Abstract interpretation for compilers

A journey from theory to practice back to ...

Laure Gonnord,
http://laure.gonnord.org/pro

PLISS 2022

Intro

Programs, compilers

The goal of a compiler writer is to bridge the gap between programming languages and the hardware ; hence, making programmers more productive

Image (and intro inspiration) by F. Pereira, UFMG, Brasil

Compiler Goals

Compiler goals :

- program optimization : register allocation, copy elimination, constant propagation . . .
- bug tracking : null pointer dereference, division by zero, illicit memory accesses (or prove the absence of bugs).

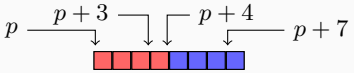
▶ **static analyses** (yes, there exist cool dynamic analyses, too !).

Goal : performance 1/2

Enable loop parallelism :

```
void fill_array (char *p){
  unsigned int i;
  for (i=0; i<4; i++)
       *(p + i) = 0 ;
  for (i=4; i<8; i++)
       *(p + i) = 2*i ;
}
```

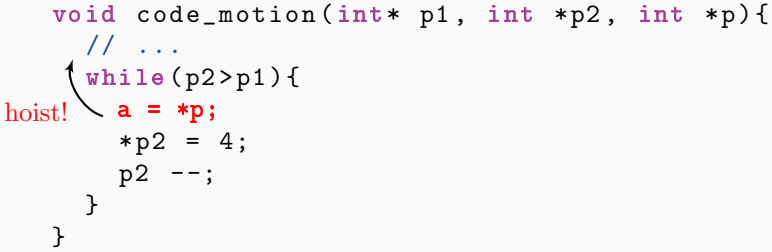
Parallel loops



- The two regions do not overlap.

Goal : performance 2/2

Enable code motion :



- If p and p_2 do not alias, then `a=*p` is invariant.
- Hoisting this instruction saves one `load` per loop.

Goal : safety

Prove that (some) memory accesses are safe (tabs.c) :

```
int main () {
  int v[10];
  v[0]=0; ✓
  return v[20]; ✗
}
```

- This program has an illegal array access.
- Fight against bugs and overflow attacks.

Goal : safety 2/2

Prove program correctness/absence of functional bug :

```
void find_mini (int a[N], int l, int u){
  unsigned int i=l;
  int b=a[l]
  while (i <= u){
    if(a[i]<b) b=a[i] ;
    i++ ;
  }
  // here b = min(a[l..u])
}
```

Invariants on “unbounded arrays” (“forall”).

Autopromo : `https://github.com/vaphor`

Some history

Some key dates in compiler history - 60's

- Frances E. Allen, working alone or jointly with John Cocke, introduced many of the concepts for optimization:
  - Control flow graphs
  - Many dataflow analyses
  - A description of many different program optimizations
  - Interprocedural dataflow analyses
  - Worklist algorithms
- A lot of these inventions and discoveries have been made in the IBM labs.

Slides courtesy of F. Pereira

Some key dates in compiler history - 70's

- Most of the compiler theory and technology in use today is based on the notion of the dataflow monotone framework.
  - Propagation of information
  - Iterative algorithms
  - Termination of fixed point computations
  - The meet over all paths solution to dataflow problems
- These ideas came, mostly, from the work of Gary Kildall, who is one of the fathers of the modern theory of code analysis and optimization.

The inventor of the BIOS system!

In addition of being paramount to the development of modern compiler theory, Gary Kildall used to host a talk show called "*The Computer Chronicles*". Nevertheless, he is mostly known for the deal with the Ms-DOS system that involved IBM and Bill Gates.

Some key dates in compiler history - end80s

- Once in a while we see smart ideas. Perhaps, the smartest idea in compiler optimization was the Static Single Assignment Form.
  - A program representation in which every variable has only one definition site.
- SSA form was introduced by Cytron *et al.*, in the late eighties♡ in IBM.
- There were many improvements since then, such as pruned SSA form, SSA-based register allocation, etc.
- The idea took off very quickly. Today almost every compiler uses this intermediate representation.

SSA Seminar: celebrated the 20th anniversary of the Static Single Assignment form, April 27-30, Autrans, France

♡: An Efficient Method of Computing Static Single Assignment Form

And other

- Register Allocation : from graph coloring (80's) to linear scan (99)
- Type theory : 2000's
- Mechanical proofs : Compcert, Twelf, Velvm 2010's and actual

A success story (77-...)

ABSTRACT INTERPRETATION : A UNIFIED LATTICE MODEL FOR STATIC ANALYSIS
OF PROGRAMS BY CONSTRUCTION OR APPROXIMATION OF FIXPOINTS

Patrick Cousot* and Radhia Cousot**

Laboratoire d'Informatique, U.S.M.G., BP. 53
38041 Grenoble cedex, France

1. Introduction

A program denotes computations in some universe of objects. Abstract interpretation of programs consists in using that denotation to describe computations in another universe of abstract objects,

Abstract program properties are modeled by a complete semilattice, Birkhoff[61]. Elementary program constructs are locally interpreted by order preserving functions which are used to associate a system of recursive equations with a program. The program global properties are then defined as one

A success story (77-. . .)

ABSTRACT INTERPRETATION : A UNIFIED LATTICE MODEL FOR STATI
OF PROGRAMS BY CONSTRUCTION OR APPROXIMATION

Patrick Cousot* and R

Laboratoire d'
380

1. Introd

universe of
of programs con-
tion to describe compu-
verse of abstract objects,

Abstract program properties are modeled by a complete semilattice, Birkhoff[61]. Elementary program constructs are locally interpreted by order preserving functions which are used to associate a system of recursive equations with a program. The program global properties are then defined as one

- In Nov. 2003, Astrée was able to prove completely automatically the absence of any RTE in the primary flight control software of the Airbus A340 fly-by-wire system, a program of 132,000 lines of C analyzed in 1^h20 on a 2.8 GHz 32-bit PC using 300 Mb of memory (and 50mn on a 64-bit AMD Athlon™ 64 using 580 Mb of memory).

A success story (77-. . .)

ABSTRACT INTERPRETATION : A UNIFIED LATTICE MODEL FOR STATI
OF PROGRAMS BY CONSTRUCTION OR APPROXIMATION
Patrick Cousot* and Ra
Laboratoire d'I
380

1. Introd

am properties are modeled by a com-
milattice, Birkhoff[61]. Elementary pro-
am constructs are locally interpreted by order
preserving functions which are used to associate
a system of recursive equations with a program. The
program global properties are then defined as one
jects,

- In Nov. 2003, Astrée was able to prove completely automatically the absence of any RTE in the primary flight control software of the Airbus A340 fly-by-wire system, a program of 132,000 lines of C analyzed in 1h20 on a 2.8 GHz 32-bit PC using 300 Mb of memory (and 50mn on a 64-bit AMD Athlon™ 64 using 580 Mb of memory).

Exploitation license of Astrée

Starting Dec. 2009, Astrée is available from AbsInt Angewandte Informatik AbsInt Angewandte Informatik (www.absint.de/astree/).

A success story (77-. . .)

ABSTRACT INTERPRETATION : A UNIFIED LATTICE MODEL FOR STATI
OF PROGRAMS BY CONSTRUCTION OR APPROXIMATION
Patrick Cousot* and Ra
Laboratoire d'I
380

1. Introd

am properties are modeled by a com-
milattice, Birkhoff[61]. Elementary pro-
am constructs are locally interpreted by order
preserving functions which are used to associate
a system of recursive equations with a program. The
program global properties are then defined as one
jects,

- In Nov. 2003, Astrée was able to prove completely automatically the absence of any RTE in the primary flight control software of the Airbus A340 fly-by-wire system, a program of 132,000 lines of C analyzed in 1h20 on a 2.8 GHz 32-bit PC using 300 Mb of memory (and 50mn on a 64-bit AMD Athlon™ 64 using 580 Mb of memory).

