Polyhedra
Domain
IKOS
Verification
Report

Verification Report

- **Safe**: The instruction is **proven free of runtime errors**
- **Error**: The instruction **always produces** a **runtime error**
- **Warning**:
  - The instruction **can produce an error** depending on the input
  - The instruction is **safe** but IKOS could **not prove it** (also called **false positive**)

Example

```
int tab[10];

for (int i = 0; i < 10; i++) {
  tab[i] = i * i;
}
```

- **The analysis discovers program properties:** $0 \leq$ **`i`** ≤ 9

Example

```
int tab[10];

for (int i = 0; i < 10; i++) {
  tab[i] = i * i;
}
```

Access within bounds?

- **The analysis discovers program properties:** $0 \leq$ **`i`** ≤ 9
- **The verification uses the properties discovered:**
  - Array-bound compliance
  - Check that array **tab** has at least 10 elements

IKOS Checks

- Buffer overflow
- Division by zero
- Null pointer dereference
- Assertion prover
- Unaligned pointer
- Uninitialized variable
- Integer overflow (signed, unsigned)
- Invalid bit shift
- Invalid pointer comparison
- Invalid function pointer call
- Dead code
- Double free and Invalid lifetime

IKOS Abstract Domains

Domain	Constraints	Complexity
Interval	$x \in [a, b]$	n
Congruence	$x \in aZ+b$	n
Gauge	$x \in [a*i + b*k + \ldots, a'*i + b'*k + \ldots]$	K*n
Difference Bound Matrices	$x - y \in [a, b]$	n^3
APRON Octagon	$x \pm y \in [a, b]$	n^3
APRON Polka Polyhedra	$a*x + b*y + \ldots + c <= 0$	Exponential
APRON PPL Polyhedra	$a*x + b*y + \ldots + c <= 0$	Exponential
Variable Packing of …	…	n

Live demo

IKOS Installation

- Supported platforms:
  - Mac OS
  - Linux
  - Windows (using *MinGW*)
- **Dependencies** can be installed with a **package manager** (`brew`, `apt-get`, `yum`, ..)
- **Installation instructions** for each platform available in: `doc/install/`
- **Bootstrap script** for **non-admin** installations: downloads and compiles all missing dependencies (*slow*)

IKOS Usage

- Analyze a single file: `ikos file.c`
  - Runs the analysis
  - Prints the results
  - Generates an output database containing the analysis results: `output.db`
- Analyze a whole project:
  - `ikos-scan make`
  - `ikos program.bc`
- Generate a report from an output database: `ikos-report output.db`
- Examine the results in a graphical interface: `ikos-view output.db`

IKOS-SCAN

- Analyze a whole project with: `ikos-scan <command>`
- It compiles all executables to LLVM bitcode: `program.bc`
- It runs IKOS on the LLVM bitcode: `ikos program.bc`
- Works with most build systems: Make, CMake, Autoconf, etc...
- Works by overriding environment variables: `CC`, `CXX`, `LD`

IKOS-SCAN

Live demo

Analyzing a library

- The analysis needs an **entry point** (i.e, `main`)
- **Workaround**: create a small program that uses the library
- Analyze a program with a specific entry point: `ikos file.bc –e=MyMain`

IKOS-VIEW

- **Graphical interface** to examine the analysis results
- Starts a **web server** in the terminal, opens the default **browser**
- `ikos-view output.db`

IKOS-VIEW

Live demo

IKOS Abstract Domains Guidelines

- Start with **fast** but **imprecise** domain
- Go towards **slow** but **precise** domain
- Stop when the analysis is too slow for your use case
- Recommended order:
    - Interval: `-d=interval`
    - Gauge + Interval + Congruence: `-d=gauge-interval-congruence`
    - Variable Packing DBM: `-d=var-pack-dbm`
    - Variable Packing Polyhedra: `-d=var-pack-apron-ppl-polyhedra`

IKOS Assumptions

- The source code is compiled with Clang for the **host architecture**
- Clang defines:
  - The data model (size of types)
  - The memory layout (alignments)
  - The endianness
  - The semantic of floating points
  - Etc...

IKOS Assumptions

- The program is **single-threaded**
- The program does **not** receive **signals** or **interrupts**
- **Unknown extern functions**:
  - Do not update global variables
  - Can write on their pointer arguments
  - Do not call user-defined functions (no callbacks)
- **Assembly code** is treated as a call to an unknown extern function
- **Recursive functions** can update any value in memory

False positives

- **False positive**: invalid warning

- Objective: **low rate** of **false positives**

- Common source of false positives:
  - Unknown library functions
  - “Bad” code patterns
  - Imprecision of the analysis

Modeling library functions

- The analyzer does **not** require the **libraries** used by your program
- Unknown library functions will trigger a warning ("ignored call side effect” in `ikos-view`)
- Modeling library functions can **reduce the number of warnings**
- Write “stubs”: fake implementations of library functions

Modeling library functions

```
#include <ikos/analyzer/intrinsic.h>

char* fgets(char* restrict str,
            int size,
            FILE* restrict stream) {
    __ikos_assert(size >= 0);
    __ikos_forget_mem(stream, sizeof(FILE));
    __ikos_abstract_mem(str, size);
    errno = __ikos_nondet_int();
    return __ikos_nondet_int() ? str : NULL;
}
```

IKOS Annotations

- Annotating your source code can **reduce the number of warnings**
- List of intrinsic functions: `analyzer/include/ikos/analyzer/intrinsic.h`
  - `__ikos_assert(condition)`
  - `__ikos_assume(condition)`
  - `__ikos_nondet_int()`
  - `__ikos_check_mem_access(ptr, size)`
  - `__ikos_assume_mem_size(ptr, size)`
  - `__ikos_forget_mem(ptr, size)`
  - `__ikos_abstract_mem(ptr, size)`
  - `__ikos_print_values(description, var)`

IKOS Annotations

```
ret = talg->parse_algoid_params(buf, param_len, param);
```

IKOS Annotations

```
ret = talg->parse_algoid_params(buf, param_len, param);
```

```
int (*fun)(const u8*, u16, alg_param*) =
    talg->parse_algoid_params;
__ikos_assume(fun == parse_algoid_params_generic ||
              fun == parse_algoid_params_ecdsa_with ||
              fun == parse_algoid_params_ecPublicKey ||
              fun == parse_algoid_params_rsa);
ret = fun(buf, param_len, param);
```

Bad code pattern (1)

```
CommandResult = XXX();
if (CommandResult == TRUE) {
    FilenameState = YYY();
    if (FilenameState == FM_NAME_IS_INVALID) {
        CommandResult = FALSE;
    }
}
if (CommandResult == TRUE) {
    CommandResult = ZZZ();
}
if (CommandResult == TRUE) {
    // ...
}
return CommandResult;
```

Bad code pattern (1)

- Bad readability
- Prone to errors
- Hard for static analyzers
- Please use “early return on errors”

Bad code pattern (1)

```
CommandResult = XXX();
if (!CommandResult) {
    return FALSE;
}
CommandResult = YYY();
if (CommandResult == FM_NAME_IS_INVALID) {
    return FALSE;
}
CommandResult = ZZZ();
if (!CommandResult) {
    return FALSE;
}
// ...
return TRUE;
```

Bad code pattern (2)

- Single global variable containing everything

```
AppData_t g;

typedef struct {
    PipeId_t CmdPipeId;
    uint16 usCmdPipeDepth;
    char cCmdPipeName[OS_MAX_API_NAME];
    int32 ulfd;
    uint32 uiRunStatus;
    // ...
    uint8 lastCmdBchErrorStatus;
} AppData_t;
```

Bad code pattern (2)

- Makes the buffer overflow analysis harder
- Please split it into different global variables

Bad code pattern (3)

- Small integers for loop counters

```
void f(uint16_t n) {
    for (uint16_t i = 0; i < n; i++) {
        // ...
    }
}
```

Bad code pattern (3)

- Small integers for loop counters

```
void f(uint16_t n) {
    for (uint16_t i = 0; i < n; i++) {
        // ...
    }
}
```

- Integer promotion rules of C

```
void f(uint16_t n) {
    for (uint16_t i = 0;
         (unsigned int)i < (unsigned int)n;
         i = (uint16_t)((unsigned int)i + 1)) {
        // ...
    }
}
```

Bad code pattern (3)

- Creates temporary variables in the LLVM bitcode
- Leads to imprecision of the analysis
- Please use `size_t` (or `int`) for loop indexes

Imprecision

- Initialization functions returning an error code

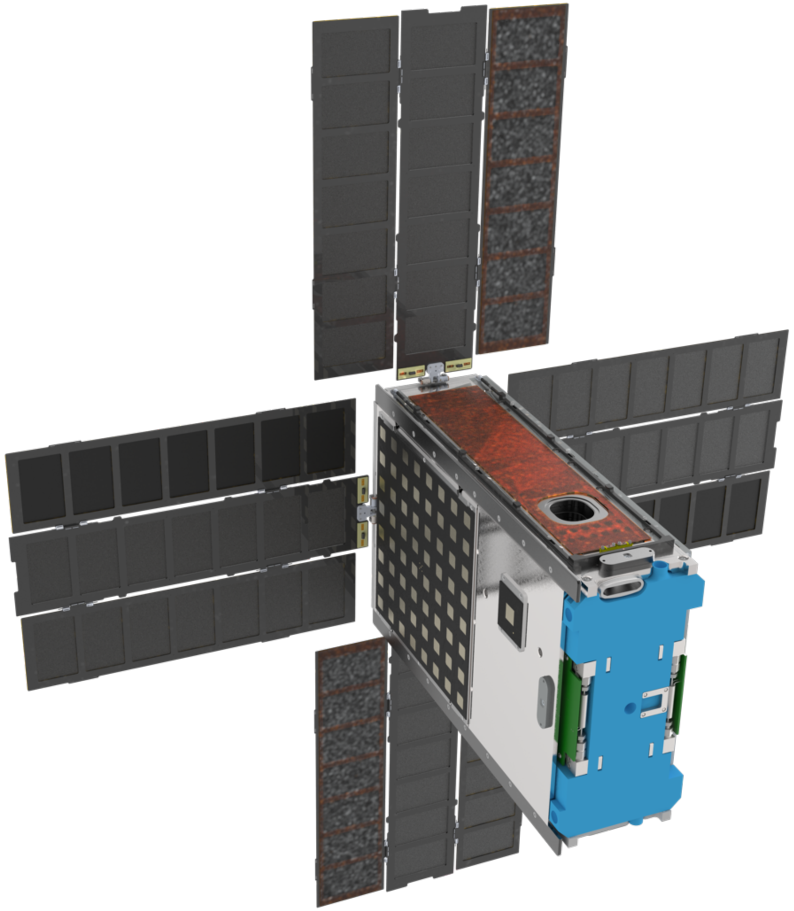
```
int Init(void) {

    int status = Register();
    if (status != SUCCESS) {
        return status;
    }

    status = InitEvent();
    if (status != SUCCESS) {
        return status;
    }

    // ...
```

Imprecision

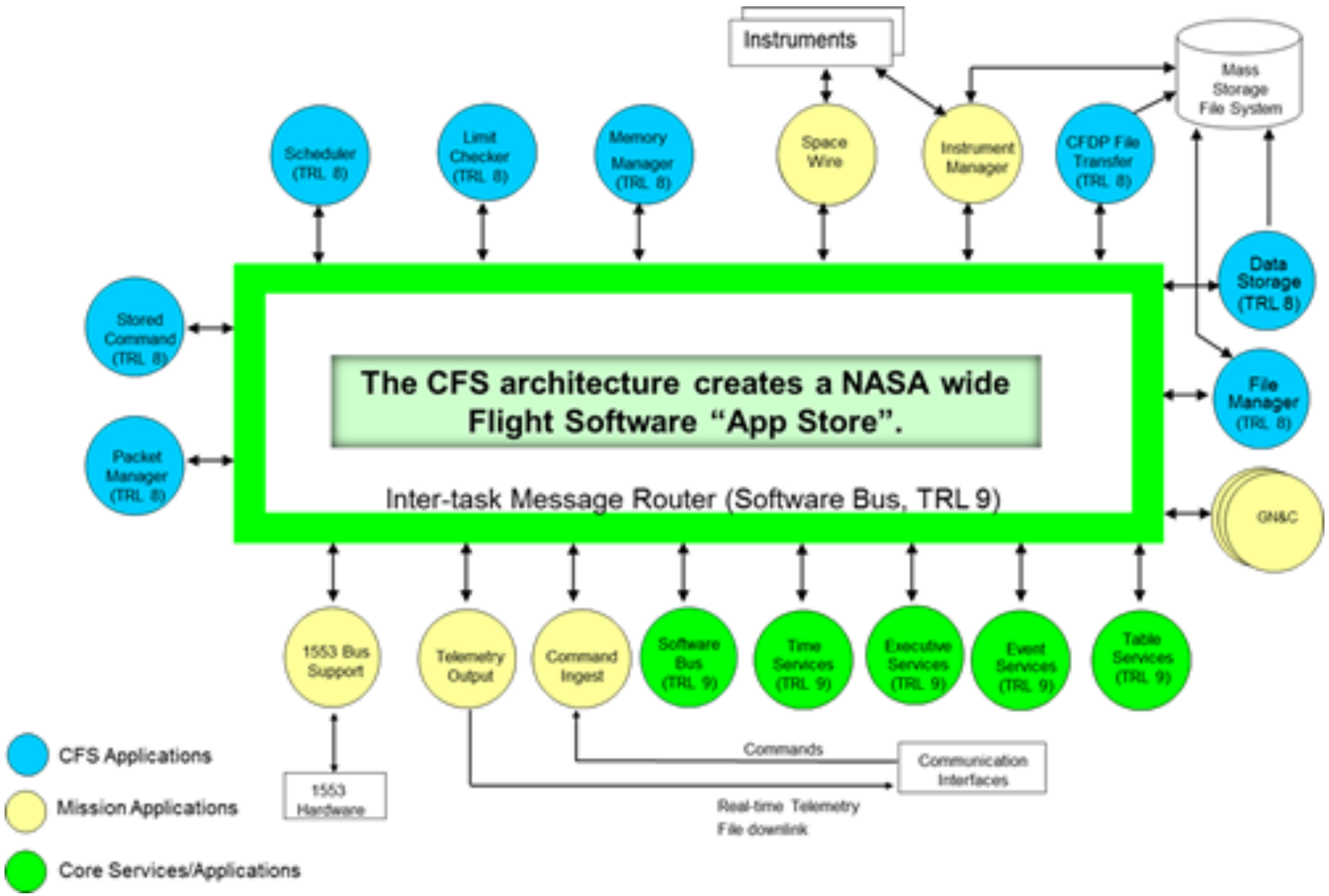
- Imprecision due to the abstract union in the analysis
- Analysis option: `--partitioning=return`

Success Story: BioSentinel

- Space biology mission
- CubeSat spacecraft
- Developed at NASA Ames, in collaboration with JPL, JSC, MSFC
- Flight software built on top of CFS



Success Story: BioSentinel



Success Story: BioSentinel

- Each application was analyzed with IKOS
- The CFE framework was modeled to improve the analysis (~ 1200 LOC)
- Low rate of warnings: 1.31% in average
- Found ~ 17 real bugs

Success Story: BioSentinel

Application	Abstract Domain	Time	Errors	Warnings	Warnings%	Checks
adio	var-pack-dbm	1 min 6.92 sec	0	1	0.07%	1334
brdio	var-pack-dbm	8.02 sec	0	8	0.97%	818
ci	var-pack-dbm	19.98 sec	0	6	0.65%	923
comio	var-pack-dbm	1 min 4.83 sec	0	4	0.26%	1494
epsio	var-pack-dbm	30.64 sec	0	5	0.42%	1181
letio	var-pack-dbm	24.33 sec	0	18	1.64%	1095
ms	interval	0.16 sec	0	0	0%	444
saio	var-pack-dbm	22.35 sec	0	8	0.64%	1246
sensio	var-pack-dbm	4.67 sec	0	79	9.56%	826
spe	interval	0.16 sec	0	0	0%	445
thrio	var-pack-dbm	19.33 sec	0	4	0.38%	1043
to	var-pack-dbm	2 min 18.32 sec	0	33	1.98%	1666
xactio	var-pack-dbm	22.18 sec	0	6	0.51%	1165

BioSentinel Bug (1)

```
warning: Possible buffer overflow, pointer '&FrameBuffer[0]' with offset 0
bytes points to global variable 'FrameBuffer' of size 4096 bytes
```

```
ssize_t numbytes = read(fd, &FrameBufferRaw[0], 4096);

if (numbytes < 0) {
    return ERROR;
}

// ...

numbytes -= 9; // Integer overflow

memcpy(/*dst*/ FrameBuffer, /*src*/ FrameBufferRaw, /*size*/ numbytes);
^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

BioSentinel Bug (2)

```
warning: Possible buffer overflow, pointer '&cmd[n + 2]' accesses 1 bytes at
offset between 8 and 16 bytes of local variable 'cmd' of size 16 bytes
```