Exploitation license of Astrée

Starting Dec. 2009, Astrée is available from AbsInt Angewandte Informatik AbsInt Angewandte Informatik (www.absint.de/astree/).

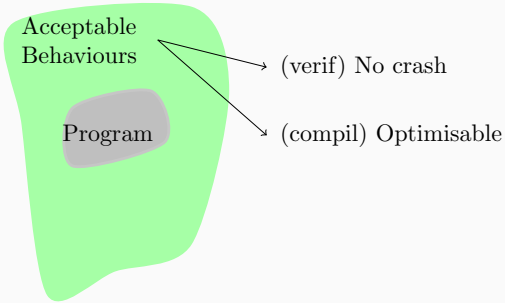
- And now in some production tools (Infer), . . .

Abstract Interpretation Genealogy - kind of partial

```
Patrick & Radhia Cousot ----- Nicolas Halbwachs ---- me --- Maroua Maalej
                        ----- ...
                        ----- David Monniaux
                        ----- Antoine Miné
                        ----- Xavier Rival
                        ----- ...
                        ----- Caterina Urban :-)
```

What we want : proving non trivial properties of software

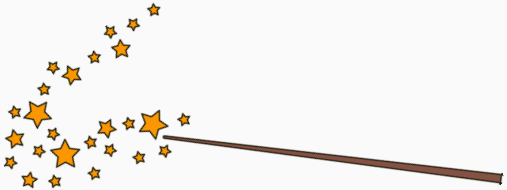
- Basic idea : software has **mathematically defined behaviour**



There is no free lunch

i.e. no magical static analyser. It is impossible to prove interesting properties :

- automatically
- exactly
- on unbounded programs

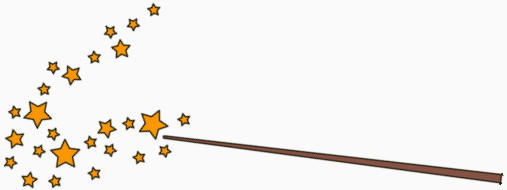


▶ **Dataflow/Abstract Interpretation** = conservative approximations.

There is no free lunch

i.e. no magical static analyser. It is ~~im~~ possible to prove interesting properties :

- automatically
- ~~exactly~~ with false positives !
- on unbounded programs



▶ **Dataflow/Abstract Interpretation** = conservative approximations.

Research questions

Question

How to design *static analyses* that are **correct by construction** ?

- From dataflow analyses to abstract interpretation (course 1)

Question

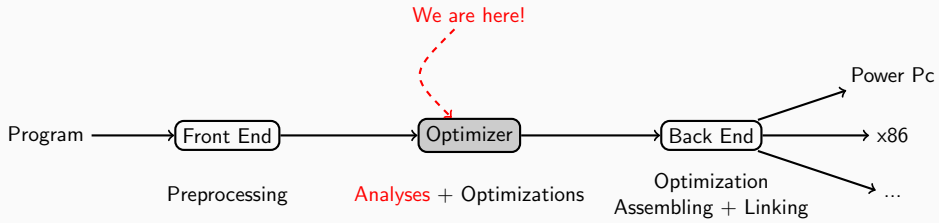
How to design *static analyses* that **scale enough** to be embedded inside compilers ?

- From abstract interpretation to sparse abstract interpretation (course 2)

Dataflow analyses inside compilers

Where ?

Static dataflow analyses on the control flow graph.



some static analyses might also be performed at the AST tree/ LLVM level/ at the binary level

Liveness analysis

In the sequel we call **variable** a pseudo-register or a physical register.

Definition - Alive variable

In a given program point, a variable is said to be *alive* if the value she contains may be used in the rest of the execution.

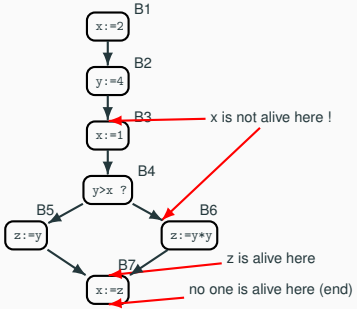
May : non decidable property ▶ overapproximation.

Definition for liveness analysis

Definition

A variable is **live** at the exit of a block if there exists a path from the block to a use of the variable that does not redefine the variable.

```
x:=2;
y:=4;
x:=1;
if (y>x)
   then z:=y
   else z=y*y ;
x:=z;
```



- The information flow is **backward** : from uses to definitions.

Data flow definition

Definition

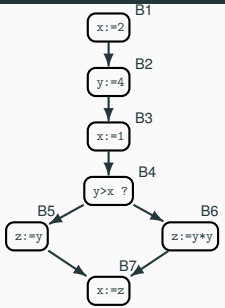
A variable that appears on the left hand side of an assignment is **killed** by the block. Tests do no kill variables.

```
x:=2 // x is killed
```

Definition

A **generated** variable is a variable that appears in the block.

```
y > x // x,y generated
```

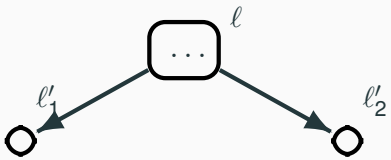


ℓ	$kill(\ell)$	$gen(\ell)$
1	$\{x\}$	$\emptyset$
2	$\{y\}$	$\emptyset$
3	$\{x\}$	$\emptyset$
4	$\emptyset$	$\{x, y\}$
5	$\{z\}$	$\{y\}$
6	$\{z\}$	$\{y\}$
7	$\{x\}$	$\{z\}$

Data flow equations

Idea : propagate along edges ! **Ex** : if x alive in ℓ_1 and not at ℓ'_2, it may be alive at ℓ

$$LV_{exit}(\ell) = \begin{cases} \emptyset & \text{if } \ell = final \\ \bigcup_{(\ell,\ell') \in flow(G)} LV_{entry}(\ell') & \end{cases}$$



$$LV_{entry}(\ell) = (LV_{exit}(\ell) \setminus kill_{LV}(\ell)) \cup gen_{LV}(\ell)$$

Data flow equations - solving

Algorithm :

- Initialise LV sets to $\emptyset$.
- Compute LV_{entry} sets, then LV_{exit}, and continue.
- Stop when a fix point is reached.

▶ (vector of) Sets are strictly growing, and the live range set is at most the set of all variables, thus **this algorithm terminates**.

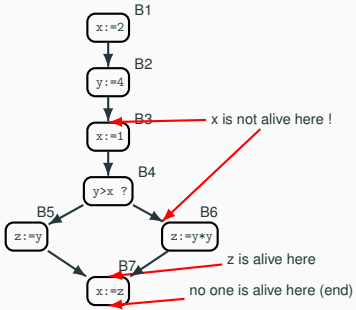
Steps

$LV_{entry}(\ell)$ denoted by $In(\ell)$, $LV_{entry}(\ell)$ by $Out(\ell)$ initilisation to emptysets is not depicted.

			Step 1		Step 2		Step 3 (stable)
ℓ	$kill(\ell)$	$gen(\ell)$	$In(\ell)$	$Out(\ell)$	$In(\ell)$	$Out(\ell)$	$In(\ell)$
1	$\{x\}$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
2	$\{y\}$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\{y\}$	$\emptyset$
3	$\{x\}$	$\emptyset$	$\emptyset$	$\{x, y\}$	$\{y\}$	$\{x, y\}$	$\{y\}$
4	$\emptyset$	$\{x, y\}$	$\{x, y\}$	$\{y\}$	$\{x, y\}$	$\{y\}$	$\{x, y\}$
5	$\{z\}$	$\{y\}$	$\{y\}$	$\{z\}$	$\{y\}$	$\{z\}$	$\{y\}$
6	$\{z\}$	$\{y\}$	$\{y\}$	$\{z\}$	$\{y\}$	$\{z\}$	$\{y\}$
7	$\{x\}$	$\{z\}$	$\{z\}$	$\emptyset$	$\{z\}$	$\emptyset$	$\{z\}$

Final result

ℓ	$LV_{entry}(\ell)$	$LV_{exit}(\ell)$
1	$\emptyset$	$\emptyset$
2	$\emptyset$	$\{y\}$
3	$\{y\}$	$\{x, y\}$
4	$\{x, y\}$	$\{y\}$
5	$\{y\}$	$\{z\}$
6	$\{y\}$	$\{z\}$
7	$\{z\}$	$\emptyset$



Liveness information use in compilers

What for :

- dead code elimination : any $x :=$ with x not alive is a dead assignment.
- register alloc : two simultaneous alive variable cannot be assigned to the same memory location.

The Monotone Framework - a first framework to dataflow

	Backward	Forward
May	$IN(p) = (OUT(p) \setminus \{v\}) \cup vars(E)$	$IN(p) = \bigcup OUT(p_s), p_s \in pred(p)$
	$OUT(p) = \bigcup IN(p_s), p_s \in succ(p)$	$OUT(p) = (IN(p) \setminus \{defs(v)\}) \cup \{p\}$
	Liveness	Reaching Defs
Must	$IN(p) = (OUT(p) \setminus \{v\}) \cup \{E\}$	$IN(p) = \bigcap OUT(p_s), p_s \in pred(p)$
	$OUT(p) = \bigcap IN(p_s), p_s \in succ(p)$	$OUT(p) = (IN(p) \cup \{E\}) \setminus \{Expr(v)\}$
	Very Busy Expressions	Available Expressions

Reference

Kam, J. B. and J. D. Ullman, "Monotone Data Flow Analysis Frameworks", Actal Informatica 7 :3 (1977), pp. 305-318.

Monotone framework : Chaotic Iterations

$$
\begin{array}{l}
x_1 = \bot, x_2 = \bot, \ldots, x_n = \bot \\
\text{do} \\
\qquad t_1 = x_1 ; \ldots ; t_n = x_n \\
\qquad x_1 = F_1(x_1, \ldots, x_n) \\
\qquad \ldots \\
\qquad x_n = F_n(x_1, \ldots, x_n) \\
\text{while } (x_1 \neq t_1 \text{ or } \ldots \text{ or } x_n \neq t_n)
\end{array}
$$

Questions :

- Why does it always stop ?
- Is it correct ?

Monotone framework : properties

Theorem we want

If we use any (fair) iterative worklist algorithm to solve a set of equations with $<$**insert properties**$>$, then it terminates and is correct.

What about liveness ?

- It works because set always grow : **orderings, monotonicity**
- It works because sets are finite.

Monotone framework : a first abstract interpretation instance ?

Abstract Interpretation as the follow up of dataflow analyses

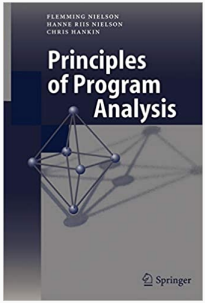
All ingredients of abstract interpretation are already in the monotone framework, except :

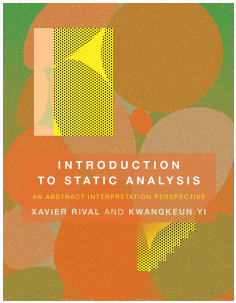
- The notion of domain : domains are all sets.
- The notion of (possibly) infinite computations.

(Classical) abstract interpretation

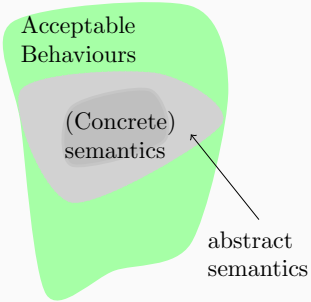
Reference books

Nice books for AI :





Plan



- Program meaning : **concrete** semantics
- Acceptable behaviors : see later.
- Approximation of the program meaning : **abstract** semantics

▶ here we will focus on **numerical** properties.

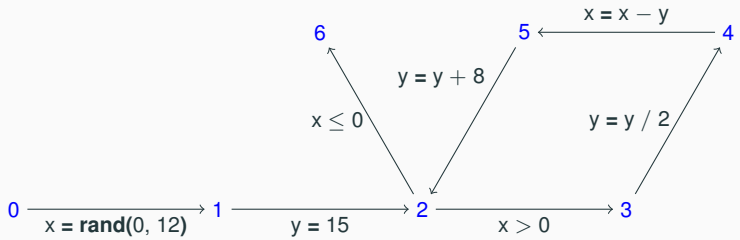
Abstract interpretation

We will design interpreters that work abstractly.

Programs as transition systems

Our programs are CFGs-like :

```
0 x = rand(0, 12);
1 y = 15;
while 2(x > 0) {
    3 y = y / 2;
    4 x = x − y;
    5 y = y + 8;
}6
```

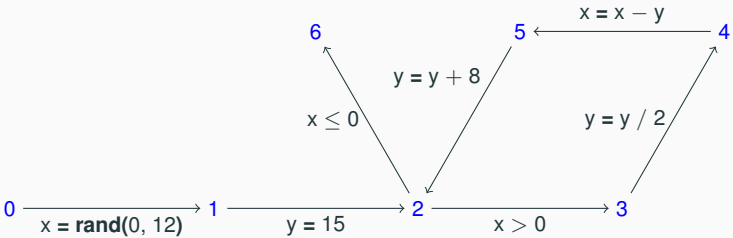


With two types of “statements” : guards, and actions.

Semantics

The effet of the program on memory (set of values) is called the concrete (collecting) semantics.

Concrete semantics



- A **state** is a pair
  - k is a control point $\in L$
  - $\sigma : \text{Var} \to \mathbf{Z}$.
- **Initial** states : $(0, allv)$.

Semantics : what is the effect (on states) of

- commands : guards,actions ?
- the whole program $P = (L, A)$?

Concrete semantics of individual commands

Semantics of commands (guards/actions)

$[\![c]\!]_{\mathrm{C}}$ or "what is the effect of a step" ?

These functions have type $\mathcal{P}(\mathrm{Var} \to \mathbf{Z}) \to \mathcal{P}(\mathrm{Var} \to \mathbf{Z})$

Exercise Compute the semantics for :

- The (application of the) command $x := 39 + x$
- With initial set of valuations $\Sigma = \{[x \mapsto 3], [x \mapsto 4]\}$

Concrete semantics of the whole program

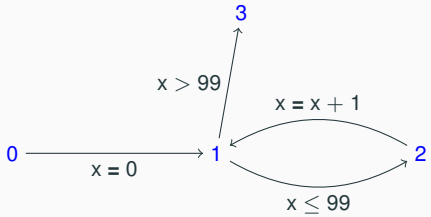
Semantic of P

$[\![P = (L, A)]\!]$ or "What are all possible values for all variables at all control points in any execution" ?

This function has type : $L \to \mathcal{P}(\text{Var} \to \mathbf{Z})$

Example

$_0$x = 0 ;
while $_1$(x $\leq$ 99) {
  $_2$x = x + 1 ;
}$_3$



- $0 \mapsto \{[x \mapsto i], i \in \mathbf{Z}\}$
- **Other control points ?**

How to compute the concrete semantics ?