```
uint8_t  cmd[16];
uint8_t  n;
// ...
switch(cmd_request) {
    case CMD_OPEN:
        n = CMD_OUT + CMD_OPEN; // 6 + 8 = 14
        break;
    // ...
}
// ...
cmd[n + 2] = 0; // 14 + 2 = 16
```

Guidelines

- Use a lightweight static analyzer first: cppcheck, clang-tidy, pvs-studio, etc.
- Use ikos-scan to generate the llvm bitcode (*.bc):* `ikos-scan make`
- Use ikos on the llvm bitcode (.bc): `ikos program.bc`
- Try different abstract domains: `ikos –d=var-pack-dbm program.bc`
- Use ikos-view to examine the results: `ikos-view output.db`
- (Optional) Model key library functions
- (Optional) Annotate the code
- (Optional) Avoid “bad” patterns
- (Optional) Add ikos in your continuous build system

IKOS at a glance

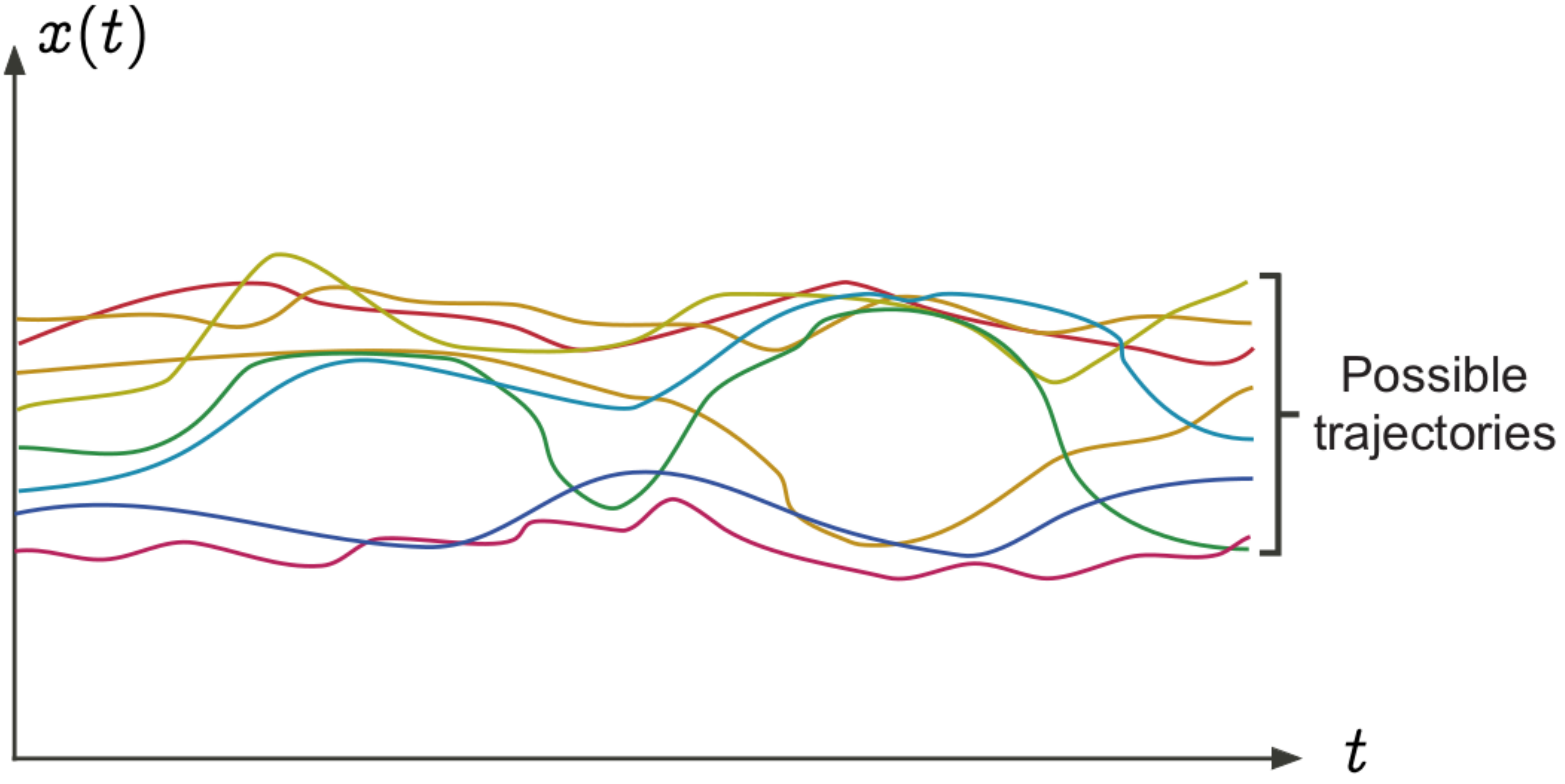
- IKOS is a **static analyzer** for **C/C++** targeting **safety critical** software
- IKOS is **open source**: https://github.com/NASA-SW-VnV/ikos
- Contact: ikos@lists.nasa.gov

Table of contents

Abstract Interpretation

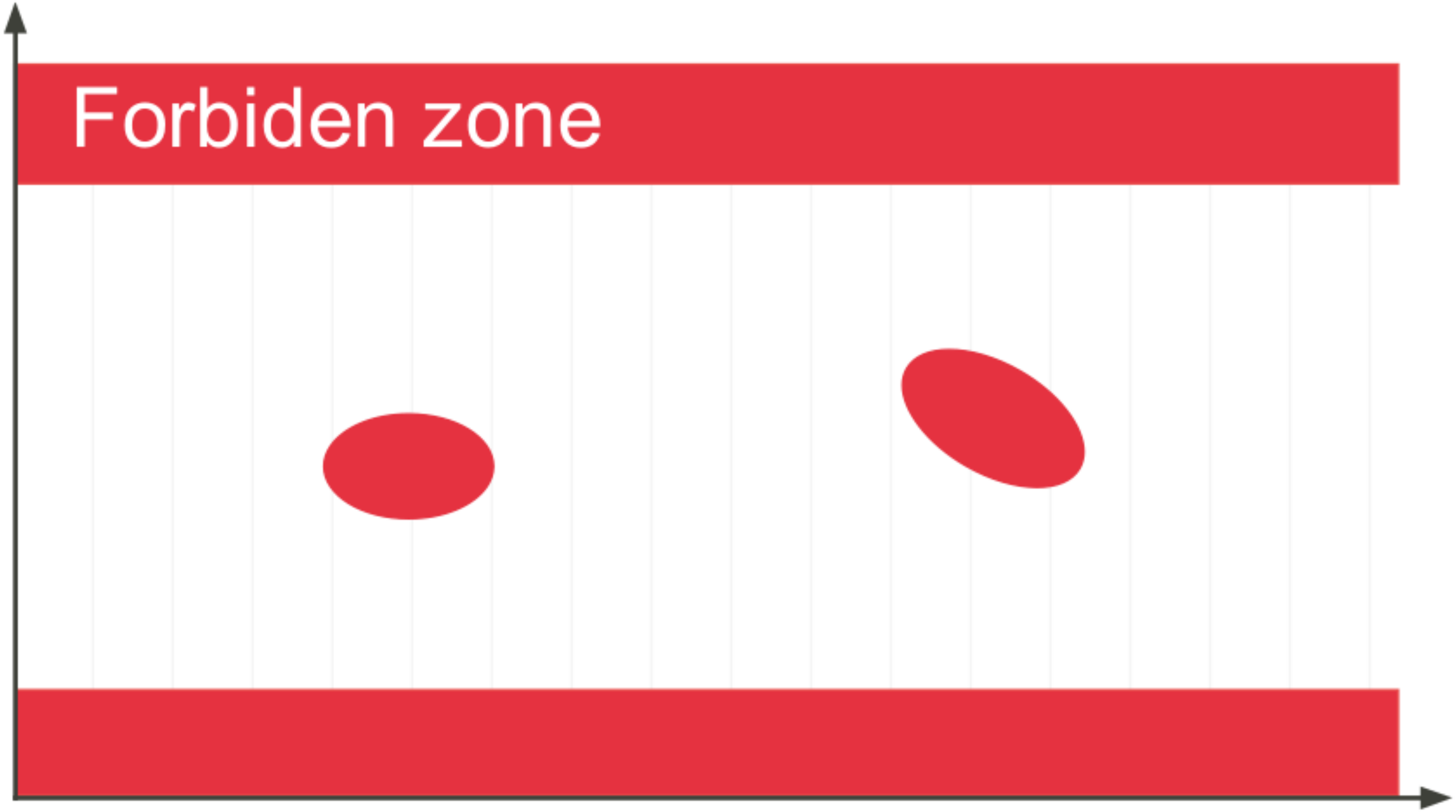
- Mathematical framework
- **Approximation** of the reachable **states** of a program
- **Fully automated**: No user interaction
- **Sound:** Cannot miss a bug
- Formalized by Patrick and Radhia Cousot in the late 1970s

Overview



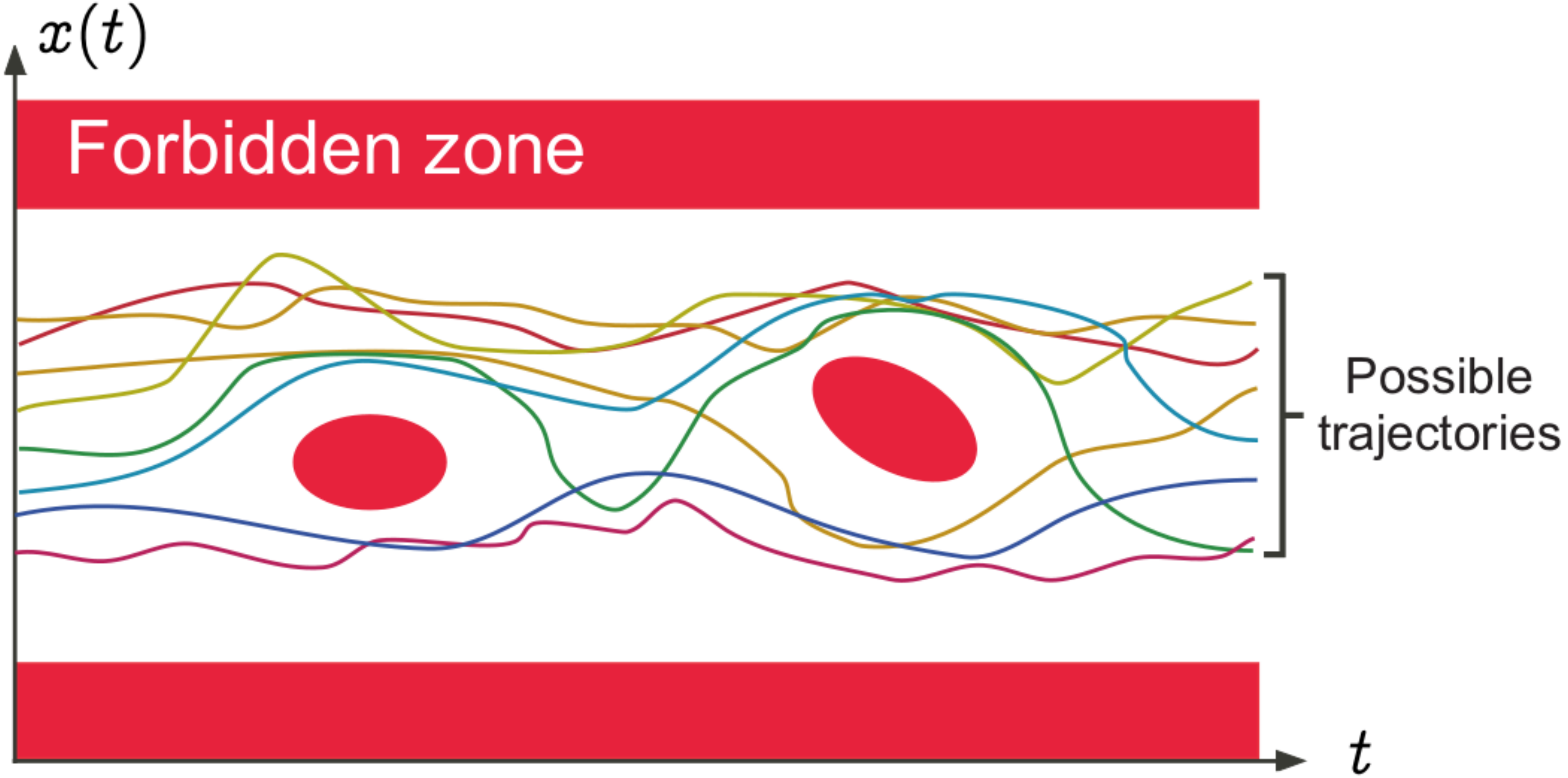
Semantics(P)

Overview



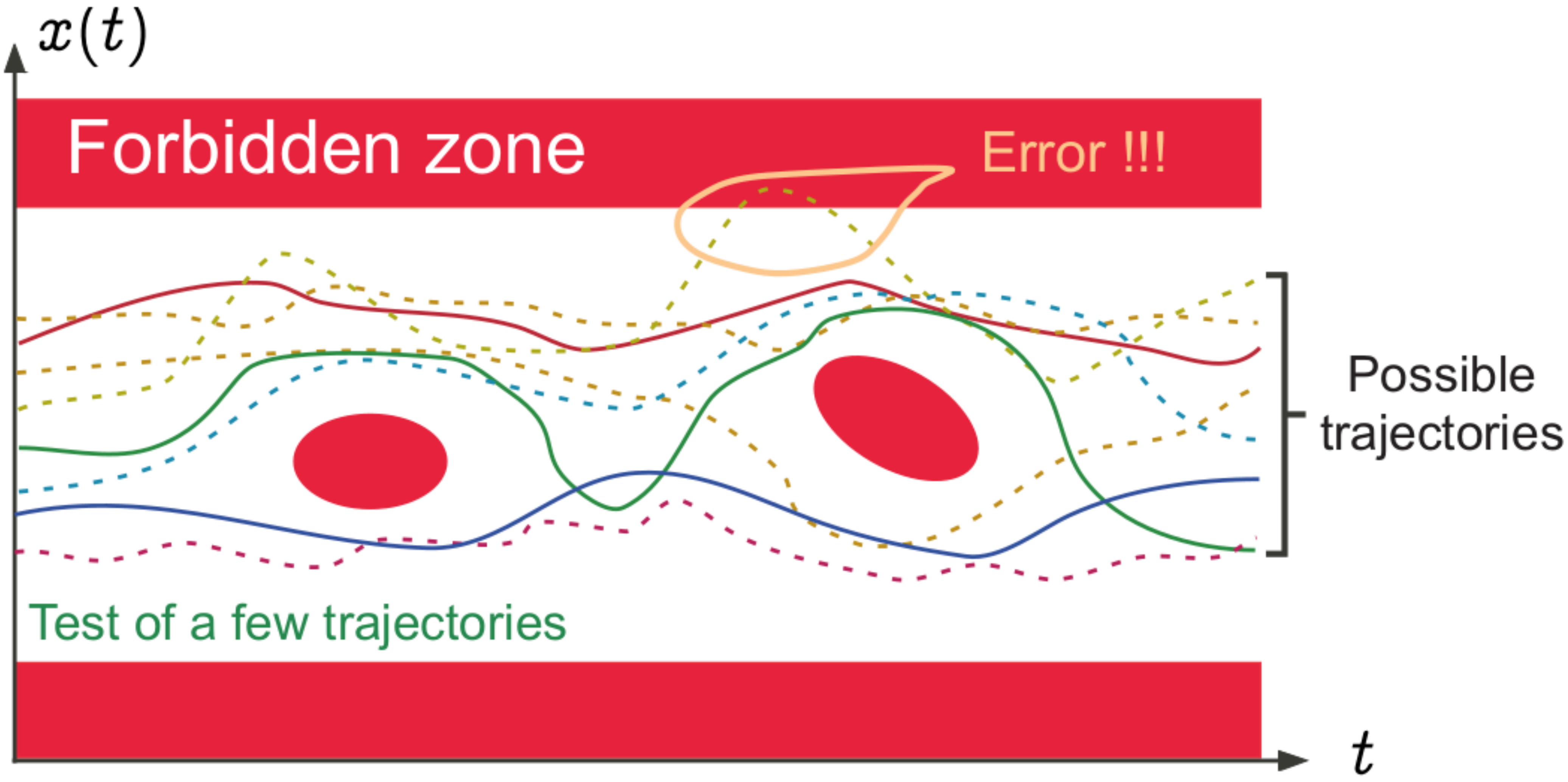
Specification(P)

Overview



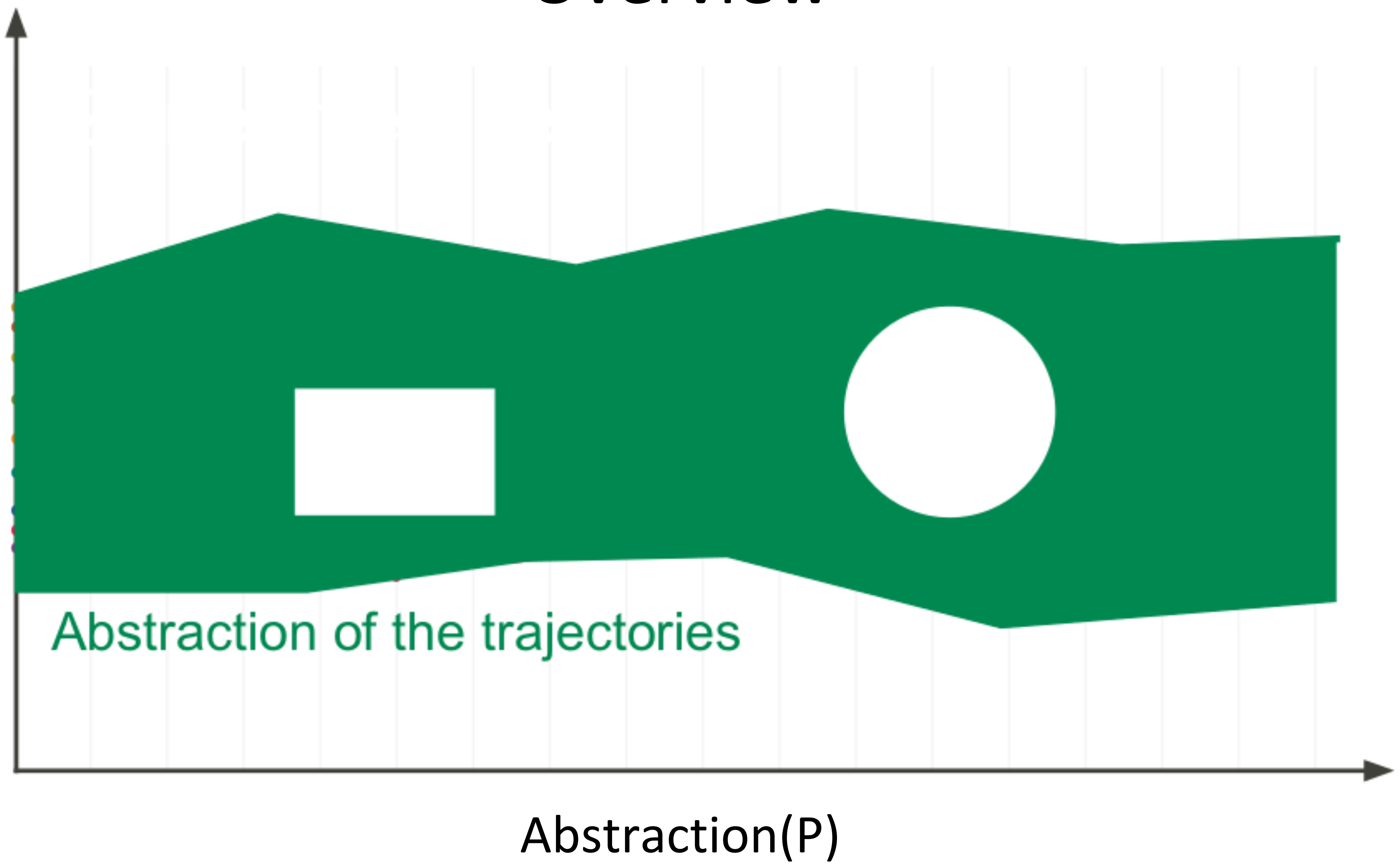
Semantics(P) ⊆ Specification(P)

Overview



Using Testing

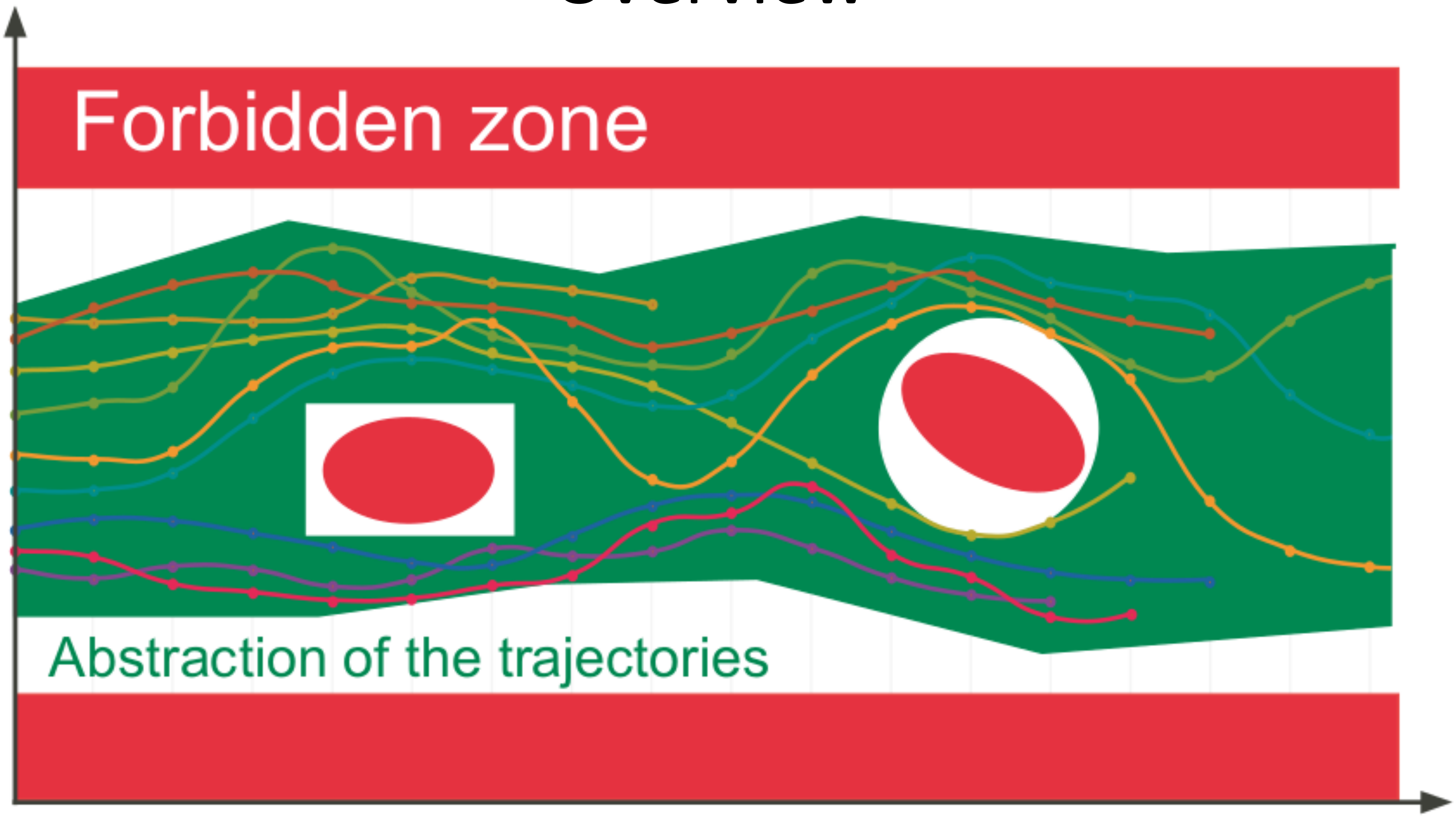
Overview



Overview

Abstraction(P) ⊆ Specification(P)

Overview



Semantics(P) ⊆ Abstraction(P) ⊆ Specification(P) ✓

Toy language - Syntax

```
stmt ::= | v = expr;
         | stmt stmt
         | if (expr > 0) { stmt }
           else { stmt }
         | while (expr > 0) { stmt }

expr ::= | v ∈ 𝕍
         | n ∈ ℤ
         | rand(a, b)
         | expr + expr
         | expr – expr
         | expr * expr
         | expr / expr
```

$\mathbb{V}$ set of variables
$\mathbb{Z}$ set of integers

`rand(a, b)` represents an integer between **a** and **b** (simulate an input)

Toy language - Example

```
x = rand(0, 12);
y = 42;
while (x > 0) {
    x = x - 2;
    y = y + 4;
}
```

An execution:
(values at the beginning of the loop)

```
x |  7   5   3   1  -1
y | 42  46  50  54  58
```

Toy language - Example

```
x = rand(0, 12);
y = 42;
while (x > 0) {
    x = x - 2;
    y = y + 4;
}
```

An execution:
(values at the beginning of the loop)

```
x |  7   5   3   1  -1
y | 42  46  50  54  58
```

Notes:

- A very **simple** language, no functions, no arrays, ...
- But it is an **imperative language** like C
- It is actually a **subset of C**
- It can compute everything – **Turing complete**

Toy language - Semantic

- We need to define the semantic of our language

- Also called **Formal Specification**

- **Collective** semantic:

  Mathematical definition of the reachable states of a given program

Control flow graph

A ***control flow graph*** is a triplet (*L, O, A*) with a set of program points *L*, an entry point $O \in L$ and a set of edges $A \subseteq L \times command \times L$ with:

```
command ::= | v = expr
            | expr > 0
```

Control flow graph

A ***control flow graph*** is a triplet (*L, O, A*) with a set of program points *L*, an entry point $O \in L$ and a set of edges $A \subseteq L \times command \times L$ with:

```
command ::= | v = expr
            | expr > 0
```

$_0\mathsf{x} = \mathbf{rand}(0, 12)\,;_1\mathsf{y} = 42\,;$

$\mathbf{while}\ {}_2(\mathsf{x} > 0)\ \{$

$\quad {}_3\mathsf{x} = \mathsf{x} - 2\,;$

$\quad {}_4\mathsf{y} = \mathsf{y} + 4\,;$

$\}_5$

Control flow graph

A ***control flow graph*** is a triplet (*L, O, A*) with a set of program points *L*, an entry point *O* ∈ *L* and a set of edges *A* ⊆ *L* x *command* x *L* with:

```
command ::= | v = expr
            | expr > 0
```

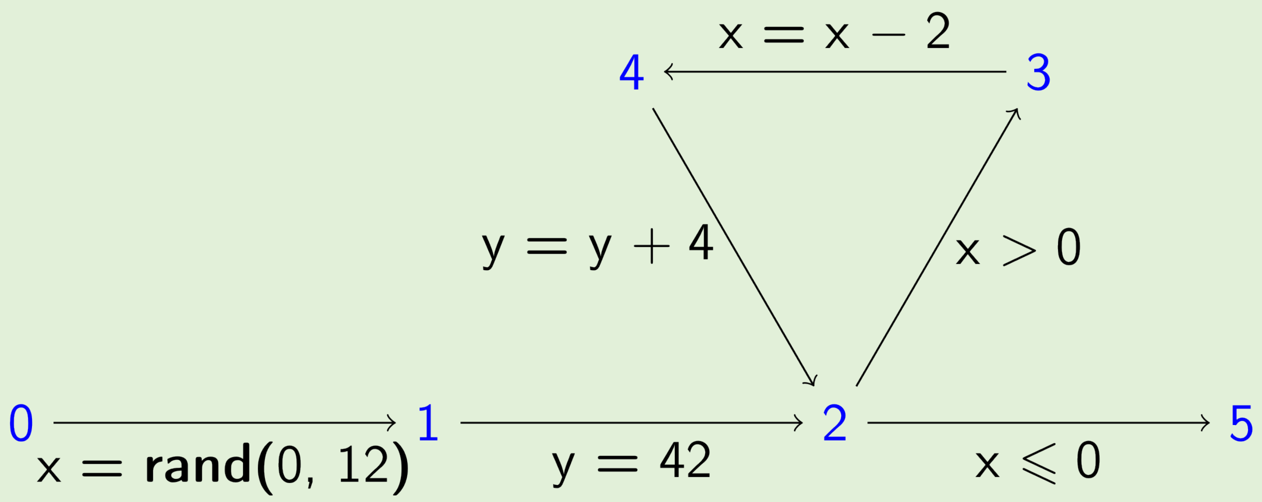
$_0\mathsf{x} = \textbf{rand}(0, 12)\,;_1\mathsf{y} = 42\,;$

$\textbf{while}\ {}_2(\mathsf{x} > 0)\ \{$

$\quad {}_3\mathsf{x} = \mathsf{x} - 2\,;$

$\quad {}_4\mathsf{y} = \mathsf{y} + 4\,;$

$\}_5$



Semantic of Expressions

Semantic of expressions: $[\![e]\!]_{\mathrm{E}} : (\mathbb{V} \to \mathbb{Z}) \to \mathcal{P}(\mathbb{Z})$

$$
\begin{aligned}
[\![v]\!]_{\mathrm{E}}(\rho) &= \{\rho(v)\} \\
[\![n]\!]_{\mathrm{E}}(\rho) &= \{n\} \\
[\![\mathbf{rand}(n_1, n_2)]\!]_{\mathrm{E}}(\rho) &= \{n \in \mathbb{Z} \mid n_1 \leqslant n \leqslant n_2\} \\
[\![e_1 + e_2]\!]_{\mathrm{E}}(\rho) &= \{n_1 + n_2 \mid n_1 \in [\![e_1]\!]_{\mathrm{E}}(\rho) \wedge n_2 \in [\![e_2]\!]_{\mathrm{E}}(\rho)\} \\
\ldots
\end{aligned}
$$

Semantic of Expressions

Semantic of expressions: $[\![e]\!]_{\mathrm{E}} : (\mathbb{V} \to \mathbb{Z}) \to \mathcal{P}(\mathbb{Z})$

$$
\begin{aligned}
[\![v]\!]_{\mathrm{E}}(\rho) &= \{\rho(v)\} \\
[\![n]\!]_{\mathrm{E}}(\rho) &= \{n\} \\
[\![\textbf{rand}(n_1, n_2)]\!]_{\mathrm{E}}(\rho) &= \{n \in \mathbb{Z} \mid n_1 \leqslant n \leqslant n_2\} \\
[\![e_1 + e_2]\!]_{\mathrm{E}}(\rho) &= \{n_1 + n_2 \mid n_1 \in [\![e_1]\!]_{\mathrm{E}}(\rho) \wedge n_2 \in [\![e_2]\!]_{\mathrm{E}}(\rho)\} \\
\ldots &
\end{aligned}
$$

An environment $\boldsymbol{\rho}$ is a function $\mathbb{V} \to \mathbb{Z}$ that associates a value to each variable

Errors

Notes on errors:

We can reach 2 kind of errors during the execution:

Errors

Notes on errors:

We can reach 2 kind of errors during the execution:

- `rand(n1, n2)` with $n_1 > n_2$:

$$[\![\mathbf{rand}(n_1, n_2)]\!]_{\mathrm{E}} = \{x \in \mathbb{Z} \mid n_1 \leqslant x \leqslant n_2\} = \emptyset \,;$$

Errors

Notes on errors:

We can reach 2 kind of errors during the execution:

- `rand(n₁, n₂)` with $n_1 > n_2$:

  $[\![\textbf{rand}(n_1, n_2)]\!]_{\mathrm{E}} = \{x \in \mathbb{Z} \mid n_1 \leqslant x \leqslant n_2\} = \emptyset\,;$

- Division by zero :

  $[\![e/0]\!]_{\mathrm{E}} = \emptyset.$

Errors

Notes on errors:

We can reach 2 kind of errors during the execution:

- `rand(n₁, n₂)` with $n_1 > n_2$:

  $[\![\textbf{rand}(n_1, n_2)]\!]_{\mathrm{E}} = \{x \in \mathbb{Z} \mid n_1 \leqslant x \leqslant n_2\} = \emptyset\,;$

- Division by zero :

  $[\![e/0]\!]_{\mathrm{E}} = \emptyset.$

We assume the program aborts on errors.

Semantic of commands

Semantic of commands: $[\![c]\!]_{\mathrm{C}} : \mathcal{P}(\mathbb{V} \to \mathbb{Z}) \to \mathcal{P}(\mathbb{V} \to \mathbb{Z})$

$$[\![v = e]\!]_{\mathrm{C}}(R) = \{\rho[v \mapsto n] \mid \rho \in R, n \in [\![e]\!]_{\mathrm{E}}(\rho)\}$$
$$[\![e > 0]\!]_{\mathrm{C}}(R) = \{\rho \mid \rho \in R, \exists n \in [\![e]\!]_{\mathrm{E}}(\rho), n > 0\}$$

Semantic of commands

Semantic of commands: $[\![c]\!]_{\mathrm{C}} : \mathcal{P}(\mathbb{V} \to \mathbb{Z}) \to \mathcal{P}(\mathbb{V} \to \mathbb{Z})$

$$[\![v = e]\!]_{\mathrm{C}}(R) = \{\rho[v \mapsto n] \mid \rho \in R, n \in [\![e]\!]_{\mathrm{E}}(\rho)\}$$
$$[\![e > 0]\!]_{\mathrm{C}}(R) = \{\rho \mid \rho \in R, \exists n \in [\![e]\!]_{\mathrm{E}}(\rho), n > 0\}$$

Note that `e ≤ 0` can be rewritten as `1 – e > 0` (syntactic sugar)

Semantic of programs

Semantic of programs: $[\![(L, A)]\!] : L \to \mathcal{P}(\mathbb{V} \to \mathbb{Z})$

For each program point, it gives the set of environments

Semantic of programs

Semantic of programs: $[\![(L, A)]\!] : L \to \mathcal{P}(\mathbb{V} \to \mathbb{Z})$

For each program point, it gives the set of environments

It is the smallest solution (in term of inclusion) of the following system:

$$\begin{cases} R_0 = \mathbb{V} \to \mathbb{Z} & \\ R_{l'} = \bigcup_{(l,c,l') \in A} [\![c]\!]_{\mathrm{C}}(R_l) & l' \neq 0 \end{cases}$$

Semantic of programs

Semantic of programs: $[\![(L, A)]\!] : L \to \mathcal{P}(\mathbb{V} \to \mathbb{Z})$

For each program point, it gives the set of environments

It is the smallest solution (in term of inclusion) of the following system:

$$\begin{cases} R_0 = \mathbb{V} \to \mathbb{Z} \\ R_{l'} = \displaystyle\bigcup_{(l,c,l') \in A} [\![c]\!]_{\mathrm{C}}\,(R_l) \qquad\qquad l' \neq 0 \end{cases}$$

The theorem of Knaster-Tarski tells us that the solution always exists!

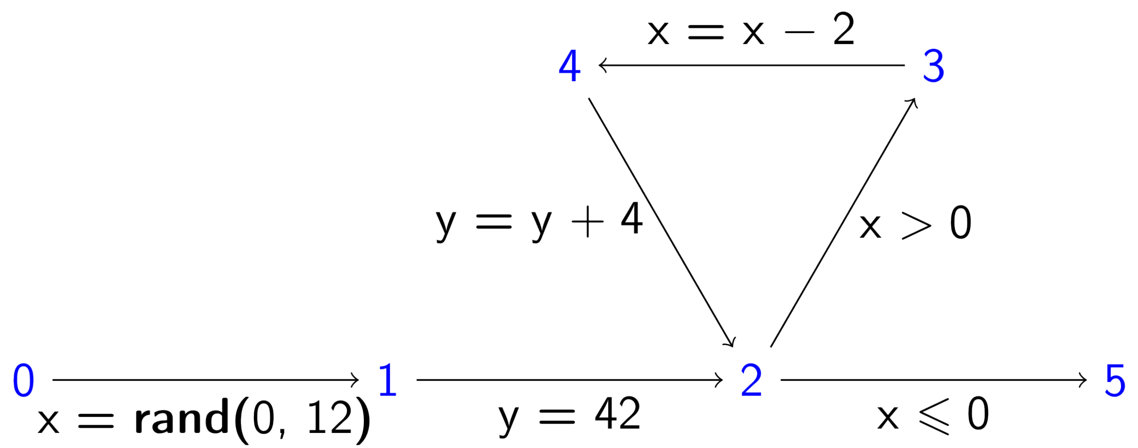
${}_0\mathsf{x} = \mathbf{rand}(0,\ 12)\ ;{}_1\mathsf{y} = 42\ ;$

while ${}_2(\mathsf{x} > 0)$ **{**

  ${}_3\mathsf{x} = \mathsf{x} - 2\ ;$

  ${}_4\mathsf{y} = \mathsf{y} + 4\ ;$

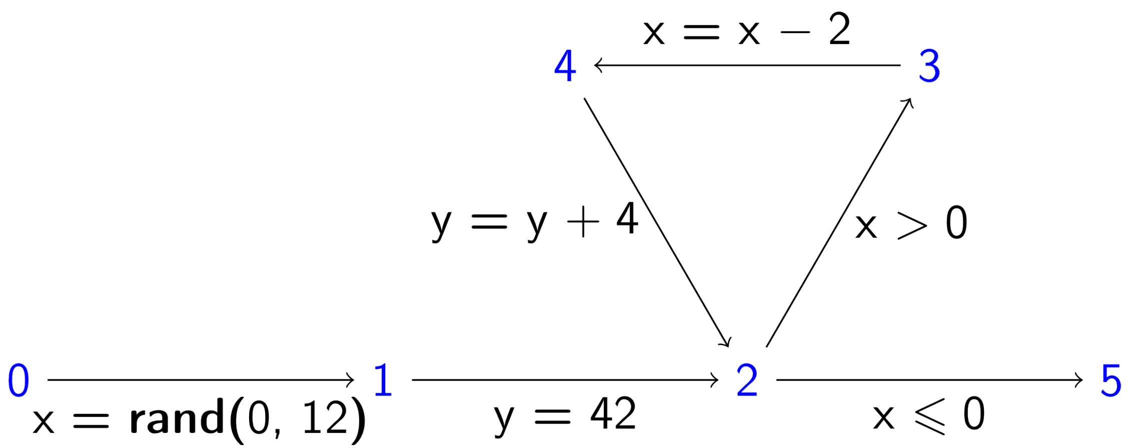
}$_5$



Equations:

$$R_0 = \{\, x \in \mathbb{Z}, y \in \mathbb{Z} \,\}$$

$_0$x = **rand**(0, 12) ;$_1$y = 42 ;
while $_2$(x > 0) **{**
    $_3$x = x − 2 ;
    $_4$y = y + 4 ;
}$_5$



Equations:

$$R_0 = \{\, x \in \mathbb{Z}, y \in \mathbb{Z} \,\}$$
$$R_1 = \{\, x \in [\![0, 12]\!], y \in \mathbb{Z} \,\}$$

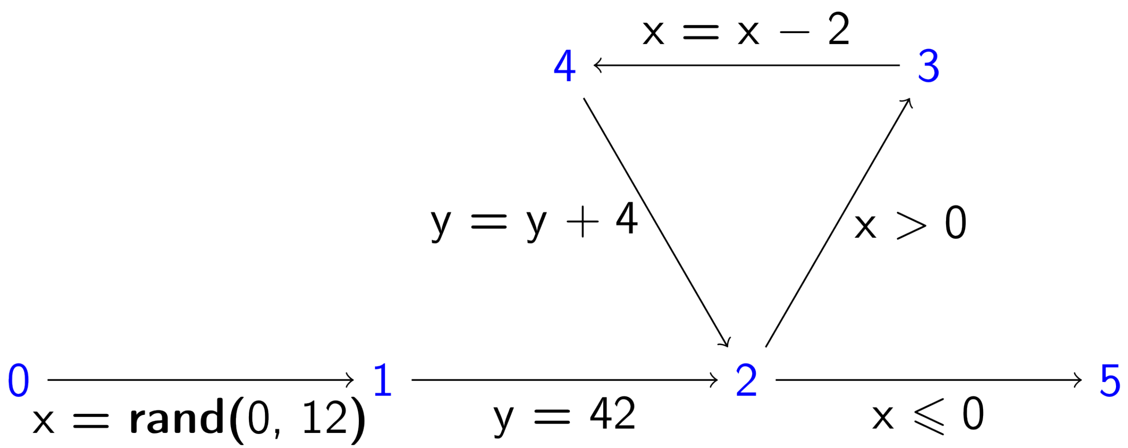
$_0$x = **rand**(0, 12) ;$_1$y = 42 ;

while $_2$(x > 0) **{**

  $_3$x = x − 2 ;

  $_4$y = y + 4 ;

}$_5$



Equations:

$$
\begin{aligned}
R_0 &= \{\, x \in \mathbb{Z}, y \in \mathbb{Z} \,\} \\
R_1 &= \{\, x \in [\![0, 12]\!], y \in \mathbb{Z} \,\} \\
R_2 &= R_1[y \mapsto 42] \cup R_4[y \mapsto y + 4]
\end{aligned}
$$

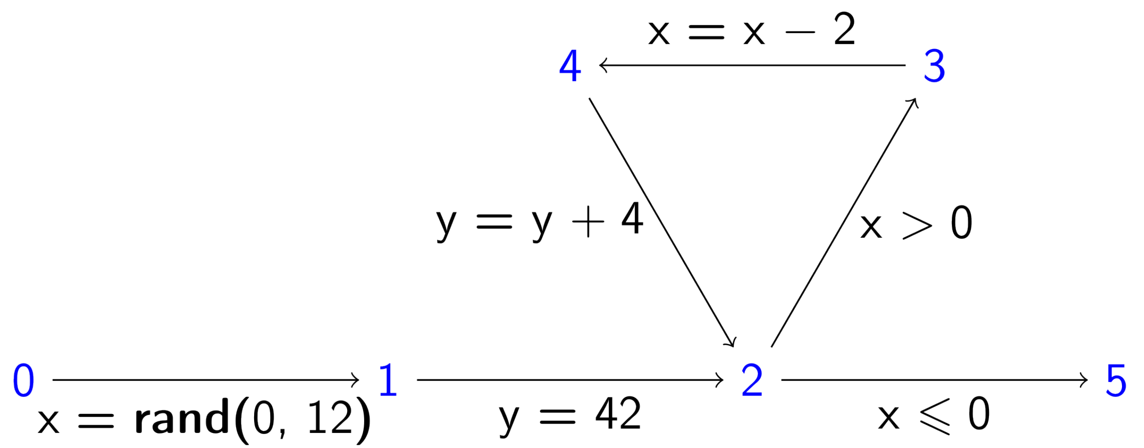
$_0\mathsf{x} = \mathbf{rand}(0,\ 12)\ ;_1\mathsf{y} = 42\ ;$

$\mathbf{while}\ _2(\mathsf{x} > 0)\ \boldsymbol{\{}$

$\quad {}_3\mathsf{x} = \mathsf{x} - 2\ ;$

$\quad {}_4\mathsf{y} = \mathsf{y} + 4\ ;$

$\boldsymbol{\}}_5$



Equations:

$$R_0 = \{\, x \in \mathbb{Z}, y \in \mathbb{Z} \,\}$$
$$R_1 = \{\, x \in [\![0, 12]\!], y \in \mathbb{Z} \,\}$$
$$R_2 = R_1[y \mapsto 42] \cup R_4[y \mapsto y + 4]$$

$$R_3 = R_2 \cap \{\, x > 0, y \in \mathbb{Z} \,\}$$

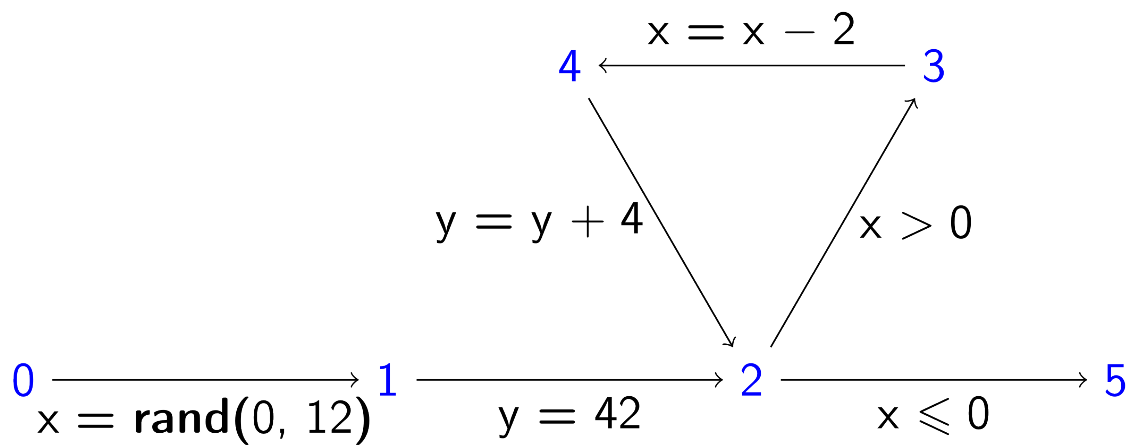
$_0\mathsf{x} = \textbf{rand}(0,\ 12)\ ;_1\mathsf{y} = 42\ ;$

$\textbf{while}\ _2(\mathsf{x} > 0)\ \{$

$\qquad _3\mathsf{x} = \mathsf{x} - 2\ ;$

$\qquad _4\mathsf{y} = \mathsf{y} + 4\ ;$

$\}_5$



Equations:

$$
\begin{aligned}
R_0 &= \{\, x \in \mathbb{Z}, y \in \mathbb{Z} \,\} \\
R_1 &= \{\, x \in [\![0, 12]\!], y \in \mathbb{Z} \,\} \\
R_2 &= R_1[y \mapsto 42] \cup R_4[y \mapsto y + 4] \\
\\
R_3 &= R_2 \cap \{\, x > 0, y \in \mathbb{Z} \,\} \\
\\
R_4 &= R_3[x \mapsto x - 2]
\end{aligned}
$$

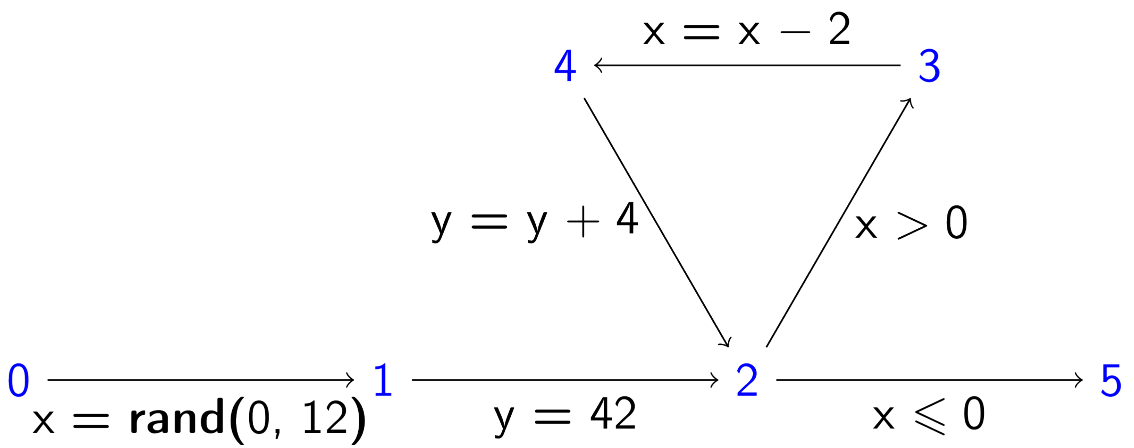
$_0$x = **rand**(0, 12) ;$_1$y = 42 ;

while $_2$(x > 0) **{**

    $_3$x = x − 2 ;

    $_4$y = y + 4 ;

}$_5$



Equations:

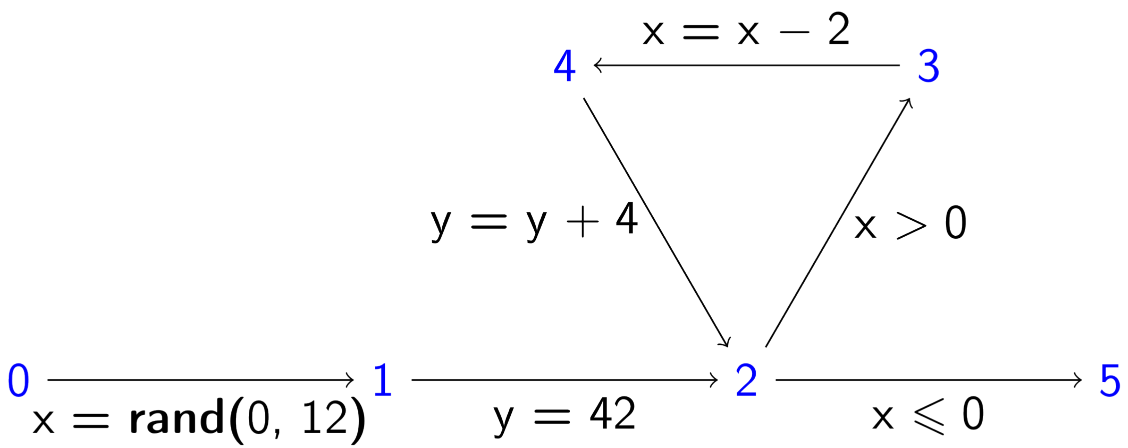
$$R_0 = \{\, x \in \mathbb{Z}, y \in \mathbb{Z} \,\}$$
$$R_1 = \{\, x \in [\![0, 12]\!], y \in \mathbb{Z} \,\}$$
$$R_2 = R_1[y \mapsto 42] \cup R_4[y \mapsto y + 4]$$

$$R_3 = R_2 \cap \{\, x > 0, y \in \mathbb{Z} \,\}$$

$$R_4 = R_3[x \mapsto x - 2]$$

$$R_5 = R_2 \cap \{\, x \leqslant 0, y \in \mathbb{Z} \,\}$$

$${}_0\mathtt{x} = \mathbf{rand}(0,\ 12)\ ;{}_1\mathtt{y} = 42\ ;$$
$$\mathbf{while}\ {}_2(\mathtt{x} > 0)\ \{$$
$$\quad {}_3\mathtt{x} = \mathtt{x} - 2\ ;$$
$$\quad {}_4\mathtt{y} = \mathtt{y} + 4\ ;$$
$$\}_5$$



Equations: Smallest Solution:

$$\begin{aligned}
R_0 &= \{\, x \in \mathbb{Z}, y \in \mathbb{Z} \,\} && = \{\, x \in \mathbb{Z}, y \in \mathbb{Z} \,\} \\
R_1 &= \{\, x \in [\![0, 12]\!], y \in \mathbb{Z} \,\} && = \{\, x \in [\![0, 12]\!], y \in \mathbb{Z} \,\} \\
R_2 &= R_1[y \mapsto 42] \cup R_4[y \mapsto y + 4] && = \{x \in [\![-1, 12]\!], y \in [\![42, 66]\!] \cap 4\mathbb{Z} + 2 \\
& && \quad \mid 2x + y \in [\![42, 66]\!]\} \\