We “evaluate the program” :

- start : $i \leftarrow 0$, $R^0 \leftarrow$ “all possible valuations for k=0, empty for the other”
- loop : Compute R^{i+1} from R^i and the concrete semantics, ie for all k' control point :
  - apply $[\![c]\!]_{\mathrm{C}}$ on all incoming information R^i_k from incoming edges.
  - make the union of all these new sets to obtain $R^{i+1}_{k'}$
- If $R^{i+1} = R^i$ return this set else goto loop.

What did we do ?

We computed the least fixpoint of :

$$\begin{cases} R_0 = \mathrm{Var} \to \mathbf{Z} \\ R_{k'} = \bigcup\limits_{(k,c,k') \in A} [\![c]\!]_{\mathrm{C}}\,(R_k) \quad k' \neq 0 \end{cases}$$

Result

Such a solution always exists (Knaster-Tarski theorem)

Result 2

It is not computable in general (the algorithm do not always terminate but it may !)

- Hum, it ressembles **dataflow** equations ? Yes !

Abstract semantics : computing invariants

Recall the fixpoint equations :

$$\begin{cases} R_0 = \text{Var} \to \mathbf{Z} \\ R_{k'} = \bigcup\limits_{(k,c,k') \in A} [\![c]\!]_{\mathrm{C}} (R_k) & k' \neq 0 \end{cases}$$

That we solve using an iterative algorithm : $R^0, \ldots R^i$

We have to cope with two problems :

- **Representing the (set of) valuations** $R^i(_k)$ and computing $[\![c]\!]_{\mathrm{C}} (R^i)$ and unions.
- The number of steps can be infinite.

Representing sets of valuations 1/2

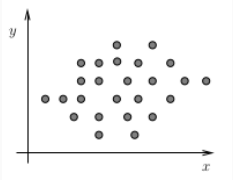
First problem to cope with : **represent sets of** valuations

$$R^{(i)} : (k \to) : \mathcal{P}(\mathrm{Var} \to \mathbf{Z})$$

Observations :

- $\mathrm{Var} \cong \mathbf{N}^d$ "variables can be numbered" $\rightsquigarrow$ **x is the first variable/component**
- Values are vectors in $\mathbf{Z}^d$:

Thus, set of values (one per control point !) :

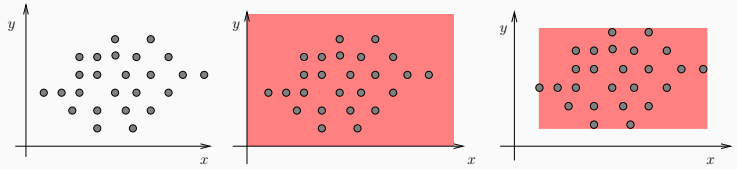


Representing sets of valuations 2/2

Idea : Represent values of variables :

$$R_k \in \mathcal{P}(\mathbf{Z}^d)$$

by a **finite computable superset** $R_k^\sharp$:



- And compute such **abstract values** for *each control point* :

Example :

$k : x \mapsto \{\text{concrete values}\}$ becomes $k : x \mapsto$ " ≥ 0"

Computing the transition relation

Second problem to cope with : **computing** the transition relation for commands and the union on these sets :

$$\bigcup_{(k,c,k')\in A} [\![c]\!]_{\mathrm{C}}(R_k) \quad \rightsquigarrow \quad \bigsqcup^{\sharp}_{(k,c,k')\in A} [\![c]\!]^{\sharp}_{\mathrm{C}}(R^{\sharp}_k)$$

- Commands are abstracted (“abstractly evaluated”)
- Union is abstracted.

▶ We have to change the concrete semantics of operations $[\![]\!]_{\mathrm{C}}$ into an **abstract semantics** $[\![]\!]^{\sharp}_{\mathrm{C}}$ and invent a proper compatible **abstract union**.

Example What is the effect of $[\![x > 0]\!]^{\sharp}_{\mathrm{C}}$ on $x \mapsto$ " ≥ 0" ?

Abstract Interpretation, Algorithm

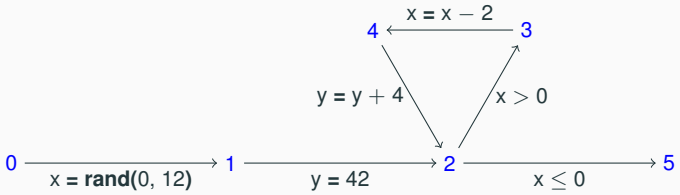
(High-level) Algorithm :

- Write the abstract equations corresponding to the abstract semantics.
- Interpret the program from the beginning, but **abstractly** :
  - Compute an abstract value element per control point.
  - Always make the union (join) with the former value.
- Stop when the iteration has stabilized (for all control points).

credit examples, P. Roux for Onera

Example of computation of the abstract fixpoint (Sign Domain) 1/2

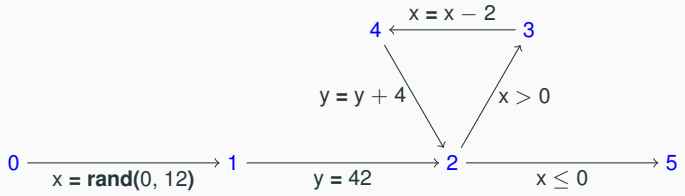
```
0 x = rand(0, 12)
1 y = 42;
while 2(x > 0) {
    3 x = x − 2;
    4 y = y + 4;
}5
```



- **Goal** : propagate the **Sign** information for each variable, with notations ≥ 0, ≤ 0, $\top$ (can be positive or negative), $\bot$ (I do not know anything yet).

Example of computation of the abstract fixpoint (Sign Domain) 2/2

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

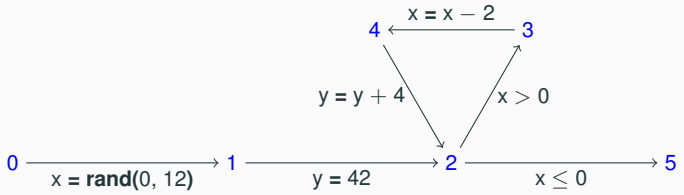


$$
\begin{aligned}
R_0^{\sharp\,i+1} &= \top_{\mathrm{nr}} \\
R_1^{\sharp\,i+1} &= R_0^{\sharp\,i+1}\,[x \mapsto \geq 0] \\
R_2^{\sharp\,i+1} &= R_1^{\sharp\,i+1}\,[y \mapsto \geq 0] \sqcup^{\sharp}_{\mathrm{nr}} \\
&\qquad R_4^{\sharp\,i}\left[y \mapsto R_4^{\sharp\,i}(y) +^{\sharp} (\geq 0)\right] \\
R_3^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \sqcap^{\sharp} \geq 0\right] \\
R_4^{\sharp\,i+1} &= R_3^{\sharp\,i+1}\left[x \mapsto R_3^{\sharp\,i+1}(x) -^{\sharp} (\geq 0)\right] \\
R_5^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1} \sqcap^{\sharp} \leq 0\right]
\end{aligned}
$$

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

Example of computation of the abstract fixpoint (Sign Domain) 2/2

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

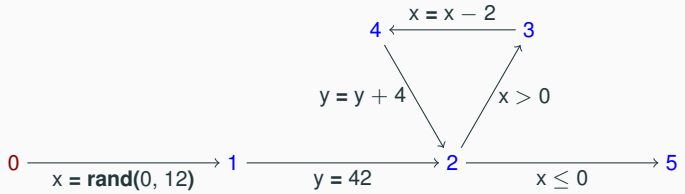


$$
\begin{aligned}
R_0^{\sharp\,i+1} &= \top_{\mathrm{nr}} \\
R_1^{\sharp\,i+1} &= R_0^{\sharp\,i+1}\,[x \mapsto \geq 0] \\
R_2^{\sharp\,i+1} &= R_1^{\sharp\,i+1}\,[y \mapsto \geq 0] \sqcup^{\sharp}_{\mathrm{nr}} \\
&\qquad R_4^{\sharp\,i}\left[y \mapsto R_4^{\sharp\,i}(y) +^{\sharp} (\geq 0)\right] \\
R_3^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \sqcap^{\sharp} \geq 0\right] \\
R_4^{\sharp\,i+1} &= R_3^{\sharp\,i+1}\left[x \mapsto R_3^{\sharp\,i+1}(x) -^{\sharp} (\geq 0)\right] \\
R_5^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1} \sqcap^{\sharp} \leq 0\right]
\end{aligned}
$$

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

Example of computation of the abstract fixpoint (Sign Domain) 2/2

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

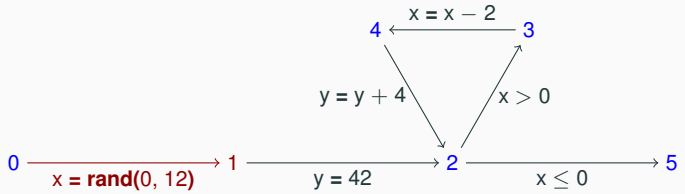


$$
\begin{aligned}
R_0^{\sharp\,i+1} &= \top_{\mathrm{nr}} \\
R_1^{\sharp\,i+1} &= R_0^{\sharp\,i+1}\,[x \mapsto \geq 0] \\
R_2^{\sharp\,i+1} &= R_1^{\sharp\,i+1}\,[y \mapsto \geq 0] \sqcup^{\sharp}_{\mathrm{nr}} \\
&\qquad R_4^{\sharp\,i}\left[y \mapsto R_4^{\sharp\,i}(y) +^{\sharp} (\geq 0)\right] \\
R_3^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \sqcap^{\sharp} \geq 0\right] \\
R_4^{\sharp\,i+1} &= R_3^{\sharp\,i+1}\left[x \mapsto R_3^{\sharp\,i+1}(x) -^{\sharp} (\geq 0)\right] \\
R_5^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1} \sqcap^{\sharp} \leq 0\right]
\end{aligned}
$$

k	$R_k^{\sharp\,0}$	$R_k^{\sharp\,1}$	$R_k^{\sharp\,2}$	$R_k^{\sharp\,3}$
0	$(\bot, \bot)$	$(\top, \top)$		
1	$(\bot, \bot)$			
2	$(\bot, \bot)$			
3	$(\bot, \bot)$			
4	$(\bot, \bot)$			
5	$(\bot, \bot)$			

Example of computation of the abstract fixpoint (Sign Domain) 2/2

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

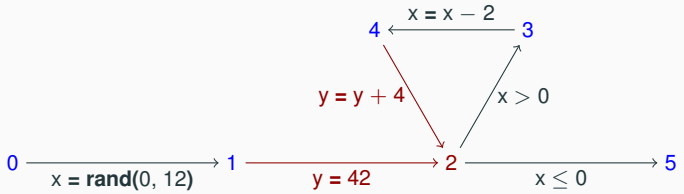


$$
\begin{aligned}
R_0^{\sharp\,i+1} &= \top_{\mathrm{nr}} \\
R_1^{\sharp\,i+1} &= R_0^{\sharp\,i+1}\,[x \mapsto \geq 0] \\
R_2^{\sharp\,i+1} &= R_1^{\sharp\,i+1}\,[y \mapsto \geq 0] \sqcup^{\sharp}_{\mathrm{nr}} \\
&\qquad R_4^{\sharp\,i}\left[y \mapsto R_4^{\sharp\,i}(y) +^{\sharp} (\geq 0)\right] \\
R_3^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \sqcap^{\sharp} \geq 0\right] \\
R_4^{\sharp\,i+1} &= R_3^{\sharp\,i+1}\left[x \mapsto R_3^{\sharp\,i+1}(x) -^{\sharp} (\geq 0)\right] \\
R_5^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1} \sqcap^{\sharp} \leq 0\right]
\end{aligned}
$$

k	$R_k^{\sharp\,0}$	$R_k^{\sharp\,1}$	$R_k^{\sharp\,2}$	$R_k^{\sharp\,3}$
0	$(\bot, \bot)$	$(\top, \top)$		
1	$(\bot, \bot)$	$(\geq 0, \top)$		
2	$(\bot, \bot)$			
3	$(\bot, \bot)$			
4	$(\bot, \bot)$			
5	$(\bot, \bot)$			

Example of computation of the abstract fixpoint (Sign Domain) 2/2

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



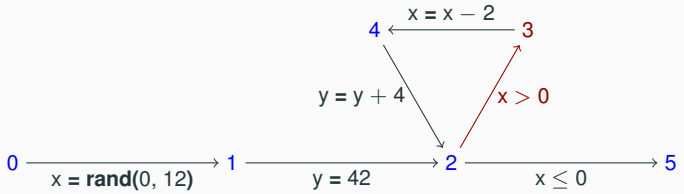
$$
\begin{aligned}
R_0^{\sharp\,i+1} &= \top_{\mathrm{nr}} \\
R_1^{\sharp\,i+1} &= R_0^{\sharp\,i+1}\,[x \mapsto \geq 0] \\
R_2^{\sharp\,i+1} &= R_1^{\sharp\,i+1}\,[y \mapsto \geq 0] \sqcup^\sharp_{\mathrm{nr}} \\
&\qquad R_4^{\sharp\,i}\left[y \mapsto R_4^{\sharp\,i}(y) +^\sharp (\geq 0)\right] \\
R_3^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \sqcap^\sharp \geq 0\right] \\
R_4^{\sharp\,i+1} &= R_3^{\sharp\,i+1}\left[x \mapsto R_3^{\sharp\,i+1}(x) -^\sharp (\geq 0)\right] \\
R_5^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1} \sqcap^\sharp \leq 0\right]
\end{aligned}
$$

k	$R_k^{\sharp\,0}$	$R_k^{\sharp\,1}$	$R_k^{\sharp\,2}$	$R_k^{\sharp\,3}$
0	$(\bot, \bot)$	$(\top, \top)$		
1	$(\bot, \bot)$	$(\geq 0, \top)$		
2	$(\bot, \bot)$	$(\geq 0, \geq 0)$		
3	$(\bot, \bot)$			
4	$(\bot, \bot)$			
5	$(\bot, \bot)$			

$$(\geq 0, \geq 0) \sqcup^\sharp_{\mathrm{nr}} (\bot, \bot)$$

Example of computation of the abstract fixpoint (Sign Domain) 2/2

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

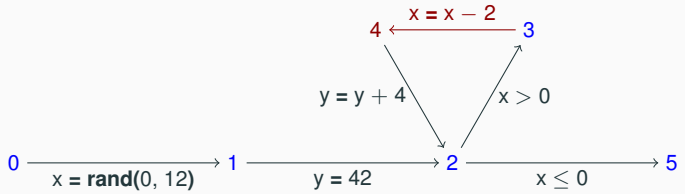


$$
\begin{aligned}
R_0^{\sharp\,i+1} &= \top_{\mathrm{nr}} \\
R_1^{\sharp\,i+1} &= R_0^{\sharp\,i+1}\,[x \mapsto \geq 0] \\
R_2^{\sharp\,i+1} &= R_1^{\sharp\,i+1}\,[y \mapsto \geq 0] \sqcup^{\sharp}_{\mathrm{nr}} \\
&\qquad R_4^{\sharp\,i}\left[y \mapsto R_4^{\sharp\,i}(y) +^{\sharp} (\geq 0)\right] \\
R_3^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \sqcap^{\sharp} \geq 0\right] \\
R_4^{\sharp\,i+1} &= R_3^{\sharp\,i+1}\left[x \mapsto R_3^{\sharp\,i+1}(x) -^{\sharp} (\geq 0)\right] \\
R_5^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1} \sqcap^{\sharp} \leq 0\right]
\end{aligned}
$$

k	$R_k^{\sharp\,0}$	$R_k^{\sharp\,1}$	$R_k^{\sharp\,2}$	$R_k^{\sharp\,3}$
0	$(\bot, \bot)$	$(\top, \top)$		
1	$(\bot, \bot)$	$(\geq 0, \top)$		
2	$(\bot, \bot)$	$(\geq 0, \geq 0)$		
3	$(\bot, \bot)$	$(\geq 0, \geq 0)$		
4	$(\bot, \bot)$			
5	$(\bot, \bot)$			