R_3 &= R_2 \cap \{\, x > 0, y \in \mathbb{Z} \,\} && = \{x \in [\![1, 12]\!], y \in [\![42, 66]\!] \cap 4\mathbb{Z} + 2 \\
& && \quad \mid 2x + y \in [\![42, 66]\!]\} \\
R_4 &= R_3[x \mapsto x - 2] && = \{x \in [\![-1, 10]\!], y \in [\![42, 66]\!] \cap 4\mathbb{Z} + 2 \\
& && \quad \mid 2x + y \in [\![38, 62]\!]\} \\
R_5 &= R_2 \cap \{\, x \leqslant 0, y \in \mathbb{Z} \,\} && = \{x \in [\![-1, 0]\!], y \in [\![42, 66]\!] \cap 4\mathbb{Z} + 2 \\
& && \quad \mid 2x + y \in [\![42, 66]\!]\}
\end{aligned}$$

Order and Supremum

An ***order*** $\sqsubseteq$ is a binary relation:

- Reflexive : $\forall x, x \sqsubseteq x$
- Transitive : $\forall x, y, z, (x \sqsubseteq y \wedge y \sqsubseteq z) \Rightarrow x \sqsubseteq z$
- Antisymmetric : $\forall x, y, (x \sqsubseteq y \wedge y \sqsubseteq y) \Rightarrow x = y$

Order and Supremum

An ***order*** $\sqsubseteq$ is a binary relation:

- Reflexive : $\forall x, x \sqsubseteq x$
- Transitive : $\forall x, y, z, (x \sqsubseteq y \wedge y \sqsubseteq z) \Rightarrow x \sqsubseteq z$
- Antisymmetric : $\forall x, y, (x \sqsubseteq y \wedge y \sqsubseteq y) \Rightarrow x = y$

The ***supremum*** $\bigsqcup : \mathcal{P}(S) \to S$ associates to each subset S' of S its smallest upper bound:

- $\forall x \in S', x \sqsubseteq \bigsqcup S'$
- $\forall y \in S, (\forall x \in S', x \sqsubseteq y) \Rightarrow \bigsqcup S' \sqsubseteq y$

Complete Lattice

A set ***S*** equipped with an order $\sqsubseteq$ is a ***complete lattice*** if it has a supremum $\bigsqcup : \mathcal{P}(S) \to S$

Complete Lattice

A set ***S*** equipped with an order $\sqsubseteq$ is a ***complete lattice*** if it has a supremum $\bigsqcup : \mathcal{P}(S) \to S$

A ***complete lattice*** automatically has:

- An ***infimum*** (greatest lower bound):

$$\bigsqcap S' = \bigsqcup \{x \mid \forall y \in S', x \sqsubseteq y\}$$

- A smallest element (***bottom***): $\bot = \bigsqcup \emptyset = \bigsqcap S$
- A greatest element (***top***): $\top = \bigsqcup S = \bigsqcap \emptyset$

Example

$\mathbb{Z}$ is not a complete lattice : $\bigsqcup \mathbb{Z}$ does not exist

Example

$\mathbb{Z}$ is not a complete lattice : $\bigsqcup \mathbb{Z}$ does not exist

$\bar{\mathbb{Z}} = \mathbb{Z} \cup \{-\infty, +\infty\}$ is a complete lattice

Example

$\mathbb{Z}$ is not a complete lattice : $\bigsqcup \mathbb{Z}$ does not exist

$\bar{\mathbb{Z}} = \mathbb{Z} \cup \{-\infty, +\infty\}$ is a complete lattice

- What is $\top$?
- What is $\bot$?

Example

$$\begin{tikzcd} & \{0,1\} \arrow[dl, dash] \arrow[dr, dash] & \\ \{0\} & & \{1\} \end{tikzcd}$$

Is it a lattice?

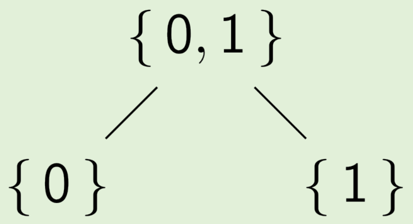
Example

$$\begin{tikzcd}[arrows=dash] & \{0,1\} \arrow[dl] \arrow[dr] & \\ \{0\} & & \{1\} \end{tikzcd}$$

Is it a lattice?

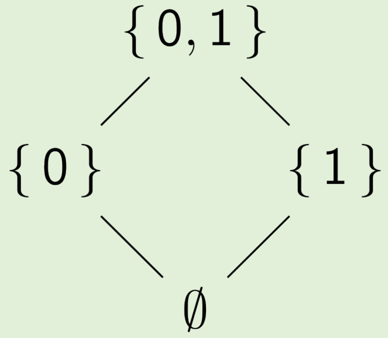
No: $\bigsqcup \emptyset$ does not exist

Example



Is it a lattice?

No: $\bigsqcup \emptyset$ does not exist



This is a lattice

Monotonic Function

A function ***f*** on a complete lattice is ***monotonic*** if and only if:

$$\forall x, y \in S, \quad x \sqsubseteq y \Rightarrow f(x) \sqsubseteq f(y)$$

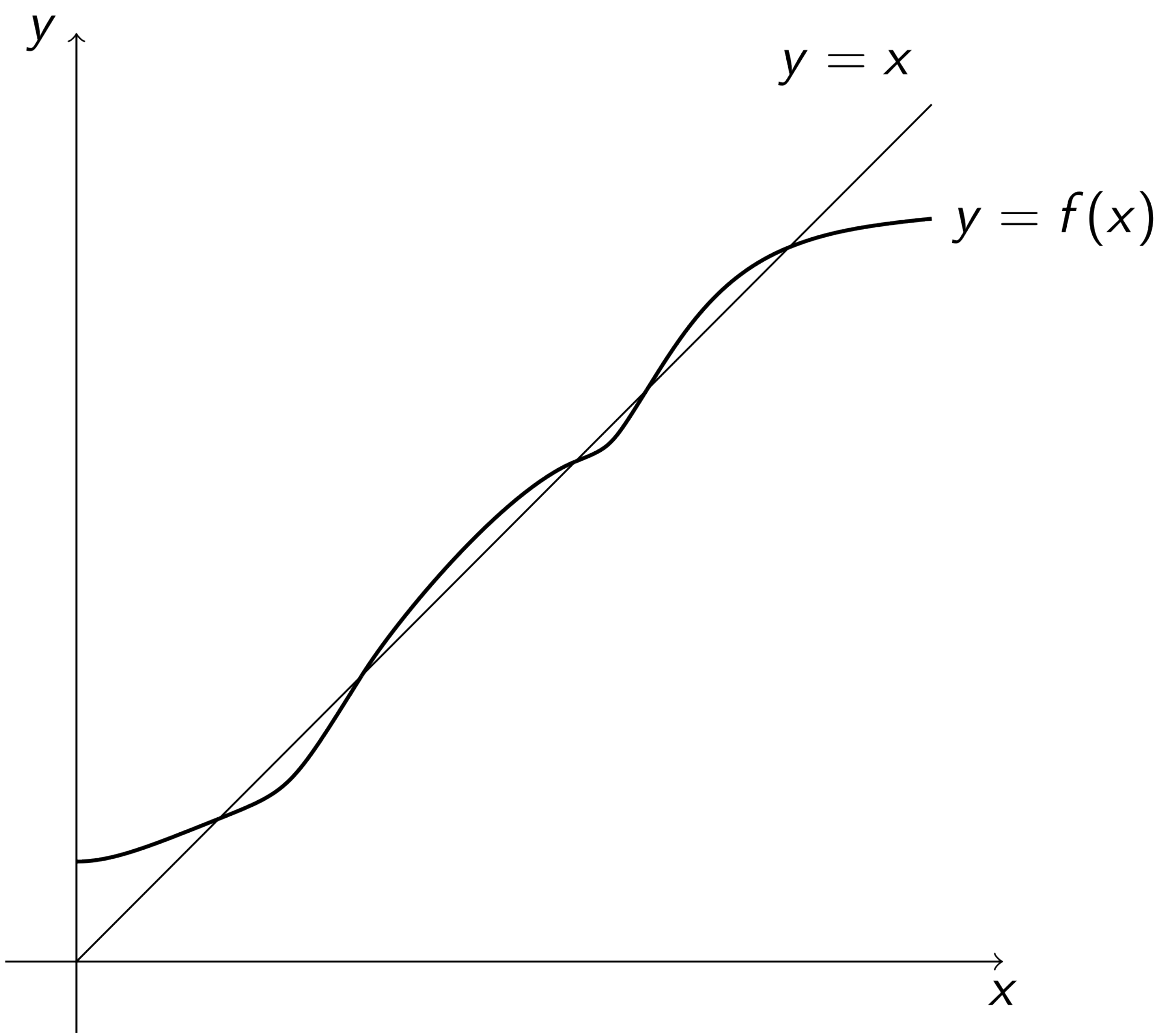
Knaster-Tarski

Theorem of Knaster-Tarski:

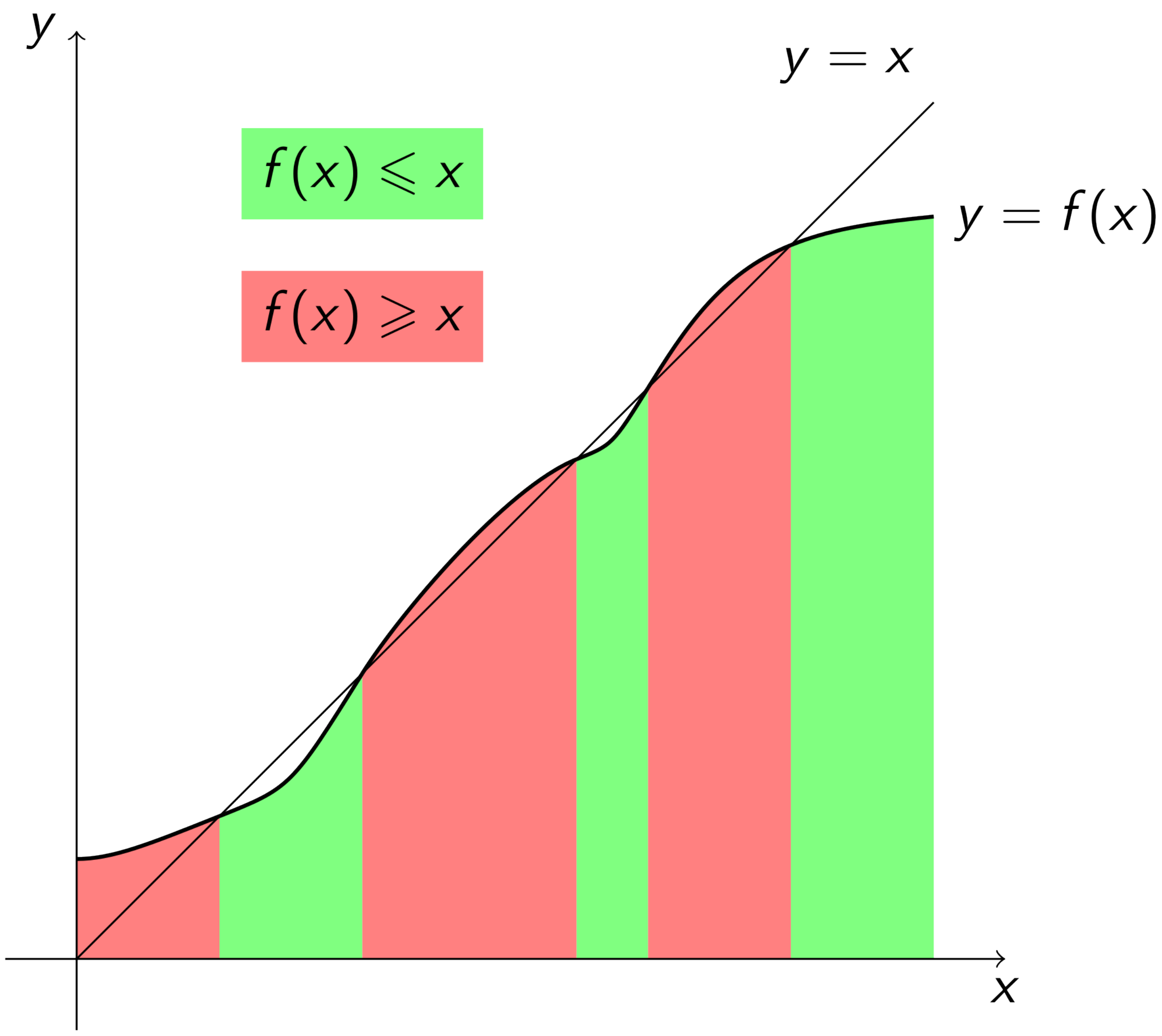
If ***S*** is a complete lattice and ***f*** is a monotonic function on this lattice, then ***f*** has a ***least fixed point***:

$$\mathrm{lfp}\, f = \bigsqcap \{x \in S \mid f(x) \sqsubseteq x\}$$

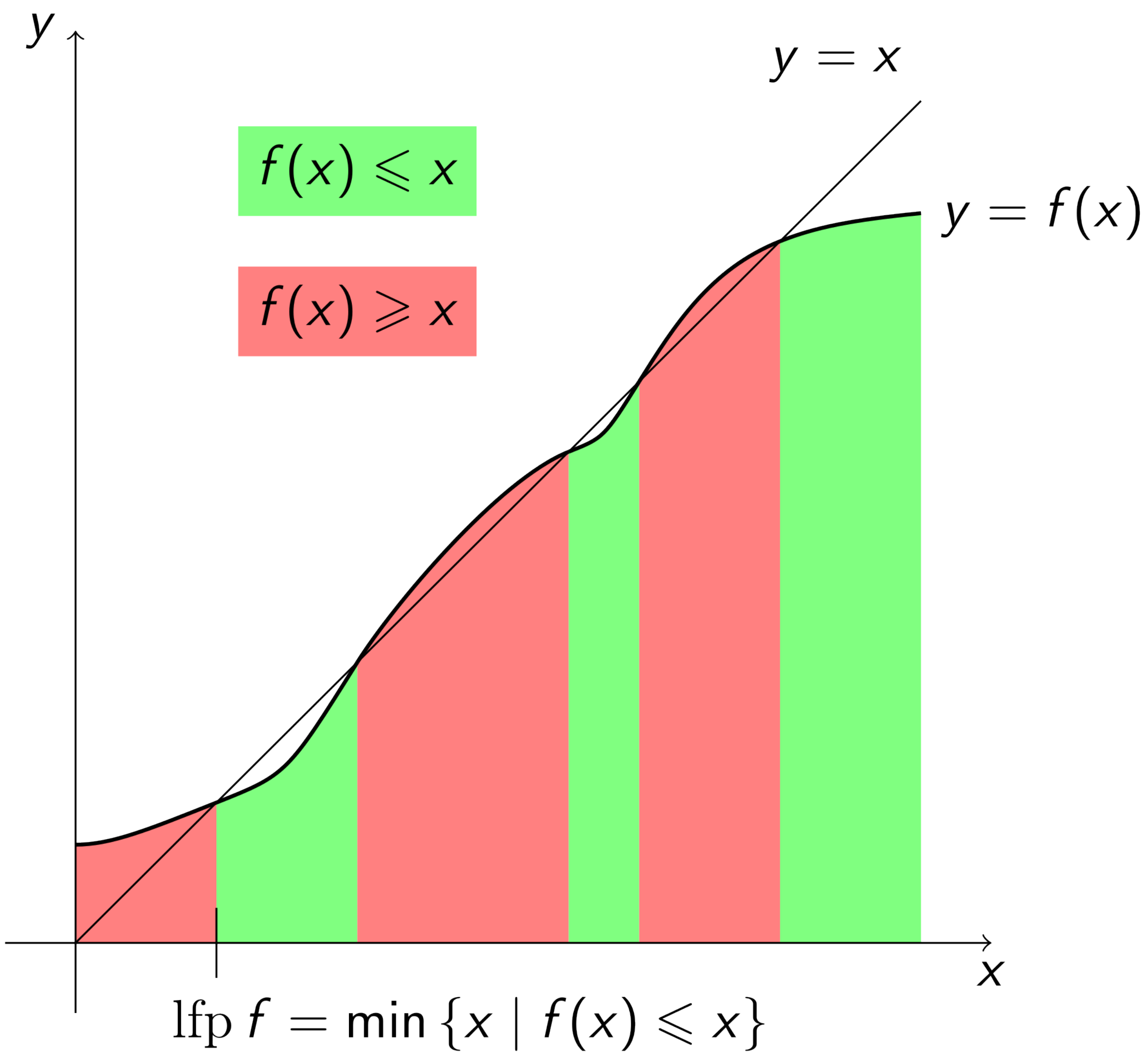
Knaster-Tarski - Idea



Knaster-Tarski - Idea



Knaster-Tarski - Idea



Semantic

- $L \to \mathcal{P}(\mathbb{V} \to \mathbb{Z})$ is a complete lattice
- Let $F : (L \to \mathcal{P}(\mathbb{V} \to \mathbb{Z})) \to (L \to \mathcal{P}(\mathbb{V} \to \mathbb{Z}))$

$$F(R) = \begin{cases} 0 & \mapsto \quad (\mathbb{V} \to \mathbb{Z}) \\ l' & \mapsto \quad \displaystyle\bigcup_{(l,c,l') \in A} [\![c]\!]_{\mathrm{C}} \, (R(l)) \end{cases}$$

- ***F*** is monotonic
- Thus ***lfp F*** exists – Knaster-Tarski

Semantic

- ***lfp F*** is also the smallest solution to our system:

$$\begin{cases} R_0 = \mathbb{V} \to \mathbb{Z} & \\ R_{l'} = \displaystyle\bigcup_{(l,c,l')\in A} [\![c]\!]_{\mathrm{C}}\,(R_l) & l' \neq 0 \end{cases}$$

- Thus our semantic is well defined!

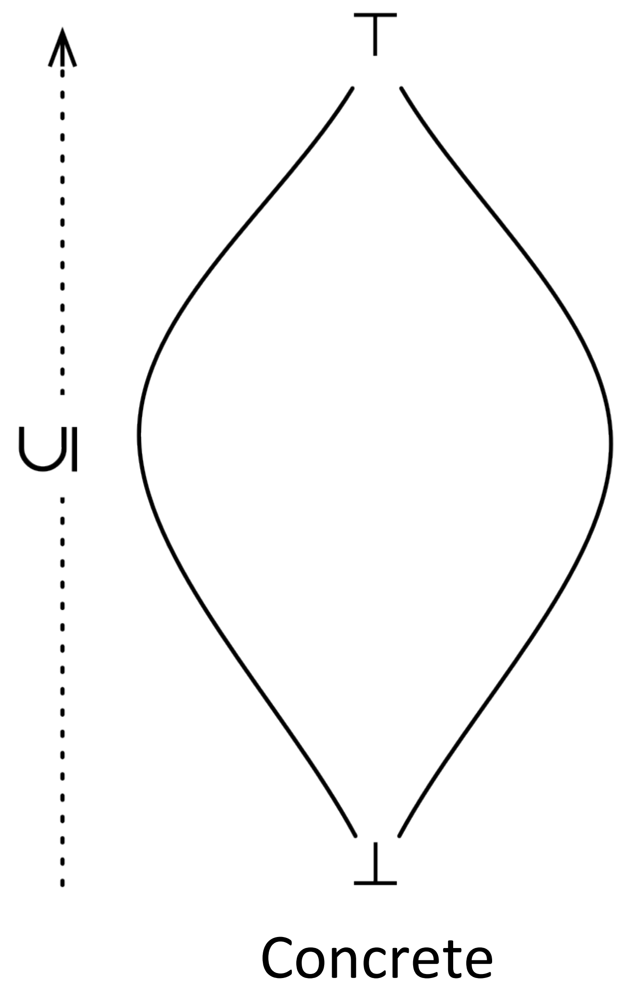
Semantic

- Unfortunately, the concrete semantic cannot be calculated
- We will compute an over-approximation!

Abstract Interpretation

Abstract Interpretation is a constructive theory of sound approximation of fixed points of monotonic functions on complete lattices.

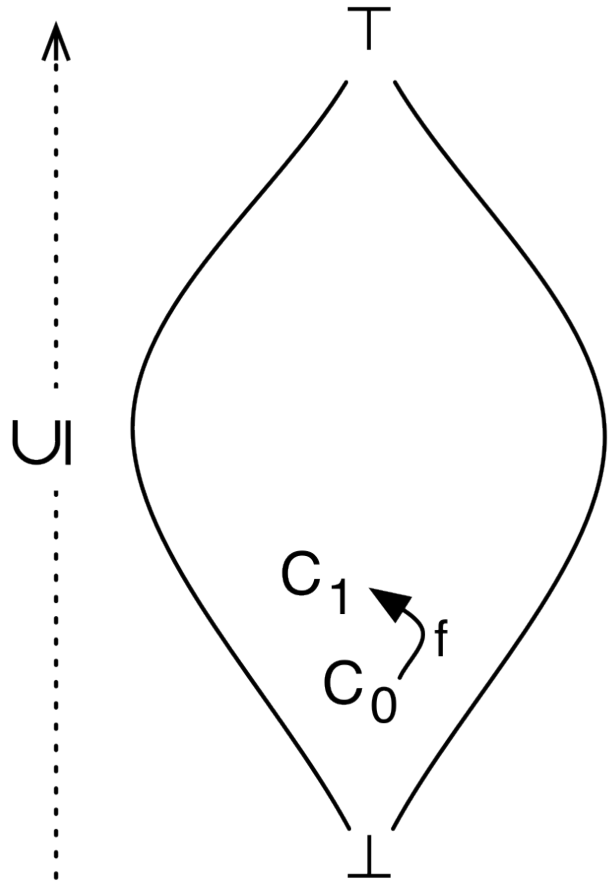
Fixed Point Iteration



Method of computing a fixed point:

x, *f*(x), *f*(*f*(x)), *f*(*f*(*f*(x))), ...

Fixed Point Iteration



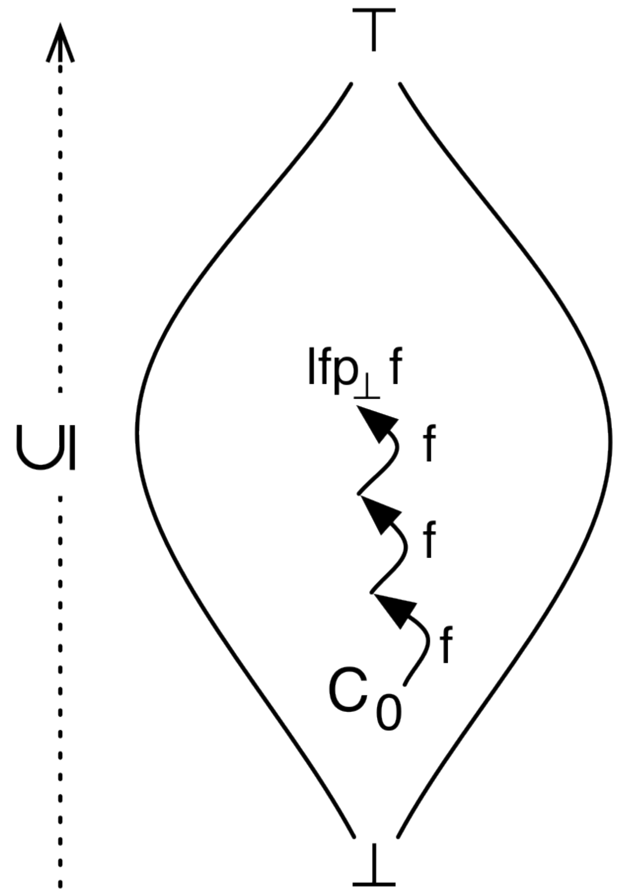
Concrete

Method of computing a fixed point:

x, f(x), f(f(x)), f(f(f(x))), ...

Start with a point C_0

Fixed Point Iteration



Concrete

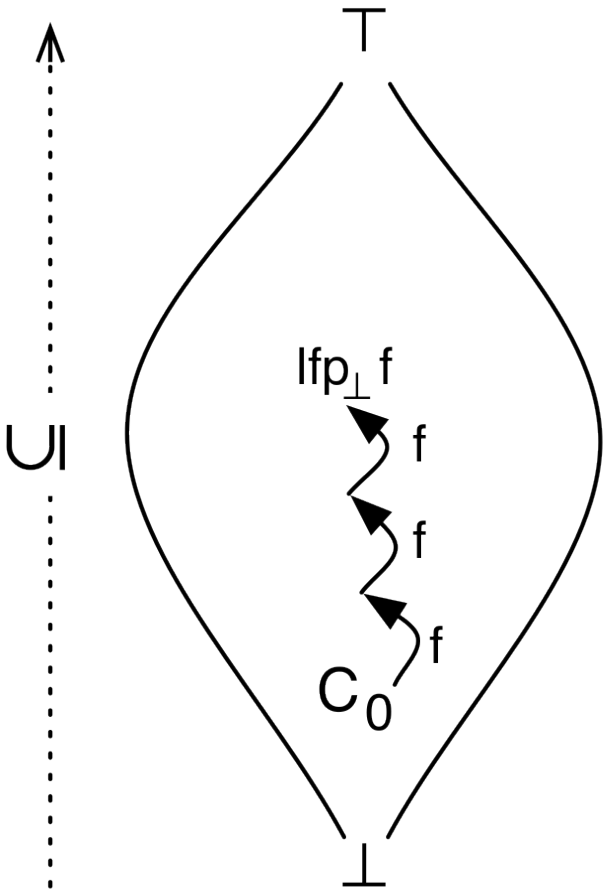
Method of computing a fixed point:

x, f(x), f(f(x)), f(f(f(x))), ...

Start with a point C_0

$f^n(x) = f^{n+1}(x) \Rightarrow$ ***lfp f*** found!

Fixed Point Iteration



Concrete

Method of computing a fixed point:

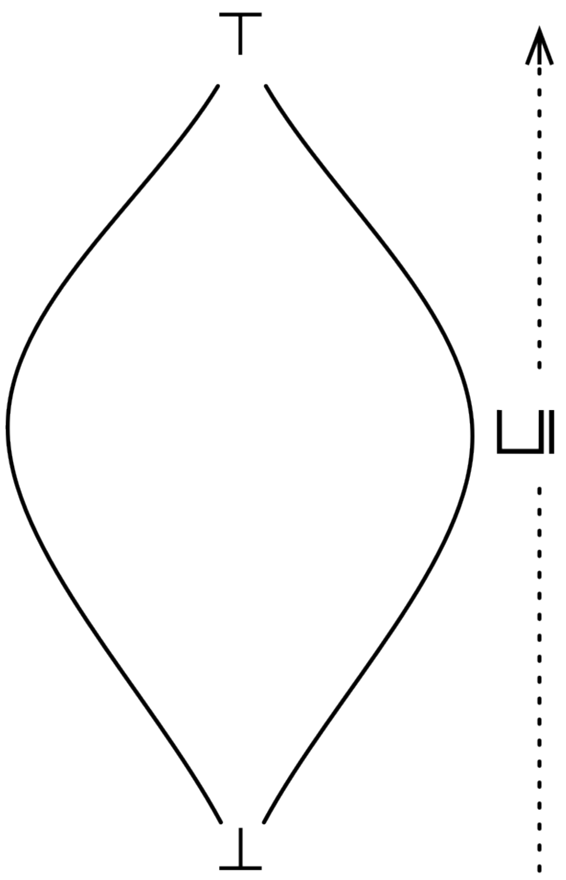
x, f(x), f(f(x)), f(f(f(x))), …

Start with a point C_0

$f^n(x) = f^{n+1}(x) \Rightarrow$ ***lfp f*** found!

Problem: Computing ***lpf f*** is undecidable

Abstraction

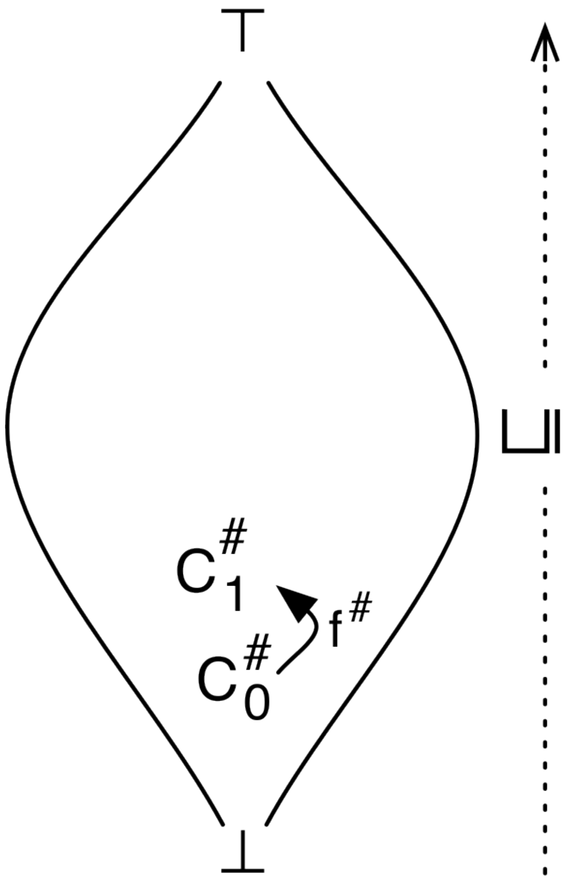


Abstract

Idea:

- Use a different ***complete lattice***
- Abstract the monotonic function: $f^{\#}$
- Abstract the entry point: $C_0^{\#}$

Abstraction

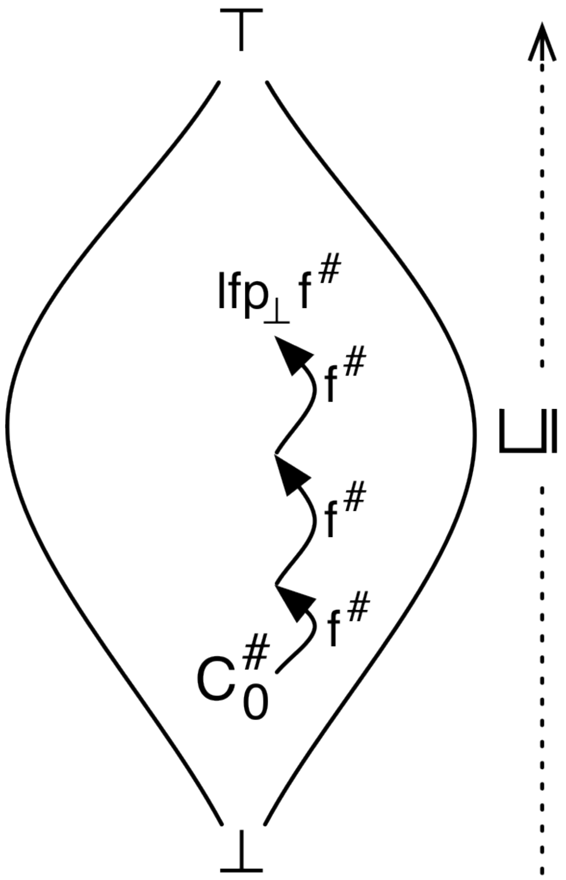


Abstract

Idea:

- Use a different ***complete lattice***
- Abstract the monotonic function: $f^\#$
- Abstract the entry point: $C_0^\#$

Abstraction

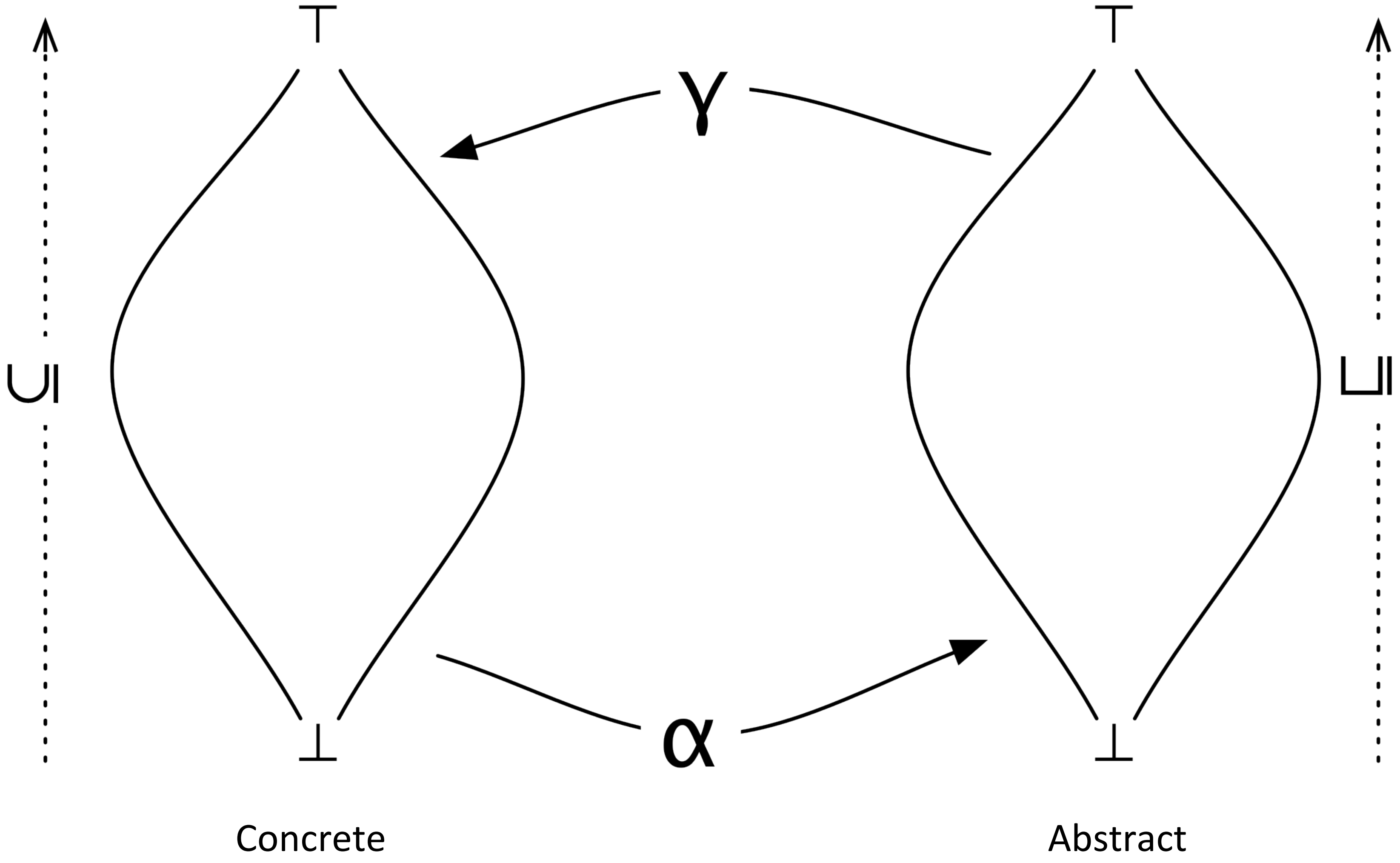


Abstract

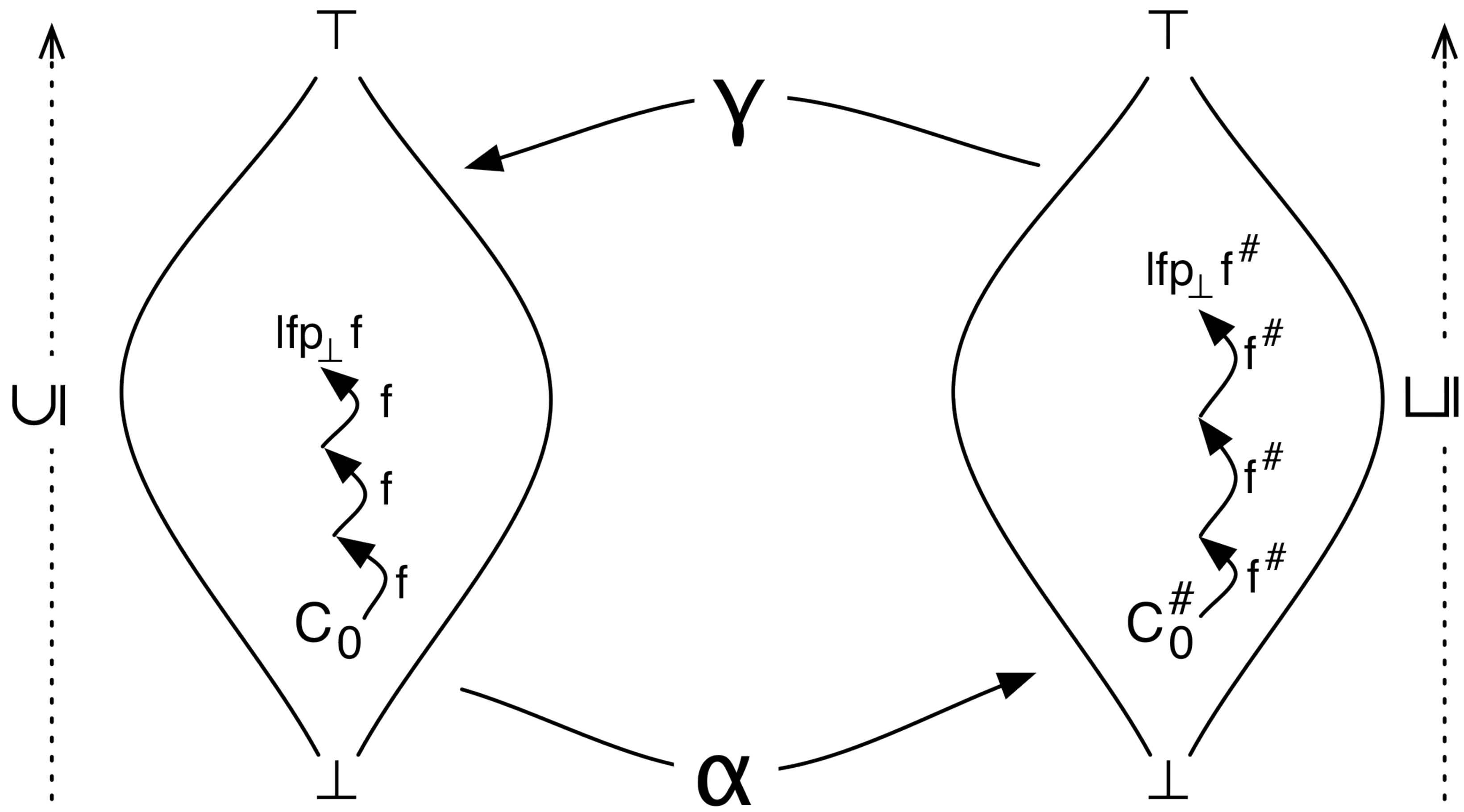
Idea:

- Use a different ***complete lattice***
- Abstract the monotonic function: $f^{\#}$
- Abstract the entry point: $C_0^{\#}$

Galois Connection



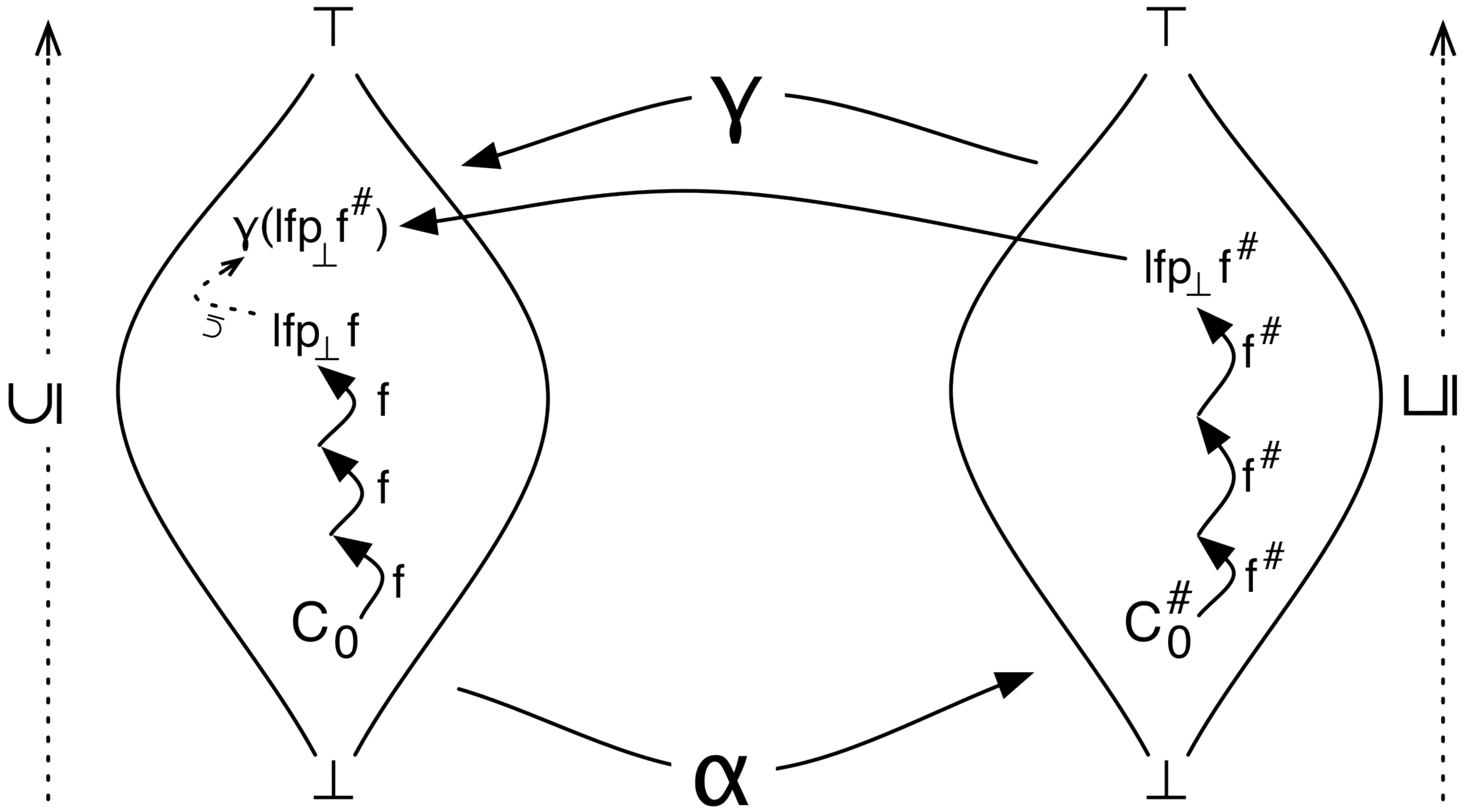
Galois Connection



Concrete

Abstract

Galois Connection



Concrete

Abstract

Build an abstraction

- Goal: Abstract $L \rightarrow \mathcal{P}(\mathbb{V} \rightarrow \mathbb{Z})$

Build an abstraction

- Goal: Abstract $L \to \mathcal{P}(\mathbb{V} \to \mathbb{Z})$
- *L* : finite set of program points – keep

Build an abstraction

- Goal: Abstract $L \to \mathcal{P}(\mathbb{V} \to \mathbb{Z})$
- L : finite set of program points – keep
- $\mathbb{V}$: finite set of variables – keep

Build an abstraction

- Goal: Abstract $L \to \mathcal{P}(\mathbb{V} \to \mathbb{Z})$
- L : finite set of program points – keep
- $\mathbb{V}$: finite set of variables – keep
- $\mathbb{Z}$: infinite set of integers – abstract!

Build an abstraction

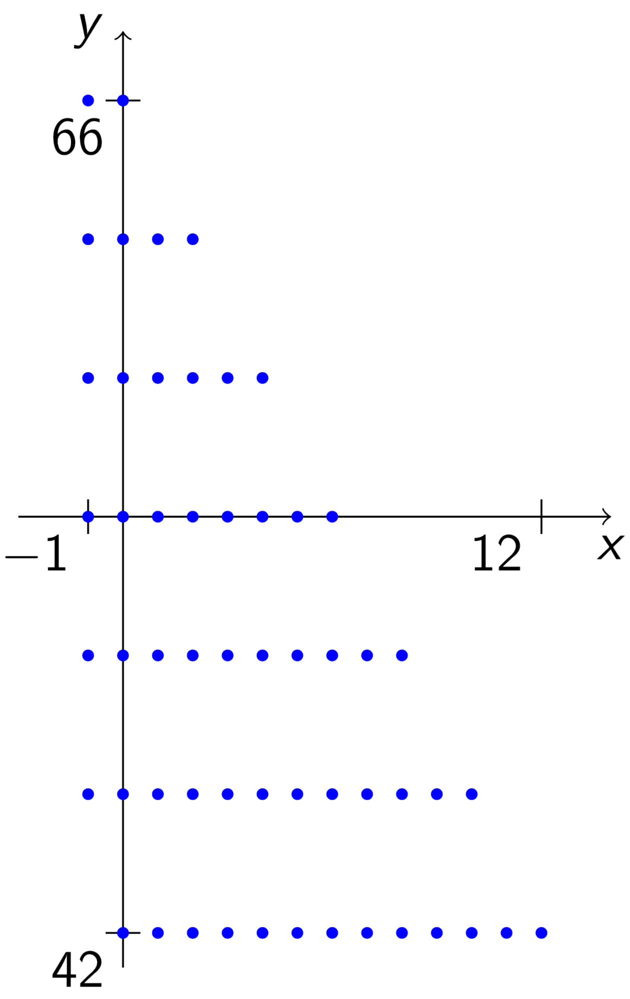
- Goal: Abstract $\mathcal{P}(\mathbb{V} \to \mathbb{Z})$

Build an abstraction

- Goal: Abstract $\mathcal{P}(\mathbb{V} \to \mathbb{Z})$
  - Non-relational : $\mathbb{V} \to \mathcal{P}(\mathbb{Z})$ then $\mathbb{V} \to \mathcal{D}^\sharp$

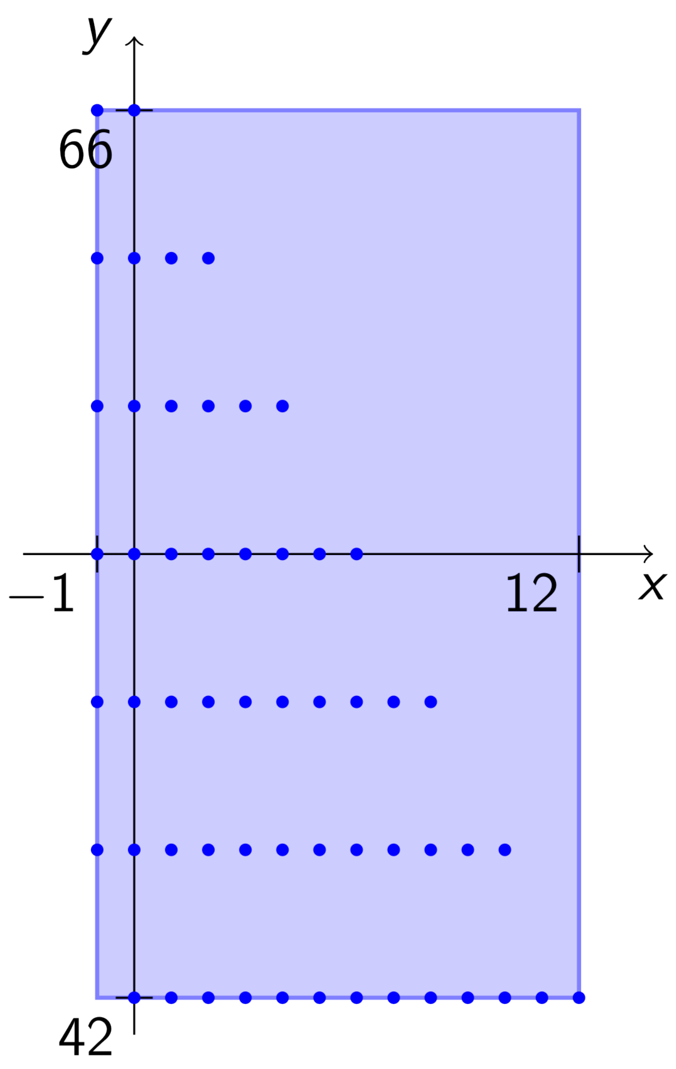
Build an abstraction

- Goal: Abstract $\mathcal{P}(\mathbb{V} \to \mathbb{Z})$
  - Non-relational : $\mathbb{V} \to \mathcal{P}(\mathbb{Z})$ then $\mathbb{V} \to \mathcal{D}^\sharp$
  - Relational : $\mathcal{D}^\sharp$

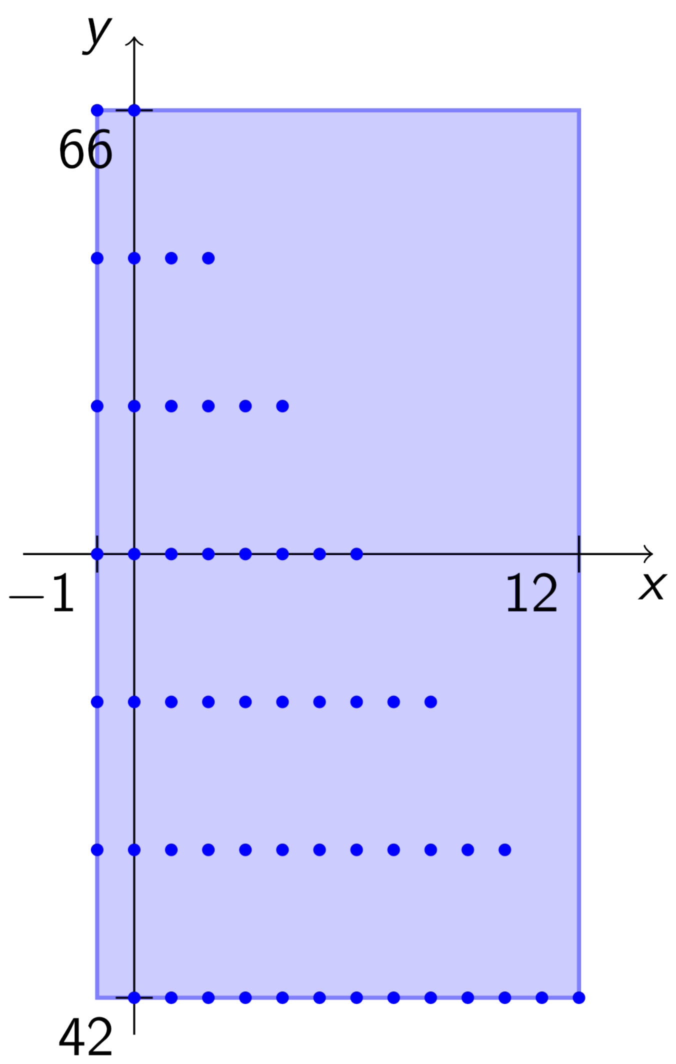


Reachable states at program point 2

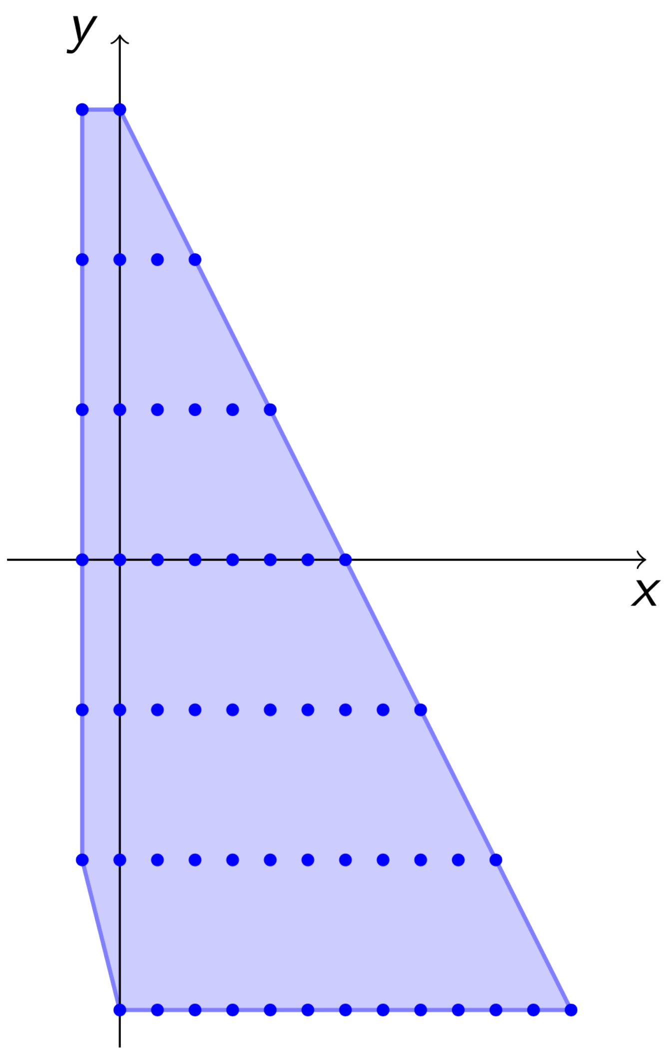
Non-relational abstraction



Non-relational abstraction



Relational abstraction



Interval domain

Interval lattice $(\mathcal{D}^\sharp, \sqsubseteq^\sharp)$:

$$
\begin{array}{ccccccc}
 & & &]\!]-\infty, +\infty[\![& & & \\
 & & & \vdots & & & \\
 & & \cdots & [\![-1, 1]\!] & \cdots & & \\
 & \cdots & [\![-1, 0]\!] & & [\![0, 1]\!] & \cdots & \\
\cdots & [\![-1, -1]\!] & & [\![0, 0]\!] & & [\![1, 1]\!] & \cdots \\
 & & & \bot & & &
\end{array}
$$

We first need to make sure it is a ***complete lattice***

Interval domain

Concretization function:

$$
\begin{aligned}
\gamma(\rrbracket -\infty, +\infty\llbracket) &= \rrbracket -\infty, +\infty\llbracket \\
\gamma(\rrbracket -\infty, n\rrbracket) &= \rrbracket -\infty, n\rrbracket \\
\gamma(\llbracket n, +\infty\llbracket) &= \llbracket n, +\infty\llbracket \\
\gamma(\llbracket n_1, n_2\rrbracket) &= \llbracket n_1, n_2\rrbracket \\
\gamma(\perp) &= \emptyset
\end{aligned}
$$

Interval domain

Abstraction function:

$$\alpha(S) = \begin{cases} [\![n_1, n_2]\!] & \text{with} \quad n_1 = \min S \text{ and } n_2 = \max S \\ \bot & \text{if} \quad S = \emptyset \end{cases}$$

Interval domain

(α, γ) is a Galois connection

Semantic of Expressions

Semantic of expressions: $[\![e]\!]^\sharp_{\mathrm{E}} : (\mathbb{V} \to \mathcal{D}^\sharp) \to \mathcal{D}^\sharp$

Semantic of Expressions

Semantic of expressions: $[\![e]\!]^\sharp_{\mathrm{E}} : (\mathbb{V} \to \mathcal{D}^\sharp) \to \mathcal{D}^\sharp$

$$
\begin{aligned}
[\![v]\!]^\sharp_{\mathrm{E}}(\rho) &= \rho(v) \\
[\![n]\!]^\sharp_{\mathrm{E}}(\rho) &= n^\sharp \\
[\![\mathbf{rand}(n_1, n_2)]\!]^\sharp_{\mathrm{E}}(\rho) &= \mathbf{rand}^\sharp(n_1, n_2) \\
[\![e_1 + e_2]\!]^\sharp_{\mathrm{E}}(\rho) &= [\![e_1]\!]^\sharp_{\mathrm{E}} +^\sharp [\![e_2]\!]^\sharp_{\mathrm{E}} \\
\ldots
\end{aligned}
$$

Semantic of Expressions

Abstract operators:

$$n^\sharp = \alpha(\{\, n \,\}) = [\![n, n]\!]$$

Semantic of Expressions

Abstract operators:

$$n^\sharp = \alpha(\{\, n \,\}) = [\![n, n]\!]$$

$$\textbf{rand}^\sharp(n_1, n_2) = \alpha([\![n_1, n_2]\!]) = \begin{cases} [\![n_1, n_2]\!] & \text{si } n_1 \leqslant n_2 \\ \perp & \text{si } n_1 > n_2 \end{cases}$$

Semantic of Expressions

Abstract operators:

$$n^\sharp = \alpha(\{\, n \,\}) = [\![n, n]\!]$$

$$\textbf{rand}^\sharp\textbf{(}n_1, n_2\textbf{)} = \alpha([\![n_1, n_2]\!]) = \begin{cases} [\![n_1, n_2]\!] & \text{si } n_1 \leqslant n_2 \\ \bot & \text{si } n_1 > n_2 \end{cases}$$

$$x^\sharp +^\sharp y^\sharp = \alpha\left(\left\{x + y \,\middle|\, x \in \gamma(x^\sharp), y \in \gamma(y^\sharp)\right\}\right) =$$

$$\begin{cases} [\![a + c, b + d]\!] & \text{avec } x^\sharp = [\![a, b]\!] \text{ et } y^\sharp = [\![c, d]\!] \\ \bot & \text{si } x^\sharp = \bot \text{ ou } y^\sharp = \bot \end{cases}$$

$\dots$

Semantic of Expressions

Abstract operators:

$$x^\sharp -^\sharp y^\sharp = \alpha\left(\left\{x - y \;\middle|\; x \in \gamma(x^\sharp), y \in \gamma(y^\sharp)\right\}\right) =$$

$$\begin{cases} [\![a - d, b - c]\!] & \text{avec } x^\sharp = [\![a, b]\!] \text{ et } y^\sharp = [\![c, d]\!] \\ \bot & \text{si } x^\sharp = \bot \text{ ou } y^\sharp = \bot \end{cases}$$

Semantic of Expressions

Abstract operators:

$$x^\sharp -^\sharp y^\sharp = \alpha\left(\left\{x - y \;\middle|\; x \in \gamma(x^\sharp), y \in \gamma(y^\sharp)\right\}\right) =$$

$$\begin{cases} [\![a - d, b - c]\!] & \text{avec } x^\sharp = [\![a, b]\!] \text{ et } y^\sharp = [\![c, d]\!] \\ \bot & \text{si } x^\sharp = \bot \text{ ou } y^\sharp = \bot \end{cases}$$

$$x^\sharp \times^\sharp y^\sharp = \alpha\left(\left\{x \times y \;\middle|\; x \in \gamma(x^\sharp), y \in \gamma(y^\sharp)\right\}\right) =$$

$$\begin{cases} [\![\min(ab, ac, ad, bd), \max(ab, ac, ad, bd)]\!] & \text{avec} \ldots \\ \bot & \text{si} \ldots \end{cases}$$

Semantic of commands

Semantic of commands: $[\![c]\!]^{\sharp}_{\mathrm{C}} : (\mathbb{V} \to \mathcal{D}^{\sharp}) \to (\mathbb{V} \to \mathcal{D}^{\sharp})$

Semantic of commands

Semantic of commands: $[\![c]\!]^{\sharp}_{\mathrm{C}} : (\mathbb{V} \to \mathcal{D}^{\sharp}) \to (\mathbb{V} \to \mathcal{D}^{\sharp})$

$$[\![v = e]\!]^{\sharp}_{\mathrm{C}}\,(\rho) = \rho\left[v \mapsto [\![e]\!]^{\sharp}_{\mathrm{E}}\,\rho\right]$$

$$[\![e > 0]\!]^{\sharp}_{\mathrm{C}}\,(\rho) = \begin{cases} \rho\left[v \mapsto \rho(v) \sqcap^{\sharp} \alpha\left([\![1, +\infty [\![\right)\right] & \text{si } e = v \\ \rho & \text{sinon} \end{cases}$$

Semantic of programs

Semantic of programs: $[\![(L, A)]\!]^\sharp \; : L \to (\mathbb{V} \to \mathcal{D}^\sharp)$

Semantic of programs

Semantic of programs: $[\![(L, A)]\!]^\sharp \; : L \to (\mathbb{V} \to \mathcal{D}^\sharp)$

It is the smallest solution (in term of inclusion) of the following system:

$$\begin{cases} R_0^\sharp = \mathbb{V} \to \top & \\ R_{l'}^\sharp = \displaystyle\bigsqcup_{(l,c,l') \in A}{}^{\sharp}_{\mathrm{nr}} \; [\![c]\!]^\sharp_{\mathrm{C}} \, (R_l^\sharp) & l' \neq 0 \end{cases}$$

Semantic of programs

Semantic of programs: $[\![(L, A)]\!]^\sharp \; : L \to (\mathbb{V} \to \mathcal{D}^\sharp)$

It is the smallest solution (in term of inclusion) of the following system:

$$\begin{cases} R_0^\sharp = \mathbb{V} \to \top & \\ R_{l'}^\sharp = \displaystyle\bigsqcup_{(l,c,l')\in A}{}^{\sharp}_{\mathrm{nr}} \; [\![c]\!]^\sharp_{\mathrm{C}}\,(R_l^\sharp) & \qquad l' \neq 0 \end{cases}$$

Knaster-Tarski: the solution exists!

Computing the fixpoint

- We define $F^\sharp : (L \to (\mathbb{V} \to \mathcal{D}^\sharp)) \to (L \to (\mathbb{V} \to \mathcal{D}^\sharp))$

$$F^\sharp(R^\sharp) = \begin{cases} 0 & \mapsto & \top_{\mathrm{nr}} \\ l' & \mapsto & \displaystyle\bigsqcup^\sharp_{\mathrm{nr}\,(l,c,l')\in A} \llbracket c \rrbracket^\sharp_{\mathrm{C}} \left(R^\sharp(l)\right) \end{cases}$$

Computing the fixpoint

- We define $F^\sharp : (L \to (\mathbb{V} \to \mathcal{D}^\sharp)) \to (L \to (\mathbb{V} \to \mathcal{D}^\sharp))$

$$F^\sharp(R^\sharp) = \left\{ \begin{array}{lcl} 0 & \mapsto & \top_{\mathrm{nr}} \\ l' & \mapsto & \displaystyle\bigsqcup^\sharp_{\mathrm{nr}}{}_{(l,c,l')\in A} \; [\![c]\!]^\sharp_{\mathrm{C}}\,(R^\sharp(l)) \end{array} \right.$$

- $F^\#$ is monotonic and computable

Computing the fixpoint

- We define $F^\sharp : (L \to (\mathbb{V} \to \mathcal{D}^\sharp)) \to (L \to (\mathbb{V} \to \mathcal{D}^\sharp))$

$$F^\sharp(R^\sharp) = \begin{cases} 0 & \mapsto & \top_{\mathrm{nr}} \\ l' & \mapsto & \displaystyle\bigsqcup^\sharp_{\mathrm{nr}\,(l,c,l')\in A} \llbracket c \rrbracket^\sharp_{\mathrm{C}} \left(R^\sharp(l)\right) \end{cases}$$

- $F^\#$ is monotonic and computable
- lfp $F^\#$ is the abstract semantic of our program

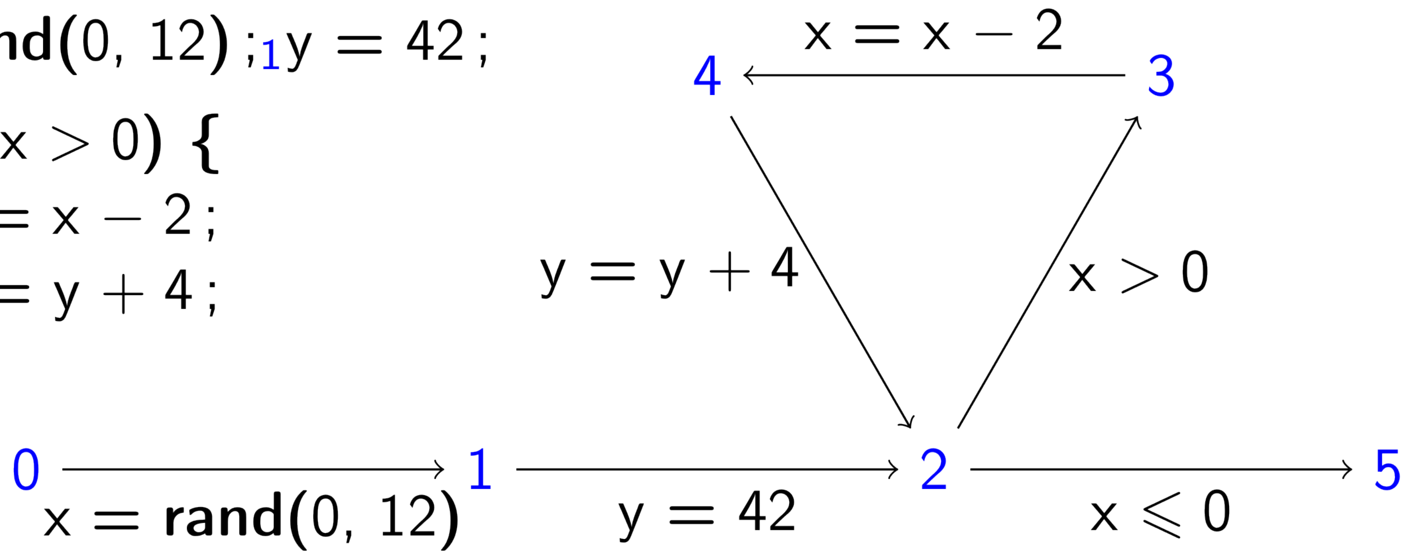
Computing the fixpoint

- We define $F^\sharp : (L \to (\mathbb{V} \to \mathcal{D}^\sharp)) \to (L \to (\mathbb{V} \to \mathcal{D}^\sharp))$

$$F^\sharp(R^\sharp) = \begin{cases} 0 & \mapsto & \top_{\mathrm{nr}} \\ l' & \mapsto & \displaystyle\bigsqcup^\sharp_{\mathrm{nr}}{}_{(l,c,l')\in A} \; [\![c]\!]^\sharp_{\mathrm{C}}\,(R^\sharp(l)) \end{cases}$$

- $F^\#$ is monotonic and computable
- lfp $F^\#$ is the abstract semantic of our program
- Iterative method to compute the least fixed point:
  - ${R^\sharp}^0 := L \to \perp_{\mathrm{nr}}$
  - ${R^\sharp}^{k+1} := F^\sharp\left({R^\sharp}^k\right)$
  - Stop when $R^{\#k+1} = R^{\#k}$