Example of computation of the abstract fixpoint (Sign Domain) 2/2

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

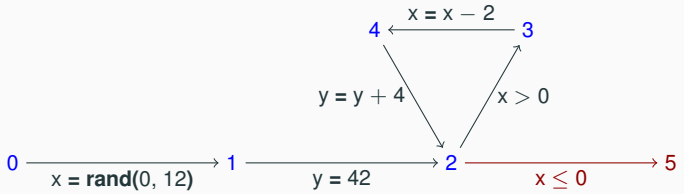


$$
\begin{aligned}
R_0^{\sharp\,i+1} &= \top_{\mathrm{nr}} \\
R_1^{\sharp\,i+1} &= R_0^{\sharp\,i+1}\,[x \mapsto \geq 0] \\
R_2^{\sharp\,i+1} &= R_1^{\sharp\,i+1}\,[y \mapsto \geq 0] \sqcup^{\sharp}_{\mathrm{nr}} \\
&\qquad R_4^{\sharp\,i}\left[y \mapsto R_4^{\sharp\,i}(y) +^{\sharp} (\geq 0)\right] \\
R_3^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \sqcap^{\sharp} \geq 0\right] \\
R_4^{\sharp\,i+1} &= R_3^{\sharp\,i+1}\left[x \mapsto R_3^{\sharp\,i+1}(x) -^{\sharp} (\geq 0)\right] \\
R_5^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1} \sqcap^{\sharp} \leq 0\right]
\end{aligned}
$$

k	$R_k^{\sharp\,0}$	$R_k^{\sharp\,1}$	$R_k^{\sharp\,2}$	$R_k^{\sharp\,3}$
0	$(\bot, \bot)$	$(\top, \top)$		
1	$(\bot, \bot)$	$(\geq 0, \top)$		
2	$(\bot, \bot)$	$(\geq 0, \geq 0)$		
3	$(\bot, \bot)$	$(\geq 0, \geq 0)$		
4	$(\bot, \bot)$	$(\top, \geq 0)$		
5	$(\bot, \bot)$			

Example of computation of the abstract fixpoint (Sign Domain) 2/2

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

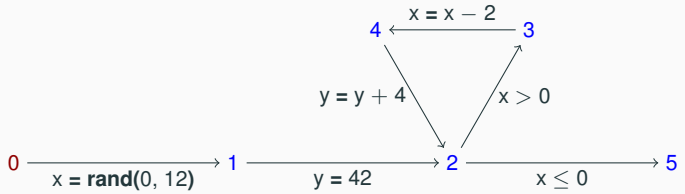


$$
\begin{aligned}
R_0^{\sharp\,i+1} &= \top_{\mathrm{nr}} \\
R_1^{\sharp\,i+1} &= R_0^{\sharp\,i+1}\,[x \mapsto \geq 0] \\
R_2^{\sharp\,i+1} &= R_1^{\sharp\,i+1}\,[y \mapsto \geq 0] \sqcup^{\sharp}_{\mathrm{nr}} \\
&\qquad R_4^{\sharp\,i}\left[y \mapsto R_4^{\sharp\,i}(y) +^{\sharp} (\geq 0)\right] \\
R_3^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \sqcap^{\sharp} \geq 0\right] \\
R_4^{\sharp\,i+1} &= R_3^{\sharp\,i+1}\left[x \mapsto R_3^{\sharp\,i+1}(x) -^{\sharp} (\geq 0)\right] \\
R_5^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1} \sqcap^{\sharp} \leq 0\right]
\end{aligned}
$$

k	$R_k^{\sharp\,0}$	$R_k^{\sharp\,1}$	$R_k^{\sharp\,2}$	$R_k^{\sharp\,3}$
0	$(\bot, \bot)$	$(\top, \top)$		
1	$(\bot, \bot)$	$(\geq 0, \top)$		
2	$(\bot, \bot)$	$(\geq 0, \geq 0)$		
3	$(\bot, \bot)$	$(\geq 0, \geq 0)$		
4	$(\bot, \bot)$	$(\top, \geq 0)$		
5	$(\bot, \bot)$	$(0, \geq 0)$		

Example of computation of the abstract fixpoint (Sign Domain) 2/2

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

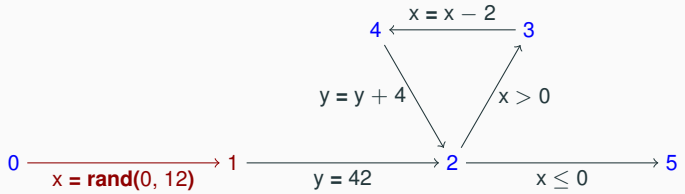


$$R_0^{\sharp\,i+1} = \top_{\mathrm{nr}}$$
$$R_1^{\sharp\,i+1} = R_0^{\sharp\,i+1}\,[x \mapsto \geq 0]$$
$$R_2^{\sharp\,i+1} = R_1^{\sharp\,i+1}\,[y \mapsto \geq 0] \sqcup^{\sharp}_{\mathrm{nr}} R_4^{\sharp\,i}\left[y \mapsto R_4^{\sharp\,i}(y) +^{\sharp} (\geq 0)\right]$$
$$R_3^{\sharp\,i+1} = R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \sqcap^{\sharp} \geq 0\right]$$
$$R_4^{\sharp\,i+1} = R_3^{\sharp\,i+1}\left[x \mapsto R_3^{\sharp\,i+1}(x) -^{\sharp} (\geq 0)\right]$$
$$R_5^{\sharp\,i+1} = R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1} \sqcap^{\sharp} \leq 0\right]$$

k	$R_k^{\sharp\,0}$	$R_k^{\sharp\,1}$	$R_k^{\sharp\,2}$	$R_k^{\sharp\,3}$
0	$(\bot, \bot)$	$(\top, \top)$	$(\top, \top)$	
1	$(\bot, \bot)$	$(\geq 0, \top)$		
2	$(\bot, \bot)$	$(\geq 0, \geq 0)$		
3	$(\bot, \bot)$	$(\geq 0, \geq 0)$		
4	$(\bot, \bot)$	$(\top, \geq 0)$		
5	$(\bot, \bot)$	$(0, \geq 0)$		