```
₀x = rand(0, 12) ;₁y = 42 ;
while ₂(x > 0) {
    ₃x = x − 2 ;
    ₄y = y + 4 ;
}₅
```



$$
\begin{aligned}
{R_0^\sharp}^{i+1} &= \top \\
{R_1^\sharp}^{i+1} &= {R_0^\sharp}^{i+1}\,[x \mapsto [\![0,12]\!]] \\
{R_2^\sharp}^{i+1} &= {R_1^\sharp}^{i+1}\,[y \mapsto [\![42,42]\!]] \sqcup^\sharp_{\mathrm{nr}} \\
&\quad {R_4^\sharp}^{i}\left[y \mapsto {R_4^\sharp}^{i}(y) +^\sharp [\![4,4]\!]\right] \\
{R_3^\sharp}^{i+1} &= {R_2^\sharp}^{i+1}\left[x \mapsto {R_2^\sharp}^{i+1}(x)\right. \\
&\qquad\qquad \left.\sqcap^\sharp [\![1,+\infty[\![\right] \\
{R_4^\sharp}^{i+1} &= {R_3^\sharp}^{i+1}\left[x \mapsto {R_3^\sharp}^{i+1}(x) -^\sharp [\![2,2]\!]\right] \\
{R_5^\sharp}^{i+1} &= {R_2^\sharp}^{i+1}\left[x \mapsto {R_2^\sharp}^{i+1}(x)\right. \\
&\qquad\qquad \left.\sqcap^\sharp]\!]-\infty,0]\!]\right]
\end{aligned}
$$

$_{0}\mathsf{x} = \mathbf{rand}(0,\ 12)\ ;_{1}\mathsf{y} = 42\,;$

$\mathbf{while}\ _{2}(\mathsf{x} > 0)\ \{$

$\quad _{3}\mathsf{x} = \mathsf{x} - 2\,;$

$\quad _{4}\mathsf{y} = \mathsf{y} + 4\,;$

$\}_{5}$

$$
\begin{tikzcd}
 & & 4 \arrow[dr, "\mathsf{y} = \mathsf{y} + 4"'] & & 3 \arrow[ll, "\mathsf{x} = \mathsf{x} - 2"'] & \\
0 \arrow[r, "\mathsf{x} = \mathbf{rand}(0,\ 12)"'] & 1 \arrow[rr, "\mathsf{y} = 42"'] & & 2 \arrow[ur, "\mathsf{x} > 0"'] \arrow[rr, "\mathsf{x} \leqslant 0"'] & & 5
\end{tikzcd}
$$

$$
\begin{aligned}
{R_0^{\sharp}}^{i+1} &= \top \\
{R_1^{\sharp}}^{i+1} &= {R_0^{\sharp}}^{i+1}\,[x \mapsto [\![0, 12]\!]] \\
{R_2^{\sharp}}^{i+1} &= {R_1^{\sharp}}^{i+1}\,[y \mapsto [\![42, 42]\!]] \sqcup^{\sharp}_{\mathrm{nr}} \\
&\quad {R_4^{\sharp}}^{i}\left[y \mapsto {R_4^{\sharp}}^{i}(y) +^{\sharp} [\![4, 4]\!]\right] \\
{R_3^{\sharp}}^{i+1} &= {R_2^{\sharp}}^{i+1}\left[x \mapsto {R_2^{\sharp}}^{i+1}(x) \right. \\
&\qquad\qquad \left. \sqcap^{\sharp} [\![1, +\infty[\![\right] \\
{R_4^{\sharp}}^{i+1} &= {R_3^{\sharp}}^{i+1}\left[x \mapsto {R_3^{\sharp}}^{i+1}(x) -^{\sharp} [\![2, 2]\!]\right] \\
{R_5^{\sharp}}^{i+1} &= {R_2^{\sharp}}^{i+1}\left[x \mapsto {R_2^{\sharp}}^{i+1}(x) \right. \\
&\qquad\qquad \left. \sqcap^{\sharp}]\!]-\infty, 0]\!]\right]
\end{aligned}
$$

l	${R_l^{\sharp}}^{0}$	${R_l^{\sharp}}^{1}$	${R_l^{\sharp}}^{2}$
0	$(\bot, \bot)$		
1	$(\bot, \bot)$		
2	$(\bot, \bot)$		
3	$(\bot, \bot)$		
4	$(\bot, \bot)$		
5	$(\bot, \bot)$		

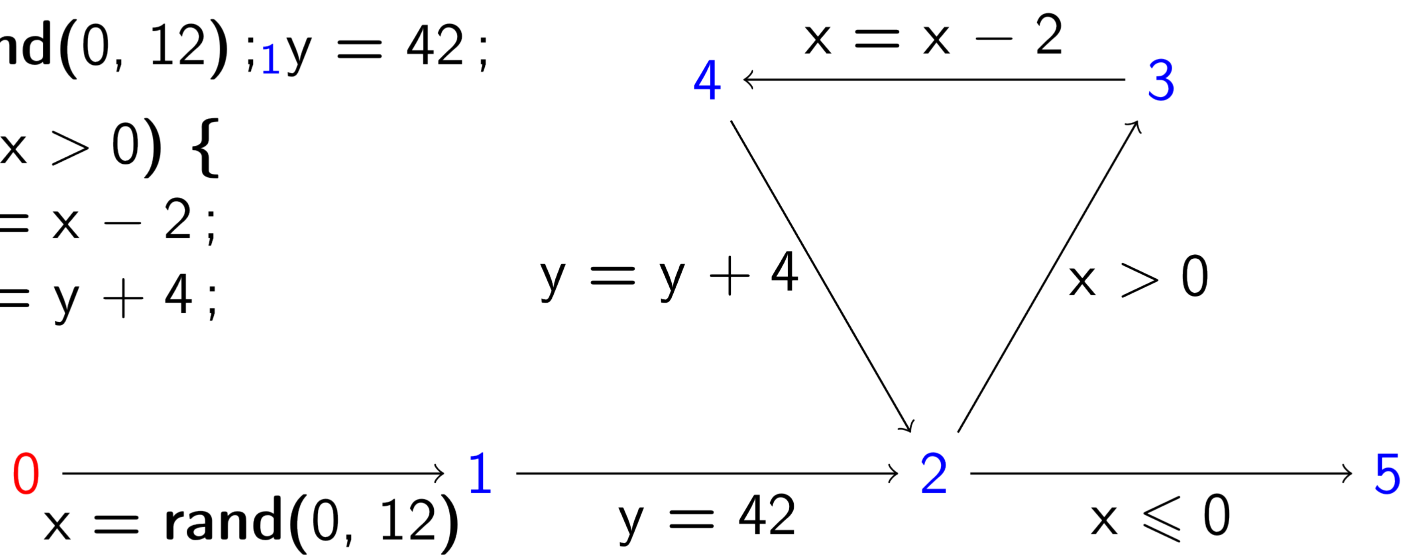
$_0$x = **rand**(0, 12) ;$_1$y = 42 ;

while $_2$(x > 0) {

  $_3$x = x − 2 ;

  $_4$y = y + 4 ;

}$_5$



$$
\begin{aligned}
{R^\sharp_0}^{i+1} &= \top \\
{R^\sharp_1}^{i+1} &= {R^\sharp_0}^{i+1}\,[x \mapsto [\![0,12]\!]] \\
{R^\sharp_2}^{i+1} &= {R^\sharp_1}^{i+1}\,[y \mapsto [\![42,42]\!]] \sqcup^\sharp_{\mathrm{nr}} \\
&\quad {R^\sharp_4}^{i}\left[y \mapsto {R^\sharp_4}^{i}(y) +^\sharp [\![4,4]\!]\right] \\
{R^\sharp_3}^{i+1} &= {R^\sharp_2}^{i+1}\left[x \mapsto {R^\sharp_2}^{i+1}(x) \right.\\
&\qquad\qquad \left. \sqcap^\sharp [\![1,+\infty[\![\right] \\
{R^\sharp_4}^{i+1} &= {R^\sharp_3}^{i+1}\left[x \mapsto {R^\sharp_3}^{i+1}(x) -^\sharp [\![2,2]\!]\right] \\
{R^\sharp_5}^{i+1} &= {R^\sharp_2}^{i+1}\left[x \mapsto {R^\sharp_2}^{i+1}(x) \right.\\
&\qquad\qquad \left. \sqcap^\sharp]\!]-\infty,0]\!]\right]
\end{aligned}
$$

l	${R^\sharp_l}^0$	${R^\sharp_l}^1$	${R^\sharp_l}^2$
0	$(\bot,\bot)$	$(\top,\top)$	
1	$(\bot,\bot)$		
2	$(\bot,\bot)$		
3	$(\bot,\bot)$		
4	$(\bot,\bot)$		
5	$(\bot,\bot)$		

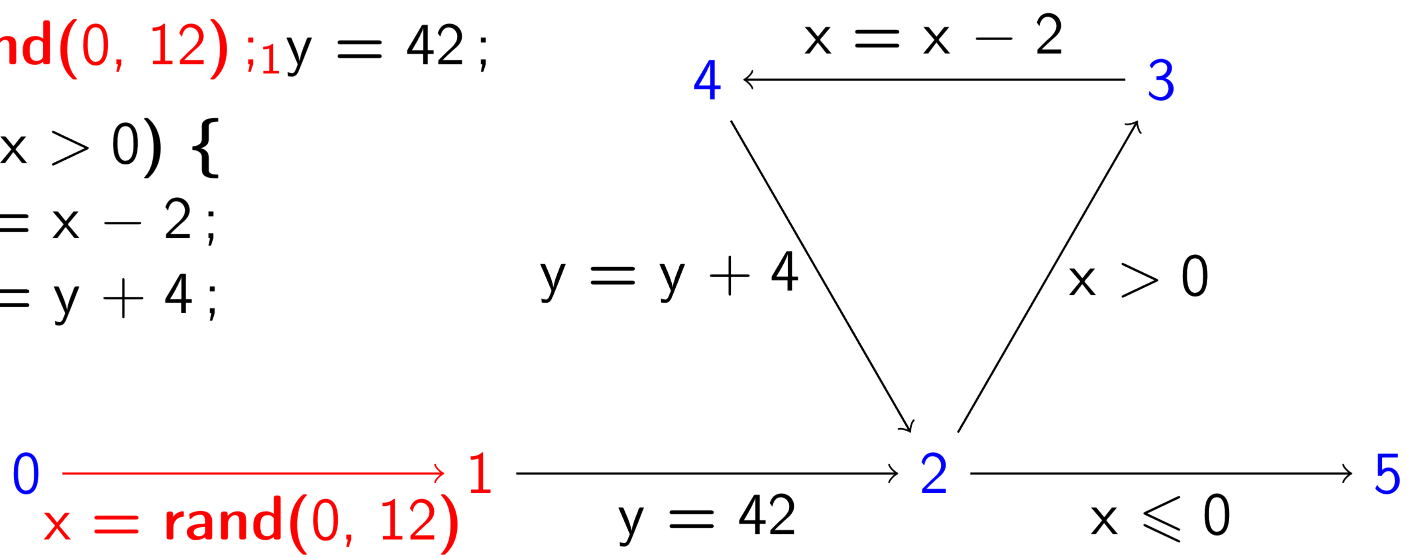
$_0$x $=$ **rand**(0, 12) ;$_1$y $=$ 42 ;

while $_2$(x $>$ 0) **{**

  $_3$x $=$ x $-$ 2 ;

  $_4$y $=$ y $+$ 4 ;

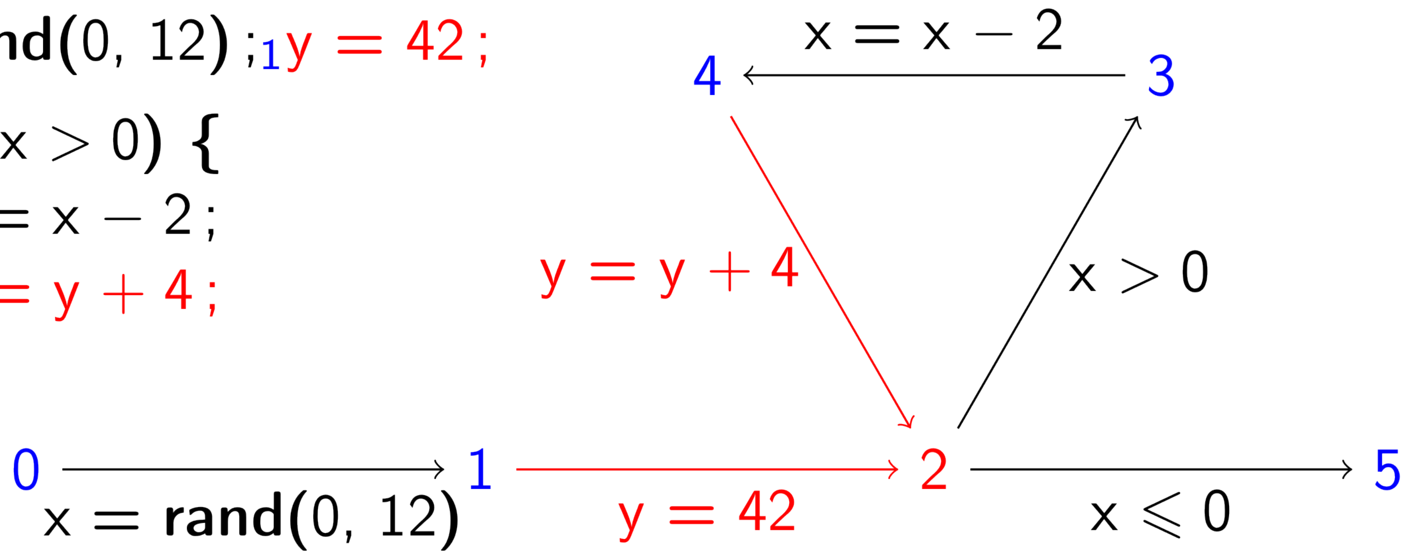
}$_5$



$$
\begin{aligned}
R_0^{\sharp\, i+1} &= \top \\
R_1^{\sharp\, i+1} &= R_0^{\sharp\, i+1}\,[x \mapsto [\![0, 12]\!]] \\
R_2^{\sharp\, i+1} &= R_1^{\sharp\, i+1}\,[y \mapsto [\![42, 42]\!]] \sqcup^{\sharp}_{\mathrm{nr}} \\
&\quad R_4^{\sharp\, i}\left[y \mapsto R_4^{\sharp\, i}(y) +^{\sharp} [\![4, 4]\!]\right] \\
R_3^{\sharp\, i+1} &= R_2^{\sharp\, i+1}\left[x \mapsto R_2^{\sharp\, i+1}(x) \right. \\
&\qquad\qquad \left. \sqcap^{\sharp} [\![1, +\infty[\![\right] \\
R_4^{\sharp\, i+1} &= R_3^{\sharp\, i+1}\left[x \mapsto R_3^{\sharp\, i+1}(x) -^{\sharp} [\![2, 2]\!]\right] \\
R_5^{\sharp\, i+1} &= R_2^{\sharp\, i+1}\left[x \mapsto R_2^{\sharp\, i+1}(x) \right. \\
&\qquad\qquad \left. \sqcap^{\sharp}]\!]-\infty, 0]\!] \right]
\end{aligned}
$$

l	$R_l^{\sharp\,0}$	$R_l^{\sharp\,1}$	$R_l^{\sharp\,2}$
0	$(\bot, \bot)$	$(\top, \top)$	
1	$(\bot, \bot)$	$([\![0, 12]\!], \top)$	
2	$(\bot, \bot)$		
3	$(\bot, \bot)$		
4	$(\bot, \bot)$		
5	$(\bot, \bot)$		

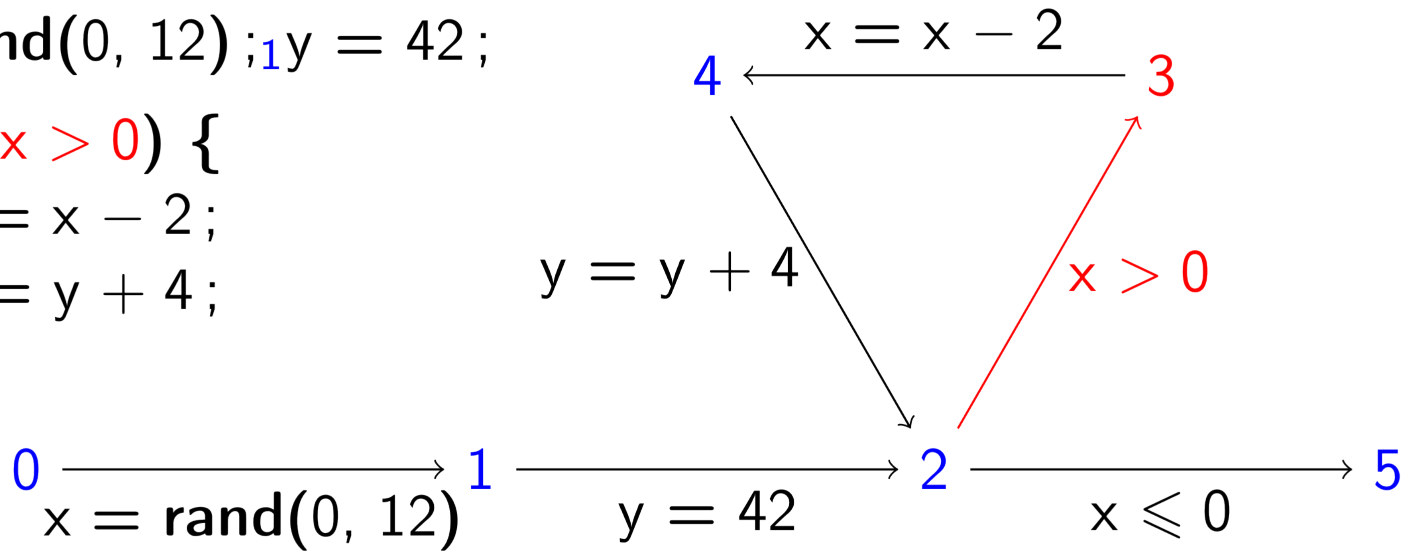
$$
\begin{array}{l}
{}_{0}\mathsf{x} = \mathbf{rand}(0,\ 12)\ ;{}_{1}\mathsf{y} = 42\ ; \\
\mathbf{while}\ {}_{2}(\mathsf{x} > 0)\ \{ \\
\quad {}_{3}\mathsf{x} = \mathsf{x} - 2\ ; \\
\quad {}_{4}\mathsf{y} = \mathsf{y} + 4\ ; \\
\}_{5}
\end{array}
$$



$$
\begin{aligned}
R_0^{\sharp\, i+1} &= \top \\
R_1^{\sharp\, i+1} &= R_0^{\sharp\, i+1}\,[x \mapsto [\![0, 12]\!]] \\
R_2^{\sharp\, i+1} &= R_1^{\sharp\, i+1}\,[y \mapsto [\![42, 42]\!]] \sqcup^{\sharp}_{\mathrm{nr}} \\
&\quad R_4^{\sharp\, i}\left[y \mapsto R_4^{\sharp\, i}(y) +^{\sharp} [\![4, 4]\!]\right] \\
R_3^{\sharp\, i+1} &= R_2^{\sharp\, i+1}\Big[x \mapsto R_2^{\sharp\, i+1}(x) \\
&\qquad\qquad \sqcap^{\sharp} [\![1, +\infty[\![\Big] \\
R_4^{\sharp\, i+1} &= R_3^{\sharp\, i+1}\left[x \mapsto R_3^{\sharp\, i+1}(x) -^{\sharp} [\![2, 2]\!]\right] \\
R_5^{\sharp\, i+1} &= R_2^{\sharp\, i+1}\Big[x \mapsto R_2^{\sharp\, i+1}(x) \\
&\qquad\qquad \sqcap^{\sharp}]\!]-\infty, 0]\!]\Big]
\end{aligned}
$$

l	$R_l^{\sharp\,0}$	$R_l^{\sharp\,1}$	$R_l^{\sharp\,2}$
0	$(\bot, \bot)$	$(\top, \top)$	
1	$(\bot, \bot)$	$([\![0, 12]\!], \top)$	
2	$(\bot, \bot)$	$([\![0, 12]\!], \{\,42\,\})$	
3	$(\bot, \bot)$		
4	$(\bot, \bot)$		
5	$(\bot, \bot)$		

$_0$x = **rand**(0, 12) ;$_1$y = 42 ;
while $_2$(x > 0) {
    $_3$x = x − 2 ;
    $_4$y = y + 4 ;
}$_5$



$$
\begin{aligned}
R_0^{\sharp\, i+1} &= \top \\
R_1^{\sharp\, i+1} &= R_0^{\sharp\, i+1}\,[x \mapsto [\![0,12]\!]] \\
R_2^{\sharp\, i+1} &= R_1^{\sharp\, i+1}\,[y \mapsto [\![42,42]\!]] \sqcup^\sharp_{\mathrm{nr}} \\
&\quad R_4^{\sharp\, i}\left[y \mapsto R_4^{\sharp\, i}(y) +^\sharp [\![4,4]\!]\right] \\
\color{red}{R_3^{\sharp\, i+1}} &\color{red}{= R_2^{\sharp\, i+1}\left[x \mapsto R_2^{\sharp\, i+1}(x) \right.} \\
&\qquad\qquad \color{red}{\left. \sqcap^\sharp [\![1,+\infty[\![\right]} \\
R_4^{\sharp\, i+1} &= R_3^{\sharp\, i+1}\left[x \mapsto R_3^{\sharp\, i+1}(x) -^\sharp [\![2,2]\!]\right] \\
R_5^{\sharp\, i+1} &= R_2^{\sharp\, i+1}\left[x \mapsto R_2^{\sharp\, i+1}(x) \right. \\
&\qquad\qquad \left. \sqcap^\sharp\,]\!]-\infty,0]\!]\right]
\end{aligned}
$$

l	${R_l^\sharp}^0$	${R_l^\sharp}^1$	${R_l^\sharp}^2$
0	$(\bot,\bot)$	$(\top,\top)$	
1	$(\bot,\bot)$	$([\![0,12]\!],\top)$	
2	$(\bot,\bot)$	$([\![0,12]\!],\{42\})$	
3	$(\bot,\bot)$	$\color{red}{([\![1,12]\!],\{42\})}$	
4	$(\bot,\bot)$		
5	$(\bot,\bot)$		

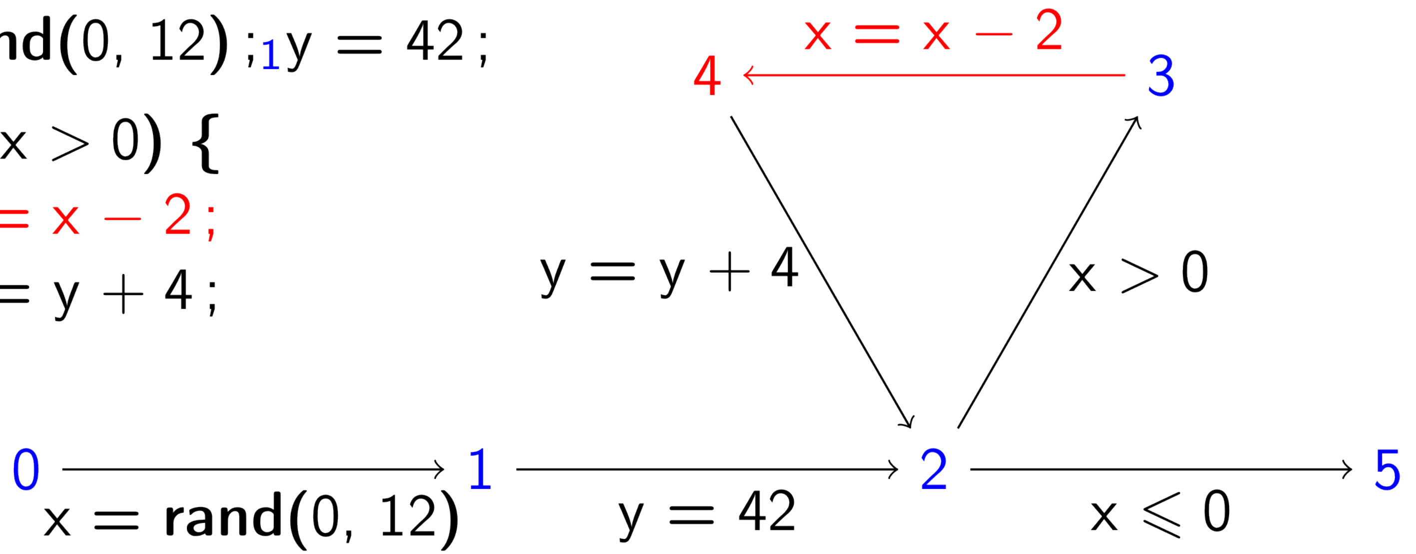
$_0\mathsf{x} = \mathbf{rand}(0,\ 12)\ ;{}_1\mathsf{y} = 42\ ;$

$\mathbf{while}\ {}_2(\mathsf{x} > 0)\ \{$

$\quad {}_3\mathsf{x} = \mathsf{x} - 2\ ;$

$\quad {}_4\mathsf{y} = \mathsf{y} + 4\ ;$

$\}_5$



$$
\begin{aligned}
R^{\sharp\,i+1}_0 &= \top \\
R^{\sharp\,i+1}_1 &= R^{\sharp\,i+1}_0\,[x \mapsto [\![0,12]\!]] \\
R^{\sharp\,i+1}_2 &= R^{\sharp\,i+1}_1\,[y \mapsto [\![42,42]\!]] \sqcup^\sharp_{\mathrm{nr}} \\
&\quad R^{\sharp\,i}_4\left[y \mapsto R^{\sharp\,i}_4(y) +^\sharp [\![4,4]\!]\right] \\
R^{\sharp\,i+1}_3 &= R^{\sharp\,i+1}_2\left[x \mapsto R^{\sharp\,i+1}_2(x)\right. \\
&\qquad\qquad \left.\sqcap^\sharp [\![1,+\infty[\![\right] \\
R^{\sharp\,i+1}_4 &= R^{\sharp\,i+1}_3\left[x \mapsto R^{\sharp\,i+1}_3(x) -^\sharp [\![2,2]\!]\right] \\
R^{\sharp\,i+1}_5 &= R^{\sharp\,i+1}_2\left[x \mapsto R^{\sharp\,i+1}_2(x)\right. \\
&\qquad\qquad \left.\sqcap^\sharp]\!]-\infty,0]\!]\right]
\end{aligned}
$$

l	$R^{\sharp\,0}_l$	$R^{\sharp\,1}_l$	$R^{\sharp\,2}_l$
0	$(\bot,\bot)$	$(\top,\top)$	
1	$(\bot,\bot)$	$([\![0,12]\!]\,,\top)$	
2	$(\bot,\bot)$	$([\![0,12]\!]\,,\{\,42\,\})$	
3	$(\bot,\bot)$	$([\![1,12]\!]\,,\{\,42\,\})$	
4	$(\bot,\bot)$	$([\![-1,10]\!]\,,\{\,42\,\})$	
5	$(\bot,\bot)$		

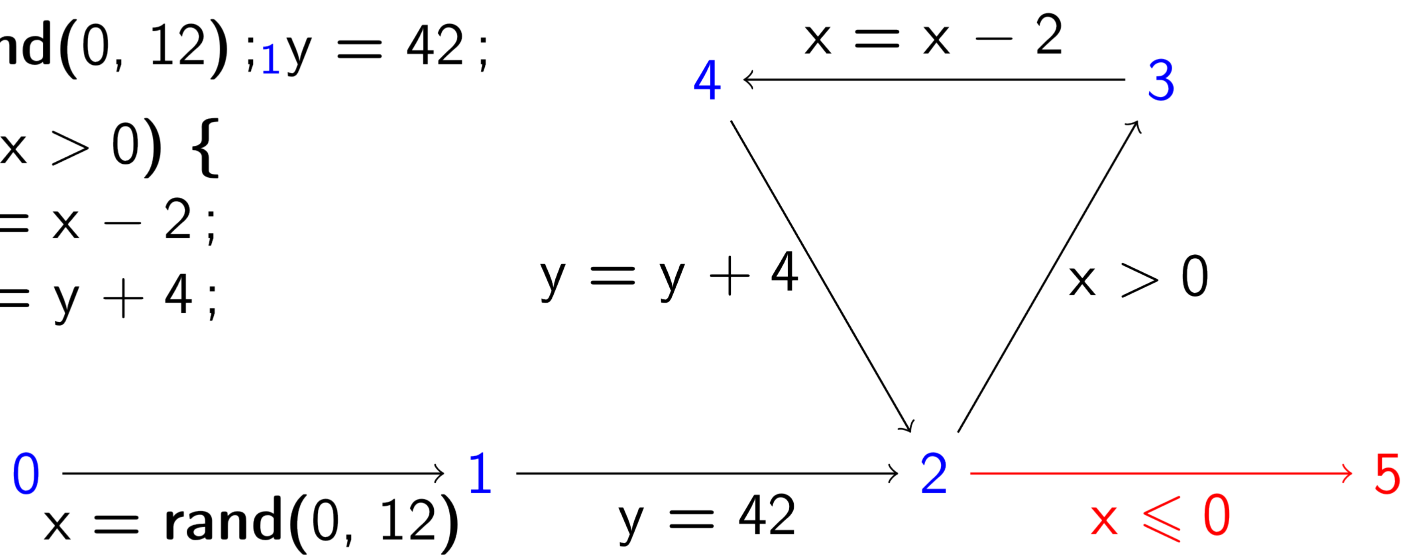
$_0$x = **rand**(0, 12) ;$_1$y = 42 ;

while $_2$(x > 0) {

  $_3$x = x − 2 ;

  $_4$y = y + 4 ;

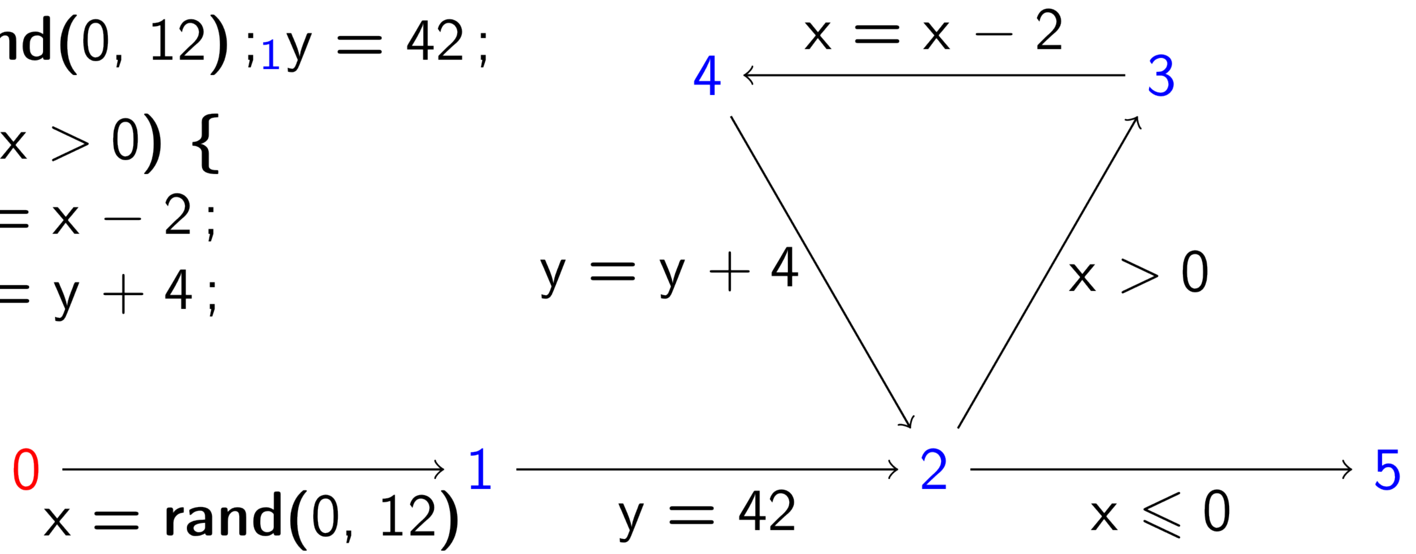
}$_5$



$$
\begin{aligned}
R_0^{\sharp\,i+1} &= \top \\
R_1^{\sharp\,i+1} &= R_0^{\sharp\,i+1}\,[x \mapsto [\![0, 12]\!]] \\
R_2^{\sharp\,i+1} &= R_1^{\sharp\,i+1}\,[y \mapsto [\![42, 42]\!]] \sqcup^{\sharp}_{\mathrm{nr}} \\
&\quad R_4^{\sharp\,i}\left[y \mapsto R_4^{\sharp\,i}(y) +^{\sharp} [\![4, 4]\!]\right] \\
R_3^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \right.\\
&\qquad\qquad \left. \sqcap^{\sharp} [\![1, +\infty[\![\right] \\
R_4^{\sharp\,i+1} &= R_3^{\sharp\,i+1}\left[x \mapsto R_3^{\sharp\,i+1}(x) -^{\sharp} [\![2, 2]\!]\right] \\
R_5^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \right.\\
&\qquad\qquad \left. \sqcap^{\sharp}]\!]-\infty, 0]\!] \right]
\end{aligned}
$$

l	$R_l^{\sharp\,0}$	$R_l^{\sharp\,1}$	$R_l^{\sharp\,2}$
0	$(\bot, \bot)$	$(\top, \top)$	
1	$(\bot, \bot)$	$([\![0, 12]\!], \top)$	
2	$(\bot, \bot)$	$([\![0, 12]\!], \{\,42\,\})$	
3	$(\bot, \bot)$	$([\![1, 12]\!], \{\,42\,\})$	
4	$(\bot, \bot)$	$([\![-1, 10]\!], \{\,42\,\})$	
5	$(\bot, \bot)$	$(\{\,0\,\}, \{\,42\,\})$	

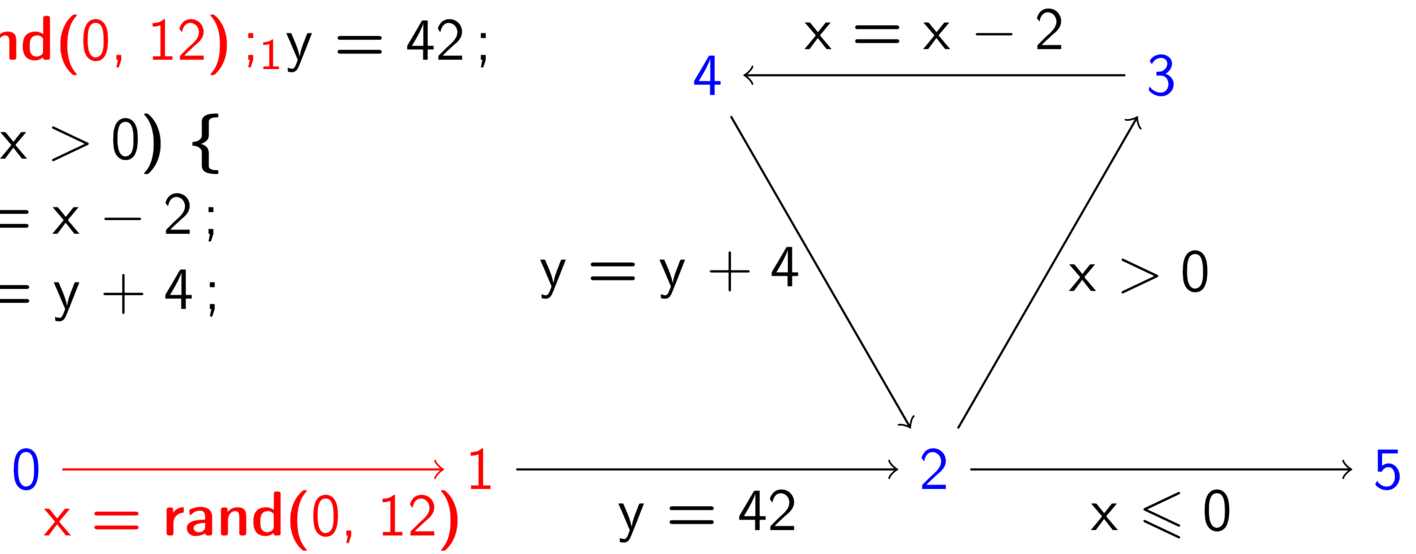
$_0$x = **rand**(0, 12) ;$_1$y = 42 ;
while $_2$(x > 0) {
    $_3$x = x − 2 ;
    $_4$y = y + 4 ;
}$_5$



$$
\begin{aligned}
{R_0^\sharp}^{i+1} &= \top\\
{R_1^\sharp}^{i+1} &= {R_0^\sharp}^{i+1}\left[x \mapsto [\![0,12]\!]\right]\\
{R_2^\sharp}^{i+1} &= {R_1^\sharp}^{i+1}\left[y \mapsto [\![42,42]\!]\right] \sqcup^\sharp_{\mathrm{nr}}\\
&\quad {R_4^\sharp}^{i}\left[y \mapsto {R_4^\sharp}^{i}(y) +^\sharp [\![4,4]\!]\right]\\
{R_3^\sharp}^{i+1} &= {R_2^\sharp}^{i+1}\left[x \mapsto {R_2^\sharp}^{i+1}(x)\right.\\
&\qquad\qquad \left.\sqcap^\sharp [\![1,+\infty[\![\right]\\
{R_4^\sharp}^{i+1} &= {R_3^\sharp}^{i+1}\left[x \mapsto {R_3^\sharp}^{i+1}(x) -^\sharp [\![2,2]\!]\right]\\
{R_5^\sharp}^{i+1} &= {R_2^\sharp}^{i+1}\left[x \mapsto {R_2^\sharp}^{i+1}(x)\right.\\
&\qquad\qquad \left.\sqcap^\sharp]\!]-\infty,0]\!]\right]
\end{aligned}
$$

l	${R_l^\sharp}^0$	${R_l^\sharp}^1$	${R_l^\sharp}^2$
0	$(\bot,\bot)$	$(\top,\top)$	$(\top,\top)$
1	$(\bot,\bot)$	$([\![0,12]\!],\top)$	
2	$(\bot,\bot)$	$([\![0,12]\!],\{42\})$	
3	$(\bot,\bot)$	$([\![1,12]\!],\{42\})$	
4	$(\bot,\bot)$	$([\![-1,10]\!],\{42\})$	
5	$(\bot,\bot)$	$(\{0\},\{42\})$	

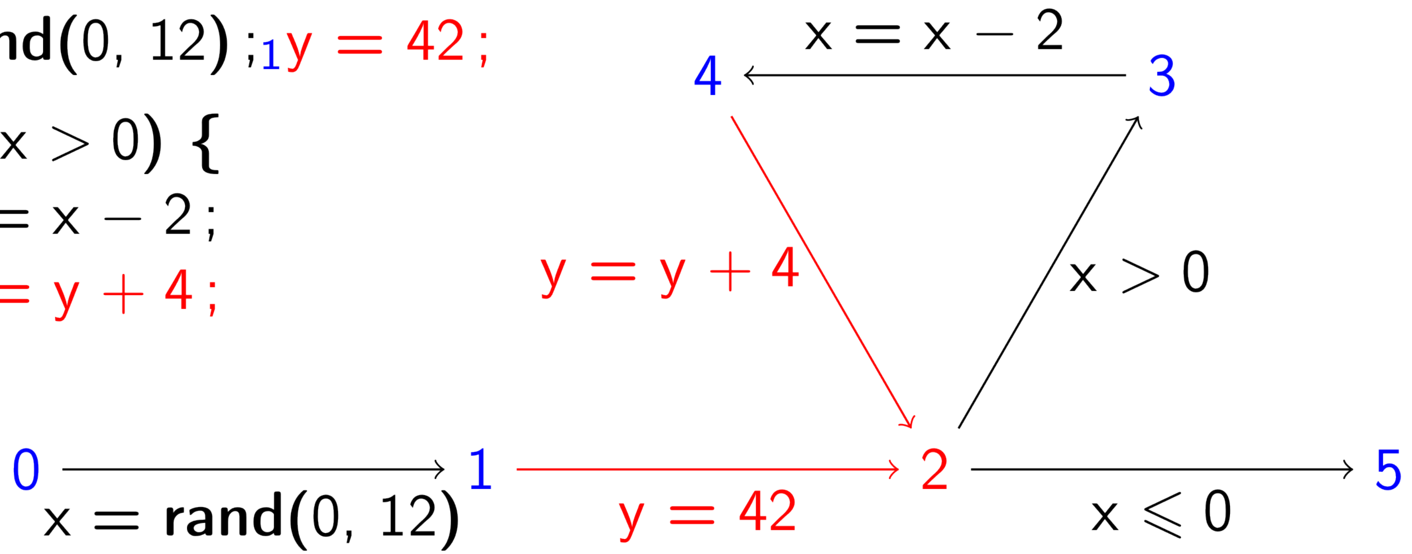
```
₀x = rand(0, 12) ;₁y = 42 ;
while ₂(x > 0) {
    ₃x = x − 2 ;
    ₄y = y + 4 ;
}₅
```



$$
\begin{aligned}
R_0^{\sharp\,i+1} &= \top\\
R_1^{\sharp\,i+1} &= R_0^{\sharp\,i+1}\,[x \mapsto [\![0,12]\!]]\\
R_2^{\sharp\,i+1} &= R_1^{\sharp\,i+1}\,[y \mapsto [\![42,42]\!]] \sqcup^{\sharp}_{\mathrm{nr}}\\
&\quad R_4^{\sharp\,i}\left[y \mapsto R_4^{\sharp\,i}(y) +^{\sharp} [\![4,4]\!]\right]\\
R_3^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x)\right.\\
&\qquad\qquad \left.\sqcap^{\sharp} [\![1,+\infty[\![\right]\\
R_4^{\sharp\,i+1} &= R_3^{\sharp\,i+1}\left[x \mapsto R_3^{\sharp\,i+1}(x) -^{\sharp} [\![2,2]\!]\right]\\
R_5^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x)\right.\\
&\qquad\qquad \left.\sqcap^{\sharp}]\!]-\infty,0]\!]\right]
\end{aligned}
$$

l	$R_l^{\sharp\,0}$	$R_l^{\sharp\,1}$	$R_l^{\sharp\,2}$
0	$(\bot,\bot)$	$(\top,\top)$	$(\top,\top)$
1	$(\bot,\bot)$	$([\![0,12]\!],\top)$	$([\![0,12]\!],\top)$
2	$(\bot,\bot)$	$([\![0,12]\!],\{42\})$	
3	$(\bot,\bot)$	$([\![1,12]\!],\{42\})$	
4	$(\bot,\bot)$	$([\![-1,10]\!],\{42\})$	
5	$(\bot,\bot)$	$(\{0\},\{42\})$	

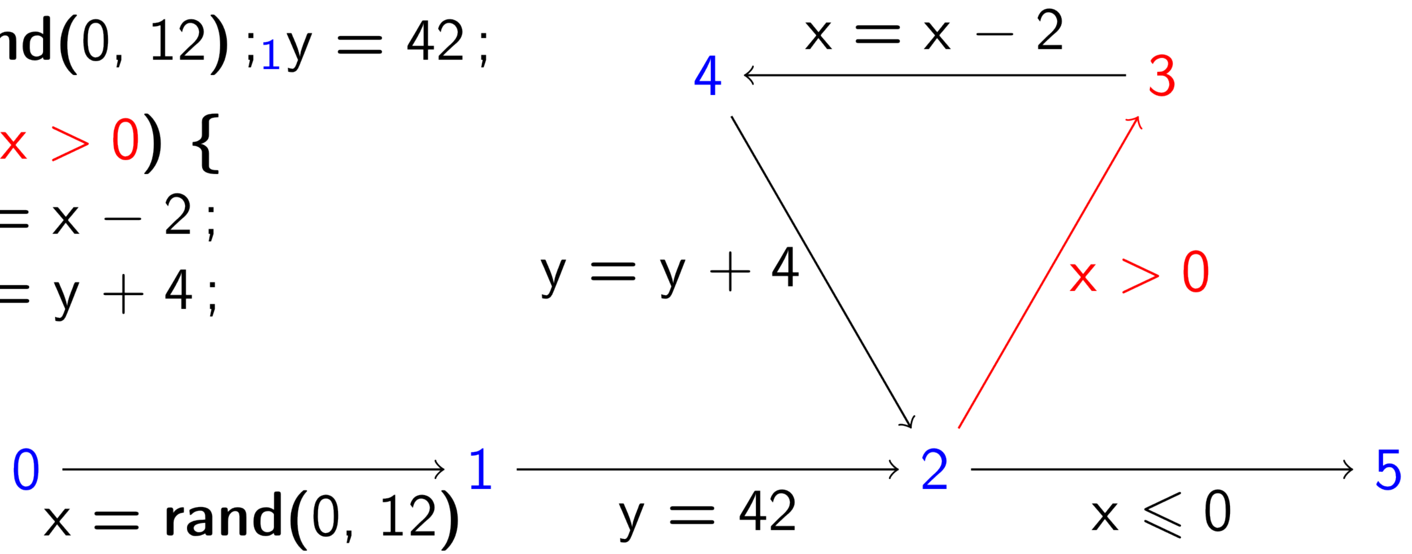
$_0$x = **rand**(0, 12) ;$_1$y = 42 ;
while $_2$(x > 0) {
  $_3$x = x − 2 ;
  $_4$y = y + 4 ;
}$_5$



$$
\begin{aligned}
R_0^{\sharp\,i+1} &= \top \\
R_1^{\sharp\,i+1} &= R_0^{\sharp\,i+1}\,[x \mapsto [\![0,12]\!]] \\
R_2^{\sharp\,i+1} &= R_1^{\sharp\,i+1}\,[y \mapsto [\![42,42]\!]] \sqcup^{\sharp}_{\mathrm{nr}} \\
&\quad R_4^{\sharp\,i}\left[y \mapsto R_4^{\sharp\,i}(y) +^{\sharp} [\![4,4]\!]\right] \\
R_3^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \sqcap^{\sharp} [\![1,+\infty[\![\right] \\
R_4^{\sharp\,i+1} &= R_3^{\sharp\,i+1}\left[x \mapsto R_3^{\sharp\,i+1}(x) -^{\sharp} [\![2,2]\!]\right] \\
R_5^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \sqcap^{\sharp}]\!]-\infty,0]\!]\right]
\end{aligned}
$$

l	$R_l^{\sharp\,0}$	$R_l^{\sharp\,1}$	$R_l^{\sharp\,2}$
0	$(\bot,\bot)$	$(\top,\top)$	$(\top,\top)$
1	$(\bot,\bot)$	$([\![0,12]\!],\top)$	$([\![0,12]\!],\top)$
2	$(\bot,\bot)$	$([\![0,12]\!],\{42\})$	$([\![-1,12]\!],[\![42,46]\!])$
3	$(\bot,\bot)$	$([\![1,12]\!],\{42\})$	
4	$(\bot,\bot)$	$([\![-1,10]\!],\{42\})$	
5	$(\bot,\bot)$	$(\{0\},\{42\})$	

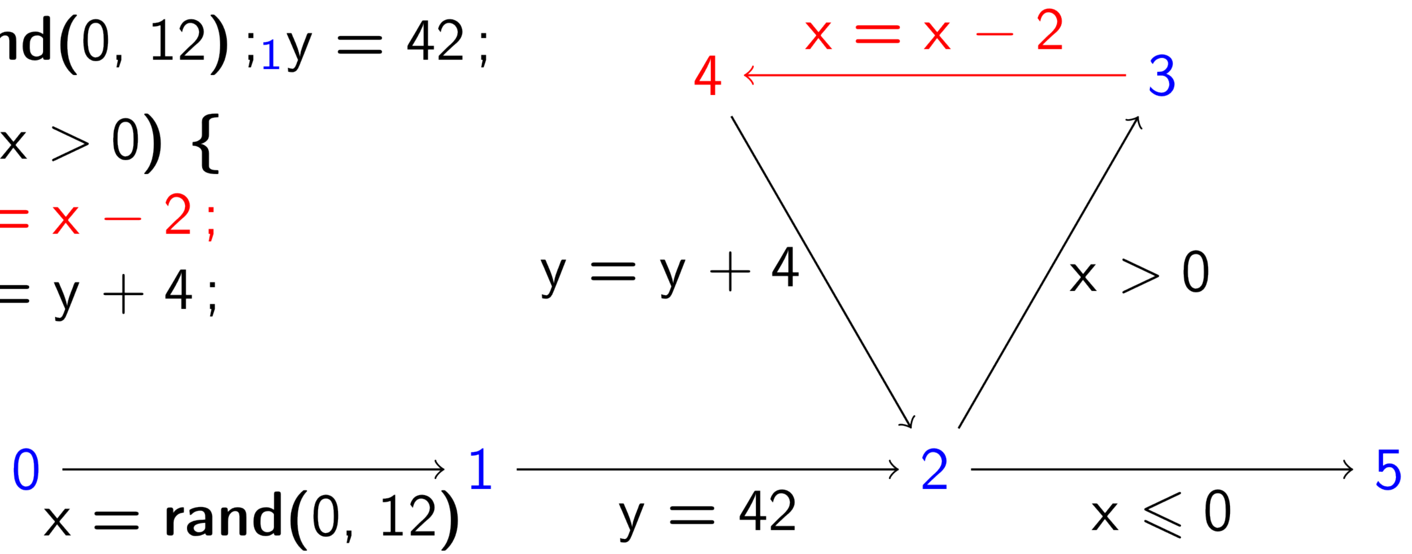
```
₀x = rand(0, 12) ;₁y = 42 ;
while ₂(x > 0) {
    ₃x = x − 2 ;
    ₄y = y + 4 ;
}₅
```



$$
\begin{aligned}
R_0^{\sharp\,i+1} &= \top \\
R_1^{\sharp\,i+1} &= R_0^{\sharp\,i+1}\,[x \mapsto [\![0, 12]\!]] \\
R_2^{\sharp\,i+1} &= R_1^{\sharp\,i+1}\,[y \mapsto [\![42, 42]\!]] \sqcup^{\sharp}_{\mathrm{nr}} \\
&\quad R_4^{\sharp\,i}\left[y \mapsto R_4^{\sharp\,i}(y) +^{\sharp} [\![4, 4]\!]\right] \\
R_3^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \right. \\
&\qquad\qquad \left. \sqcap^{\sharp} [\![1, +\infty[\![\right] \\
R_4^{\sharp\,i+1} &= R_3^{\sharp\,i+1}\left[x \mapsto R_3^{\sharp\,i+1}(x) -^{\sharp} [\![2, 2]\!]\right] \\
R_5^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \right. \\
&\qquad\qquad \left. \sqcap^{\sharp}]\!]-\infty, 0]\!]\right]
\end{aligned}
$$

l	$R_l^{\sharp\,0}$	$R_l^{\sharp\,1}$	$R_l^{\sharp\,2}$
0	$(\bot, \bot)$	$(\top, \top)$	$(\top, \top)$
1	$(\bot, \bot)$	$([\![0, 12]\!], \top)$	$([\![0, 12]\!], \top)$
2	$(\bot, \bot)$	$([\![0, 12]\!], \{ 42 \})$	$([\![-1, 12]\!], [\![42, 46]\!])$
3	$(\bot, \bot)$	$([\![1, 12]\!], \{ 42 \})$	$([\![1, 12]\!], [\![42, 46]\!])$
4	$(\bot, \bot)$	$([\![-1, 10]\!], \{ 42 \})$	
5	$(\bot, \bot)$	$(\{ 0 \}, \{ 42 \})$	

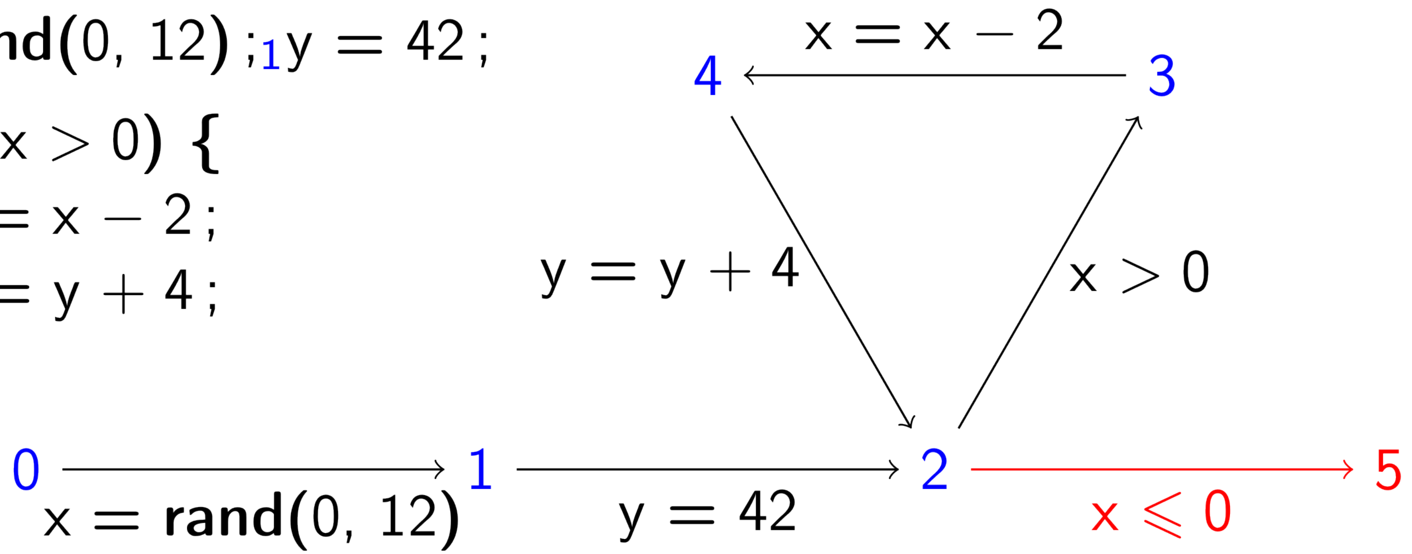
```
₀x = rand(0, 12) ;₁y = 42 ;
while ₂(x > 0) {
    ₃x = x − 2 ;
    ₄y = y + 4 ;
}₅
```



$$
\begin{aligned}
R_0^{\sharp\,i+1} &= \top \\
R_1^{\sharp\,i+1} &= R_0^{\sharp\,i+1}\,[x \mapsto [\![0,12]\!]] \\
R_2^{\sharp\,i+1} &= R_1^{\sharp\,i+1}\,[y \mapsto [\![42,42]\!]] \sqcup^{\sharp}_{\mathrm{nr}} \\
& \quad R_4^{\sharp\,i}\left[y \mapsto R_4^{\sharp\,i}(y) +^{\sharp} [\![4,4]\!]\right] \\
R_3^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \right. \\
& \qquad \left. \sqcap^{\sharp} [\![1,+\infty[\![\right] \\
{\color{red}R_4^{\sharp\,i+1}} &{\color{red}= R_3^{\sharp\,i+1}\left[x \mapsto R_3^{\sharp\,i+1}(x) -^{\sharp} [\![2,2]\!]\right]} \\
R_5^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \right. \\
& \qquad \left. \sqcap^{\sharp}]\!]-\infty,0]\!]\right]
\end{aligned}
$$

l	$R_l^{\sharp\,0}$	$R_l^{\sharp\,1}$	$R_l^{\sharp\,2}$
0	$(\bot,\bot)$	$(\top,\top)$	$(\top,\top)$
1	$(\bot,\bot)$	$([\![0,12]\!],\top)$	$([\![0,12]\!],\top)$
2	$(\bot,\bot)$	$([\![0,12]\!],\{42\})$	$([\![-1,12]\!],[\![42,46]\!])$
3	$(\bot,\bot)$	$([\![1,12]\!],\{42\})$	$([\![1,12]\!],[\![42,46]\!])$
4	$(\bot,\bot)$	$([\![-1,10]\!],\{42\})$	${\color{red}([\![-1,10]\!],[\![42,46]\!])}$
5	$(\bot,\bot)$	$(\{0\},\{42\})$	

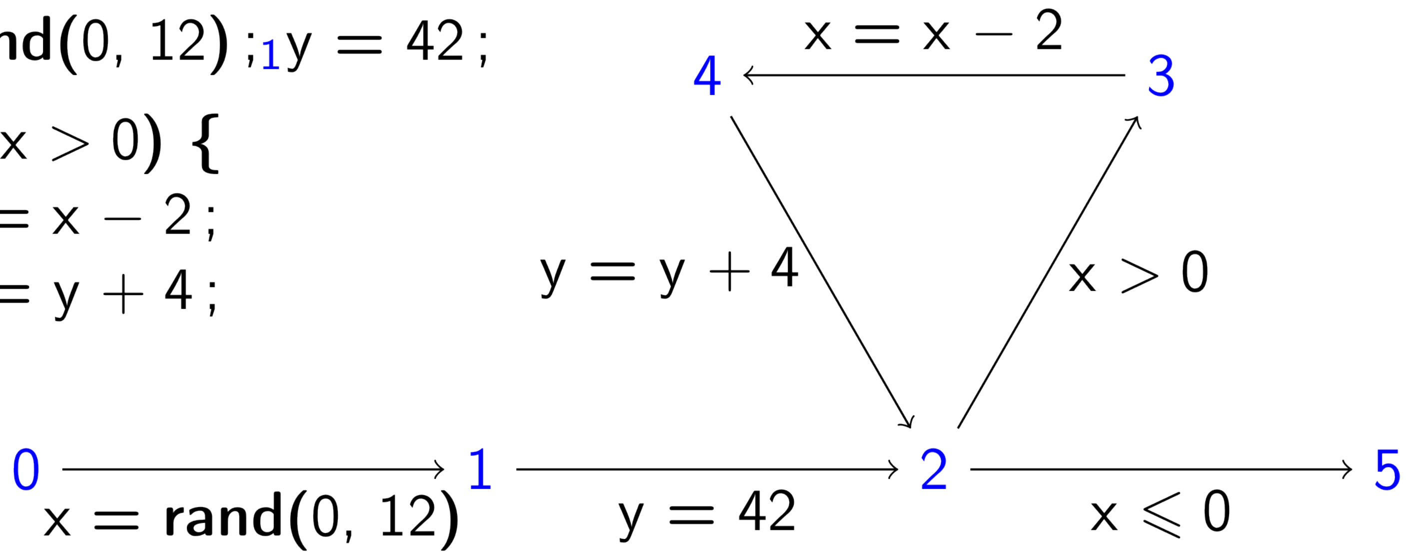
$_0$x = **rand**(0, 12) ;$_1$y = 42 ;
while $_2$(x > 0) {
    $_3$x = x − 2 ;
    $_4$y = y + 4 ;
}$_5$



$$
\begin{aligned}
R_0^{\sharp\,i+1} &= \top \\
R_1^{\sharp\,i+1} &= R_0^{\sharp\,i+1}\,[x \mapsto [\![0,12]\!]] \\
R_2^{\sharp\,i+1} &= R_1^{\sharp\,i+1}\,[y \mapsto [\![42,42]\!]] \sqcup^\sharp_{\mathrm{nr}} \\
&\quad R_4^{\sharp\,i}\left[y \mapsto R_4^{\sharp\,i}(y) +^\sharp [\![4,4]\!]\right] \\
R_3^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \right. \\
&\qquad\qquad \left. \sqcap^\sharp [\![1,+\infty[\![\right] \\
R_4^{\sharp\,i+1} &= R_3^{\sharp\,i+1}\left[x \mapsto R_3^{\sharp\,i+1}(x) -^\sharp [\![2,2]\!]\right] \\
R_5^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \right. \\
&\qquad\qquad \left. \sqcap^\sharp]\!]-\infty,0]\!]\right]
\end{aligned}
$$

l	$R_l^{\sharp\,0}$	$R_l^{\sharp\,1}$	$R_l^{\sharp\,2}$
0	$(\bot,\bot)$	$(\top,\top)$	$(\top,\top)$
1	$(\bot,\bot)$	$([\![0,12]\!],\top)$	$([\![0,12]\!],\top)$
2	$(\bot,\bot)$	$([\![0,12]\!],\{42\})$	$([\![-1,12]\!],[\![42,46]\!])$
3	$(\bot,\bot)$	$([\![1,12]\!],\{42\})$	$([\![1,12]\!],[\![42,46]\!])$
4	$(\bot,\bot)$	$([\![-1,10]\!],\{42\})$	$([\![-1,10]\!],[\![42,46]\!])$
5	$(\bot,\bot)$	$(\{0\},\{42\})$	$([\![-1,0]\!],[\![42,46]\!])$

$_0$x = **rand**(0, 12) ; $_1$y = 42 ;
while $_2$(x > 0) {
  $_3$x = x − 2 ;
  $_4$y = y + 4 ;
}$_5$



$$
\begin{aligned}
R_0^{\sharp\,i+1} &= \top \\
R_1^{\sharp\,i+1} &= R_0^{\sharp\,i+1}\,[x \mapsto \llbracket 0, 12 \rrbracket] \\
R_2^{\sharp\,i+1} &= R_1^{\sharp\,i+1}\,[y \mapsto \llbracket 42, 42 \rrbracket] \sqcup^{\sharp}_{\mathrm{nr}} \\
&\quad R_4^{\sharp\,i}\left[y \mapsto R_4^{\sharp\,i}(y) +^{\sharp} \llbracket 4, 4 \rrbracket\right] \\
R_3^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \sqcap^{\sharp} \llbracket 1, +\infty \llbracket\right] \\
R_4^{\sharp\,i+1} &= R_3^{\sharp\,i+1}\left[x \mapsto R_3^{\sharp\,i+1}(x) -^{\sharp} \llbracket 2, 2 \rrbracket\right] \\
R_5^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \sqcap^{\sharp} \rrbracket -\infty, 0 \rrbracket\right]
\end{aligned}
$$

l	$R_l^{\sharp\,0}$	$R_l^{\sharp\,1}$	$R_l^{\sharp\,2}$
0	$(\bot, \bot)$	$(\top, \top)$	$(\top, \top)$
1	$(\bot, \bot)$	$(\llbracket 0, 12 \rrbracket, \top)$	$(\llbracket 0, 12 \rrbracket, \top)$
2	$(\bot, \bot)$	$(\llbracket 0, 12 \rrbracket, \{ 42 \})$	$(\llbracket -1, 12 \rrbracket, \llbracket 42, 46 \rrbracket)$
3	$(\bot, \bot)$	$(\llbracket 1, 12 \rrbracket, \{ 42 \})$	$(\llbracket 1, 12 \rrbracket, \llbracket 42, 46 \rrbracket)$
4	$(\bot, \bot)$	$(\llbracket -1, 10 \rrbracket, \{ 42 \})$	$(\llbracket -1, 10 \rrbracket, \llbracket 42, 46 \rrbracket)$
5	$(\bot, \bot)$	$(\{ 0 \}, \{ 42 \})$	$(\llbracket -1, 0 \rrbracket, \llbracket 42, 46 \rrbracket)$

The fixpoint is still far!

Correctness

The **abstract** semantic is an over-approximation of the **concrete** semantic.

For all ***l*** ⊆ ***L*** :

$$R_l \subseteq \gamma_{\mathrm{nr}}\left(R_l^\sharp\right)$$

Termination

- Problem: This algorithm might not terminate!

Termination

- Problem: This algorithm might not terminate!

- The sequence $(\boldsymbol{F}^n(\perp))_{n \in \mathbb{N}}$ might not converge

Termination

- Problem: This algorithm might not terminate!
- The sequence $(\boldsymbol{F}^n(\perp))_{n \in \mathbb{N}}$ might not converge
- The sequence is increasing because ***F*** is monotonic

Termination

- Problem: This algorithm might not terminate!
- The sequence $(\boldsymbol{F}^{n}(\bot))_{n \in \mathbb{N}}$ might not converge
- The sequence is increasing because ***F*** is monotonic
- If $\boldsymbol{D}^{\#}$ is finite, then this would converge

Termination

- Problem: This algorithm might not terminate!
- The sequence $(\boldsymbol{F}^n(\perp))_{n \in \mathbb{N}}$ might not converge
- The sequence is increasing because ***F*** is monotonic
- If $\boldsymbol{D}^\#$ is finite, then this would converge
- But intervals have **infinitely increasing sequences**: $([0, n])_{n \in N}$

Termination

- Problem: This algorithm might not terminate!
- The sequence $(\boldsymbol{F}^n(\perp))_{n \in \mathbb{N}}$ might not converge
- The sequence is increasing because ***F*** is monotonic
- If $\boldsymbol{D}^\#$ is finite, then this would converge
- But intervals have **infinitely increasing sequences**: $([0, n])_{n \in N}$
- Even if it terminates, it could be slow..

Convergence Acceleration

- We introduce a new operator called ***widening*** ∇
- The ***widening*** is responsible for breaking **infinitely increasing sequences**
- It performs a jump forward

Widening

A ***widening*** ∇ is a binary operator $\nabla : \mathcal{D}^\sharp \times \mathcal{D}^\sharp \to \mathcal{D}^\sharp$ such that:

- $\forall x^\sharp, y^\sharp, \quad x^\sharp \sqcup^\sharp y^\sharp \sqsubseteq^\sharp x^\sharp \nabla y^\sharp$
- For all sequences $\left(x_n^\sharp\right)_{n\in\mathbb{N}}$, the following sequence **converges**:

$$\begin{cases} y_0^\sharp & = & x_0^\sharp \\ y_{i+1}^\sharp & = & y_i^\sharp \nabla x_{i+1}^\sharp \end{cases}$$

$$R^{\sharp} = {F^{\sharp}}^{N}(\bot) = \operatorname{lfp} F^{\sharp}$$

$$\uparrow$$

$$\vdots$$

$$\uparrow$$

$$ {R^{\sharp}}^{2} = F^{\sharp}\left({R^{\sharp}}^{1}\right) = {F^{\sharp}}^{2}(\bot)$$

$$\uparrow$$

$$ {R^{\sharp}}^{1} = F^{\sharp}\left({R^{\sharp}}^{0}\right) = F^{\sharp}(\bot)$$

$$\uparrow$$

$$ {R^{\sharp}}^{0} = \bot$$

Without widening

$$
\begin{array}{c}
R^\sharp = {F^\sharp}^N(\perp) = \mathrm{lfp}\, F^\sharp \\
\uparrow \\
\vdots \\
\uparrow \\
{R^\sharp}^2 = F^\sharp\left({R^\sharp}^1\right) = {F^\sharp}^2(\perp) \\
\uparrow \\
{R^\sharp}^1 = F^\sharp\left({R^\sharp}^0\right) = F^\sharp(\perp) \\
\uparrow \\
{R^\sharp}^0 = \perp
\end{array}
$$

Without widening

$$
\begin{array}{c}
R^\sharp = R^\sharp \triangledown F^\sharp(R^\sharp) \\
\mathrm{lfp}\, F^\sharp \\
\vdots \\
{R^\sharp}^2 = {R^\sharp}^1 \triangledown F^\sharp\left({R^\sharp}^1\right) \\
{R^\sharp}^1 = {R^\sharp}^0 \triangledown F^\sharp\left({R^\sharp}^0\right) \\
{R^\sharp}^0 = \perp
\end{array}
$$

With widening

Iterations with widening

- It is still a fixpoint: $R^\sharp = R^\sharp \triangledown F^\sharp(R^\sharp)$ thus $F^\sharp(R^\sharp) \sqsubseteq^\sharp R^\sharp$

Iterations with widening

- It is still a fixpoint: $R^\sharp = R^\sharp \nabla F^\sharp(R^\sharp)$ thus $F^\sharp(R^\sharp) \sqsubseteq^\sharp R^\sharp$
- It is not the ***least fixed point***

Iterations with widening

- It is still a fixpoint: $R^\sharp = R^\sharp \nabla F^\sharp(R^\sharp)$ thus $F^\sharp(R^\sharp) \sqsubseteq^\sharp R^\sharp$
- It is not the ***least fixed point***
- It includes the ***least fixed point***: $\mathrm{lfp}\ F^\sharp \sqsubseteq^\sharp R^\sharp$

Interval widening

$$x^\sharp \nabla y^\sharp = \begin{cases} [\![a, b]\!] & \text{si } x^\sharp = [\![a, b]\!]\,, y^\sharp = [\![c, d]\!]\,, c \geqslant a, d \leqslant b \\ [\![a, +\infty[\![& \text{si } x^\sharp = [\![a, b]\!]\,, y^\sharp = [\![c, d]\!]\,, c \geqslant a, d > b \\ ]\!]-\infty, b]\!] & \text{si } x^\sharp = [\![a, b]\!]\,, y^\sharp = [\![c, d]\!]\,, c < a, d \leqslant b \\ ]\!]-\infty, +\infty[\![& \text{si } x^\sharp = [\![a, b]\!]\,, y^\sharp = [\![c, d]\!]\,, c < a, d > b \\ y^\sharp & \text{si } x^\sharp = \perp \\ x^\sharp & \text{si } y^\sharp = \perp \end{cases}$$

Interval widening

$$x^\sharp \triangledown y^\sharp = \begin{cases} [\![a, b]\!] & \text{si } x^\sharp = [\![a, b]\!]\,, y^\sharp = [\![c, d]\!]\,, c \geqslant a, d \leqslant b \\ [\![a, +\infty[\![& \text{si } x^\sharp = [\![a, b]\!]\,, y^\sharp = [\![c, d]\!]\,, c \geqslant a, d > b \\ ]\!]-\infty, b]\!] & \text{si } x^\sharp = [\![a, b]\!]\,, y^\sharp = [\![c, d]\!]\,, c < a, d \leqslant b \\ ]\!]-\infty, +\infty[\![& \text{si } x^\sharp = [\![a, b]\!]\,, y^\sharp = [\![c, d]\!]\,, c < a, d > b \\ y^\sharp & \text{si } x^\sharp = \bot \\ x^\sharp & \text{si } y^\sharp = \bot \end{cases}$$

$$[\![0, 1]\!] \triangledown [\![0, 2]\!] = [\![0, +\infty[\![$$

$$[\![0, 2]\!] \triangledown [\![0, 1]\!] = [\![0, 2]\!]$$

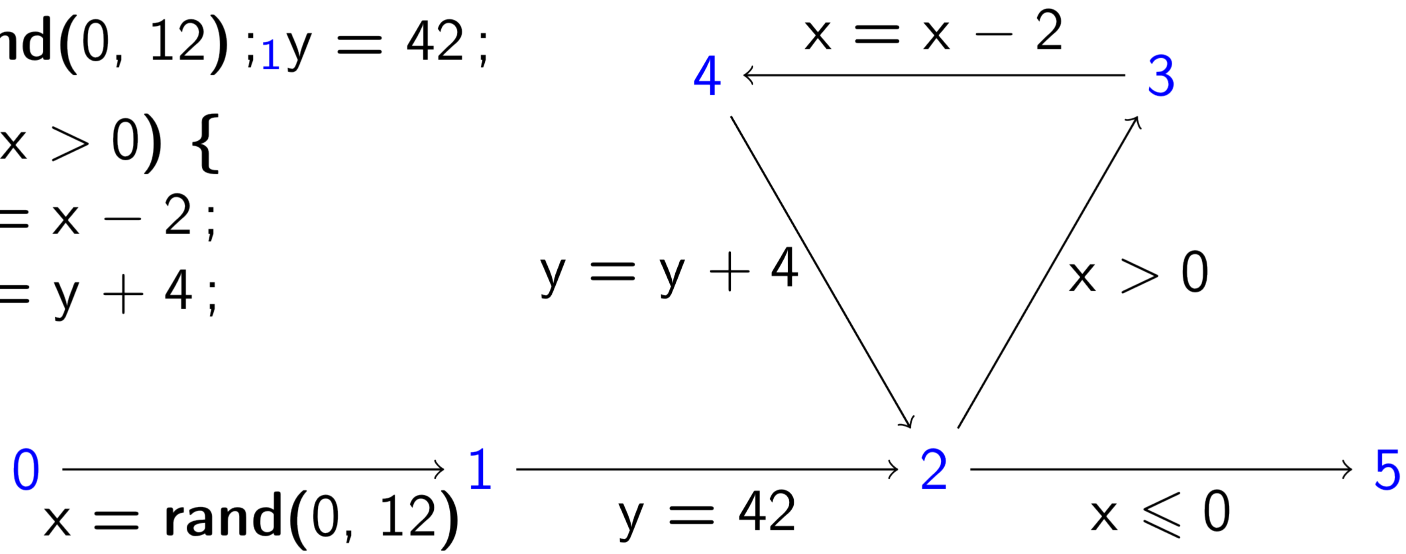
$_0\mathsf{x} = \mathbf{rand}(0,\ 12)\ ;_1\mathsf{y} = 42\ ;$

$\mathbf{while}\ {_2}(\mathsf{x} > 0)\ \{$

$\quad {_3}\mathsf{x} = \mathsf{x} - 2\ ;$

$\quad {_4}\mathsf{y} = \mathsf{y} + 4\ ;$

$\}_5$



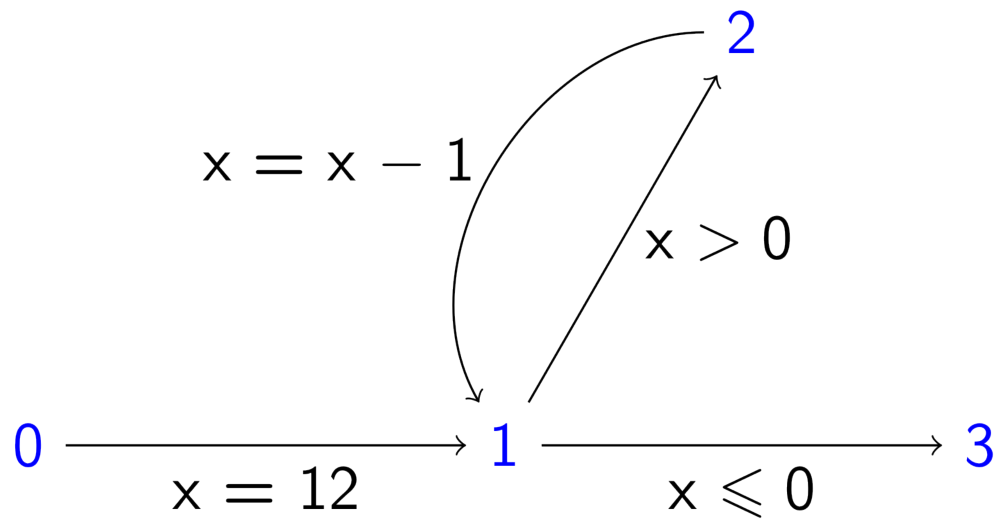
$$
\begin{aligned}
R_0^{\sharp\,i+1} &= \top \\
R_1^{\sharp\,i+1} &= R_0^{\sharp\,i+1}\,[x \mapsto [\![0, 12]\!]] \\
R_2^{\sharp\,i+1} &= R_1^{\sharp\,i+1}\,[y \mapsto [\![42, 42]\!]] \sqcup^\sharp_{\mathrm{nr}} \\
&\quad R_4^{\sharp\,i}\left[y \mapsto R_4^{\sharp\,i}(y) +^\sharp [\![4, 4]\!]\right] \\
R_3^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \right. \\
&\qquad\qquad \left. \sqcap^\sharp [\![1, +\infty[\![\right] \\
R_4^{\sharp\,i+1} &= R_3^{\sharp\,i+1}\left[x \mapsto R_3^{\sharp\,i+1}(x) -^\sharp [\![2, 2]\!]\right] \\
R_5^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \right. \\
&\qquad\qquad \left. \sqcap^\sharp]\!]-\infty, 0]\!]\right]
\end{aligned}
$$

$$
\begin{aligned}
R_0^\sharp &= \top_{\mathrm{nr}} \\
R_1^\sharp &= ([\![0, 12]\!]\,,\top) \\
R_2^\sharp &= (]\!]-\infty, 12]\!]\,, [\![42, +\infty[\![) \\
R_3^\sharp &= ([\![1, 12]\!]\,, [\![42, +\infty[\![) \\
R_4^\sharp &= ([\![-1, 10]\!]\,, [\![46, +\infty[\![) \\
R_5^\sharp &= (]\!]-\infty, 0]\!]\,, [\![42, +\infty[\![)
\end{aligned}
$$

Widening

- Widening allows the algorithm to terminate quickly
- But it might cause a loss of precision
- In practice, we only use the widening after a few iterations
- We can also use a widening with a threshold