Example of computation of the abstract fixpoint (Sign Domain) 2/2

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

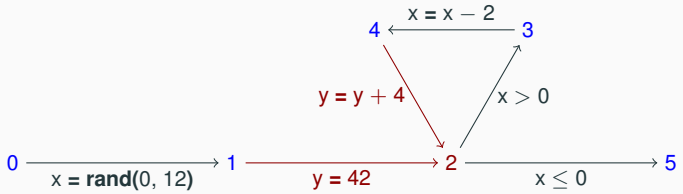


$$
\begin{aligned}
R_0^{\sharp\,i+1} &= \top_{\mathrm{nr}} \\
R_1^{\sharp\,i+1} &= R_0^{\sharp\,i+1}\,[x \mapsto \geq 0] \\
R_2^{\sharp\,i+1} &= R_1^{\sharp\,i+1}\,[y \mapsto \geq 0] \sqcup^{\sharp}_{\mathrm{nr}} \\
&\qquad R_4^{\sharp\,i}\left[y \mapsto R_4^{\sharp\,i}(y) +^{\sharp} (\geq 0)\right] \\
R_3^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \sqcap^{\sharp} \geq 0\right] \\
R_4^{\sharp\,i+1} &= R_3^{\sharp\,i+1}\left[x \mapsto R_3^{\sharp\,i+1}(x) -^{\sharp} (\geq 0)\right] \\
R_5^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1} \sqcap^{\sharp} \leq 0\right]
\end{aligned}
$$

k	$R_k^{\sharp\,0}$	$R_k^{\sharp\,1}$	$R_k^{\sharp\,2}$	$R_k^{\sharp\,3}$
0	$(\bot, \bot)$	$(\top, \top)$	$(\top, \top)$	
1	$(\bot, \bot)$	$(\geq 0, \top)$	$(\geq 0, \top)$	
2	$(\bot, \bot)$	$(\geq 0, \geq 0)$		
3	$(\bot, \bot)$	$(\geq 0, \geq 0)$		
4	$(\bot, \bot)$	$(\top, \geq 0)$		
5	$(\bot, \bot)$	$(0, \geq 0)$		

Example of computation of the abstract fixpoint (Sign Domain) 2/2

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



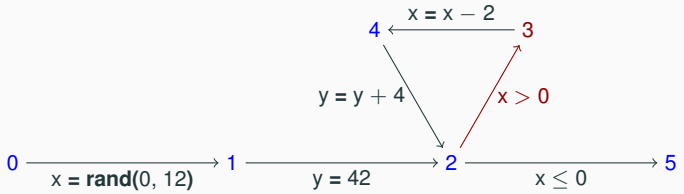
$$R_0^{\sharp\,i+1} = \top_{\text{nr}}$$
$$R_1^{\sharp\,i+1} = R_0^{\sharp\,i+1}\,[x \mapsto \geq 0]$$
$$R_2^{\sharp\,i+1} = R_1^{\sharp\,i+1}\,[y \mapsto \geq 0] \sqcup^\sharp_{\text{nr}} R_4^{\sharp\,i}\left[y \mapsto R_4^{\sharp\,i}(y) +^\sharp (\geq 0)\right]$$
$$R_3^{\sharp\,i+1} = R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \sqcap^\sharp \geq 0\right]$$
$$R_4^{\sharp\,i+1} = R_3^{\sharp\,i+1}\left[x \mapsto R_3^{\sharp\,i+1}(x) -^\sharp (\geq 0)\right]$$
$$R_5^{\sharp\,i+1} = R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1} \sqcap^\sharp \leq 0\right]$$

k	$R_k^{\sharp\,0}$	$R_k^{\sharp\,1}$	$R_k^{\sharp\,2}$	$R_k^{\sharp\,3}$
0	$(\bot, \bot)$	$(\top, \top)$	$(\top, \top)$	
1	$(\bot, \bot)$	$(\geq 0, \top)$	$(\geq 0, \top)$	
2	$(\bot, \bot)$	$(\geq 0, \geq 0)$	$(\top, \geq 0)$	
3	$(\bot, \bot)$	$(\geq 0, \geq 0)$		
4	$(\bot, \bot)$	$(\top, \geq 0)$		
5	$(\bot, \bot)$	$(0, \geq 0)$		

$$(\geq 0, \geq 0) \sqcup^\sharp_{\text{nr}} (\top, \geq 0)$$

Example of computation of the abstract fixpoint (Sign Domain) 2/2

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

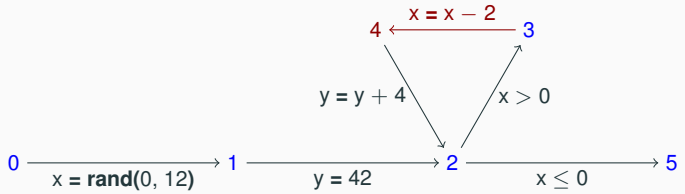


$$
\begin{aligned}
R_0^{\sharp\,i+1} &= \top_{\mathrm{nr}} \\
R_1^{\sharp\,i+1} &= R_0^{\sharp\,i+1}\,[x \mapsto \geq 0] \\
R_2^{\sharp\,i+1} &= R_1^{\sharp\,i+1}\,[y \mapsto \geq 0] \sqcup^\sharp_{\mathrm{nr}} \\
&\qquad R_4^{\sharp\,i}\left[y \mapsto R_4^{\sharp\,i}(y) +^\sharp (\geq 0)\right] \\
R_3^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \sqcap^\sharp \geq 0\right] \\
R_4^{\sharp\,i+1} &= R_3^{\sharp\,i+1}\left[x \mapsto R_3^{\sharp\,i+1}(x) -^\sharp (\geq 0)\right] \\
R_5^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1} \sqcap^\sharp \leq 0\right]
\end{aligned}
$$

k	$R_k^{\sharp\,0}$	$R_k^{\sharp\,1}$	$R_k^{\sharp\,2}$	$R_k^{\sharp\,3}$
0	$(\bot, \bot)$	$(\top, \top)$	$(\top, \top)$	
1	$(\bot, \bot)$	$(\geq 0, \top)$	$(\geq 0, \top)$	
2	$(\bot, \bot)$	$(\geq 0, \geq 0)$	$(\top, \geq 0)$	
3	$(\bot, \bot)$	$(\geq 0, \geq 0)$	$(\geq 0, \geq 0)$	
4	$(\bot, \bot)$	$(\top, \geq 0)$		
5	$(\bot, \bot)$	$(0, \geq 0)$		

Example of computation of the abstract fixpoint (Sign Domain) 2/2

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

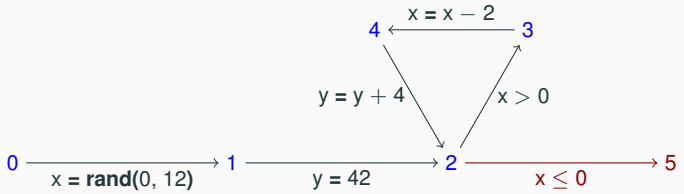


$$
\begin{aligned}
R_0^{\sharp\,i+1} &= \top_{\mathrm{nr}} \\
R_1^{\sharp\,i+1} &= R_0^{\sharp\,i+1}\,[x \mapsto \geq 0] \\
R_2^{\sharp\,i+1} &= R_1^{\sharp\,i+1}\,[y \mapsto \geq 0] \sqcup^{\sharp}_{\mathrm{nr}} \\
&\qquad R_4^{\sharp\,i}\left[y \mapsto R_4^{\sharp\,i}(y) +^{\sharp} (\geq 0)\right] \\
R_3^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \sqcap^{\sharp} \geq 0\right] \\
R_4^{\sharp\,i+1} &= R_3^{\sharp\,i+1}\left[x \mapsto R_3^{\sharp\,i+1}(x) -^{\sharp} (\geq 0)\right] \\
R_5^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1} \sqcap^{\sharp} \leq 0\right]
\end{aligned}
$$

k	$R_k^{\sharp\,0}$	$R_k^{\sharp\,1}$	$R_k^{\sharp\,2}$	$R_k^{\sharp\,3}$
0	$(\bot, \bot)$	$(\top, \top)$	$(\top, \top)$	
1	$(\bot, \bot)$	$(\geq 0, \top)$	$(\geq 0, \top)$	
2	$(\bot, \bot)$	$(\geq 0, \geq 0)$	$(\top, \geq 0)$	
3	$(\bot, \bot)$	$(\geq 0, \geq 0)$	$(\geq 0, \geq 0)$	
4	$(\bot, \bot)$	$(\top, \geq 0)$	$(\top, \geq 0)$	
5	$(\bot, \bot)$	$(0, \geq 0)$		