```
0x = 12;
while 1(x > 0) {
    2 x = x - 1;
}3
```



$$
\begin{aligned}
{R_0^\sharp}^{i+1} &= \top \\
{R_1^\sharp}^{i+1} &= {R_1^\sharp}^{i} \mathbin{\color{red}{\triangledown_{\mathrm{nr}}}} \Big({R_0^\sharp}^{i+1} \left[x \mapsto [\![12, 12]\!]\right] \sqcup^\sharp_{\mathrm{nr}} \\
&\qquad {R_2^\sharp}^{i} \left[y \mapsto {R_2^\sharp}^{i}(x) -^\sharp [\![1, 1]\!] \right] \Big) \\
{R_3^\sharp}^{i+1} &= {R_1^\sharp}^{i+1} \Big[x \mapsto {R_1^\sharp}^{i+1}(x) \\
&\qquad \sqcap^\sharp \,]\!]{-\infty}, 0]\!] \Big]
\end{aligned}
$$

l	${R_l^\sharp}^0$	${R_l^\sharp}^1$	${R_l^\sharp}^2$	${R_l^\sharp}^3$
0	$\bot$	$\top$	$\top$	$\top$
1	$\bot$	$[\![12, 12]\!]$	$]\!]{-\infty}, 12]\!]$	$]\!]{-\infty}, 12]\!]$
2	$\bot$	$[\![12, 12]\!]$	$[\![1, 12]\!]$	$[\![1, 12]\!]$
3	$\bot$	$\bot$	$]\!]{-\infty}, 0]\!]$	$]\!]\color{red}{-\infty}, 0]\!]$

Regain precision

- We introduce a new operator called narrowing Δ
- Perform decreasing iterations to regain precision

Narrowing

A ***narrowing*** $\triangle$ is a binary operator $\triangle : \mathcal{D}^\sharp \times \mathcal{D}^\sharp \to \mathcal{D}^\sharp$ such that:

- $\forall x^\sharp, y^\sharp, \quad x^\sharp \sqcap^\sharp y^\sharp \sqsubseteq^\sharp x^\sharp \triangle y^\sharp \sqsubseteq^\sharp x^\sharp$
- For all sequences $\left(x_n^\sharp\right)_{n\in\mathbb{N}}$, the following sequence **converges**:

$$\begin{cases} y_0^\sharp & = & x_0^\sharp \\ y_{i+1}^\sharp & = & y_i^\sharp \triangle x_{i+1}^\sharp \end{cases}$$

$$R^{\sharp} = R^{\sharp} \triangledown F^{\sharp}(R^{\sharp})$$

$${R^{\sharp\prime}}^{1} = R^{\sharp} \vartriangle F^{\sharp}(R^{\sharp})$$

$$\vdots$$

$$R^{\sharp\prime} = R^{\sharp\prime} \vartriangle F^{\sharp}\left(R^{\sharp\prime}\right)$$

$$\mathrm{lfp}\, F^{\sharp}$$

$$\vdots$$

$${R^{\sharp}}^{0} = \bot$$

$$R^\sharp = R^\sharp \triangledown F^\sharp(R^\sharp)$$

$$R^{\sharp'^{1}} = R^\sharp \vartriangle F^\sharp(R^\sharp)$$

$$\vdots$$

$$R^{\sharp'} = R^{\sharp'} \vartriangle F^\sharp\left(R^{\sharp'}\right)$$

$$\mathrm{lfp}\ F^\sharp \sqsubseteq^\sharp R^{\sharp'}$$

$$\mathrm{lfp}\ F^\sharp$$

$$\vdots$$

$$R^{\sharp^{0}} = \bot$$

Interval narrowing

$$x^\sharp \bigtriangleup y^\sharp = \begin{cases} [\![a, d]\!] & \text{si } x^\sharp = [\![a, +\infty[\![\,, y^\sharp = [\![c, d]\!] \\ [\![c, b]\!] & \text{si } x^\sharp =]\!]-\infty, b]\!]\,, y^\sharp = [\![c, d]\!] \\ [\![c, d]\!] & \text{si } x^\sharp =]\!]-\infty, +\infty[\![\,, y^\sharp = [\![c, d]\!] \\ x^\sharp & \text{sinon} \end{cases}$$

Interval narrowing

$$x^\sharp \vartriangle y^\sharp = \begin{cases} [\![a, d]\!] & \text{si } x^\sharp = [\![a, +\infty[\![\,, y^\sharp = [\![c, d]\!] \\ [\![c, b]\!] & \text{si } x^\sharp =]\!]-\infty, b]\!]\,, y^\sharp = [\![c, d]\!] \\ [\![c, d]\!] & \text{si } x^\sharp =]\!]-\infty, +\infty[\![\,, y^\sharp = [\![c, d]\!] \\ x^\sharp & \text{sinon} \end{cases}$$

$$[\![0, +\infty[\![\vartriangle [\![0, 1]\!] = [\![0, 1]\!]$$

$$[\![0, 2]\!] \vartriangle [\![0, 1]\!] = [\![0, 2]\!]$$

$_0$x = 12 ;
while $_1$(x > 0) **{**
    $_2$ x = x − 1 ;
}$_3$

$$\begin{tikzcd}
 & & 2 \arrow[dl, bend right, "\mathrm{x} = \mathrm{x} - 1"'] & \\
0 \arrow[r, "\mathrm{x} = 12"'] & 1 \arrow[ur, "\mathrm{x} > 0"'] \arrow[rr, "\mathrm{x} \leqslant 0"'] & & 3
\end{tikzcd}$$

$$\begin{aligned}
{R_0^\sharp}^{i+1} &= \top \\
{R_1^\sharp}^{i+1} &= {R_1^\sharp}^{i} \, \triangledown_{\mathrm{nr}} \Big({R_0^\sharp}^{i+1} \left[x \mapsto [\![12, 12]\!]\right] \sqcup^\sharp_{\mathrm{nr}} \\
&\qquad {R_2^\sharp}^{i} \left[y \mapsto {R_2^\sharp}^{i}(x) -^\sharp [\![1, 1]\!] \right] \Big) \\
{R_3^\sharp}^{i+1} &= {R_1^\sharp}^{i+1} \Big[x \mapsto {R_1^\sharp}^{i+1}(x) \\
&\qquad \sqcap^\sharp \,]\!]-\infty, 0]\!] \Big]
\end{aligned}$$

$$\begin{aligned}
R_0^\sharp &= \top_{\mathrm{nr}} \\
R_1^\sharp &= [\![0, 12]\!] \\
R_2^\sharp &= [\![1, 12]\!] \\
R_3^\sharp &= [\![0, 0]\!]
\end{aligned}$$

Abstract Domains

Domain	Constraints	Complexity
Interval	$x \in [a, b]$	n
Congruence	$x \in aZ+b$	n
Gauge	$x \in [a*i + b*k + \ldots, a'*i + b'*k + \ldots]$	K*n
Difference Bound Matrices	$x - y \in [a, b]$	n^3
Octagon	$x \pm y \in [a, b]$	n^3
Polyhedra	$a*x + b*y + \ldots + c <= 0$	Exponential

Thank you.

Questions?

maxime.arthaud@nasa.gov