Example of computation of the abstract fixpoint (Sign Domain) 2/2

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

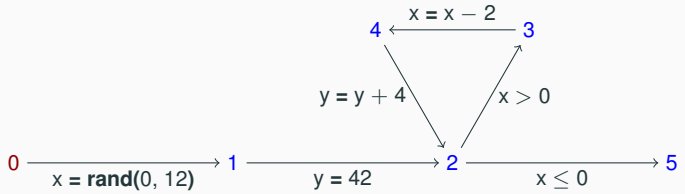


$$
\begin{aligned}
R_0^{\sharp\,i+1} &= \top_{\mathrm{nr}} \\
R_1^{\sharp\,i+1} &= R_0^{\sharp\,i+1}\,[x \mapsto \geq 0] \\
R_2^{\sharp\,i+1} &= R_1^{\sharp\,i+1}\,[y \mapsto \geq 0] \sqcup^\sharp_{\mathrm{nr}} \\
&\qquad R_4^{\sharp\,i}\left[y \mapsto R_4^{\sharp\,i}(y) +^\sharp (\geq 0)\right] \\
R_3^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \sqcap^\sharp \geq 0\right] \\
R_4^{\sharp\,i+1} &= R_3^{\sharp\,i+1}\left[x \mapsto R_3^{\sharp\,i+1}(x) -^\sharp (\geq 0)\right] \\
R_5^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1} \sqcap^\sharp \leq 0\right]
\end{aligned}
$$

k	$R_k^{\sharp\,0}$	$R_k^{\sharp\,1}$	$R_k^{\sharp\,2}$	$R_k^{\sharp\,3}$
0	$(\bot, \bot)$	$(\top, \top)$	$(\top, \top)$	
1	$(\bot, \bot)$	$(\geq 0, \top)$	$(\geq 0, \top)$	
2	$(\bot, \bot)$	$(\geq 0, \geq 0)$	$(\top, \geq 0)$	
3	$(\bot, \bot)$	$(\geq 0, \geq 0)$	$(\geq 0, \geq 0)$	
4	$(\bot, \bot)$	$(\top, \geq 0)$	$(\top, \geq 0)$	
5	$(\bot, \bot)$	$(0, \geq 0)$	$(\leq 0, \geq 0)$	

Example of computation of the abstract fixpoint (Sign Domain) 2/2

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

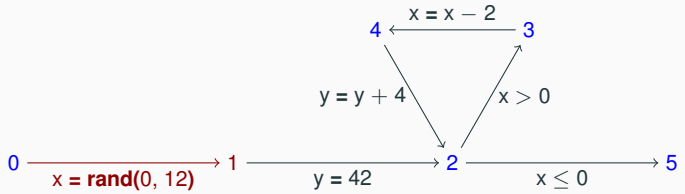


$$R_0^{\sharp\,i+1} = \top_{\mathrm{nr}}$$
$$R_1^{\sharp\,i+1} = R_0^{\sharp\,i+1}\,[x \mapsto \geq 0]$$
$$R_2^{\sharp\,i+1} = R_1^{\sharp\,i+1}\,[y \mapsto \geq 0] \sqcup^{\sharp}_{\mathrm{nr}} R_4^{\sharp\,i}\left[y \mapsto R_4^{\sharp\,i}(y) +^{\sharp} (\geq 0)\right]$$
$$R_3^{\sharp\,i+1} = R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \sqcap^{\sharp} \geq 0\right]$$
$$R_4^{\sharp\,i+1} = R_3^{\sharp\,i+1}\left[x \mapsto R_3^{\sharp\,i+1}(x) -^{\sharp} (\geq 0)\right]$$
$$R_5^{\sharp\,i+1} = R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1} \sqcap^{\sharp} \leq 0\right]$$

k	$R_k^{\sharp\,0}$	$R_k^{\sharp\,1}$	$R_k^{\sharp\,2}$	$R_k^{\sharp\,3}$
0	$(\bot, \bot)$	$(\top, \top)$	$(\top, \top)$	$(\top, \top)$
1	$(\bot, \bot)$	$(\geq 0, \top)$	$(\geq 0, \top)$	
2	$(\bot, \bot)$	$(\geq 0, \geq 0)$	$(\top, \geq 0)$	
3	$(\bot, \bot)$	$(\geq 0, \geq 0)$	$(\geq 0, \geq 0)$	
4	$(\bot, \bot)$	$(\top, \geq 0)$	$(\top, \geq 0)$	
5	$(\bot, \bot)$	$(0, \geq 0)$	$(\leq 0, \geq 0)$	

Example of computation of the abstract fixpoint (Sign Domain) 2/2

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

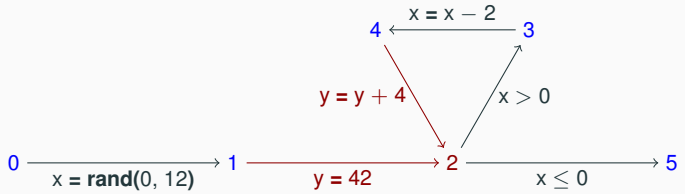


$$R_0^{\sharp\,i+1} = \top_{\mathrm{nr}}$$
$$R_1^{\sharp\,i+1} = R_0^{\sharp\,i+1}\,[x \mapsto \geq 0]$$
$$R_2^{\sharp\,i+1} = R_1^{\sharp\,i+1}\,[y \mapsto \geq 0] \sqcup^{\sharp}_{\mathrm{nr}} R_4^{\sharp\,i}\left[y \mapsto R_4^{\sharp\,i}(y) +^{\sharp} (\geq 0)\right]$$
$$R_3^{\sharp\,i+1} = R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \sqcap^{\sharp} \geq 0\right]$$
$$R_4^{\sharp\,i+1} = R_3^{\sharp\,i+1}\left[x \mapsto R_3^{\sharp\,i+1}(x) -^{\sharp} (\geq 0)\right]$$
$$R_5^{\sharp\,i+1} = R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1} \sqcap^{\sharp} \leq 0\right]$$

k	$R_k^{\sharp\,0}$	$R_k^{\sharp\,1}$	$R_k^{\sharp\,2}$	$R_k^{\sharp\,3}$
0	$(\bot, \bot)$	$(\top, \top)$	$(\top, \top)$	$(\top, \top)$
1	$(\bot, \bot)$	$(\geq 0, \top)$	$(\geq 0, \top)$	$(\geq 0, \top)$
2	$(\bot, \bot)$	$(\geq 0, \geq 0)$	$(\top, \geq 0)$	
3	$(\bot, \bot)$	$(\geq 0, \geq 0)$	$(\geq 0, \geq 0)$	
4	$(\bot, \bot)$	$(\top, \geq 0)$	$(\top, \geq 0)$	
5	$(\bot, \bot)$	$(0, \geq 0)$	$(\leq 0, \geq 0)$	

Example of computation of the abstract fixpoint (Sign Domain) 2/2

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



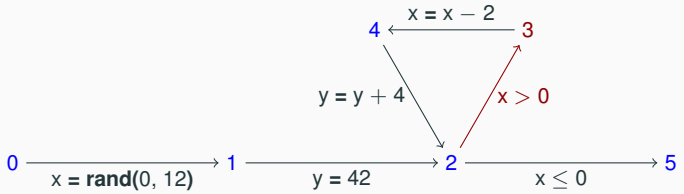
$$
\begin{aligned}
R_0^{\sharp\,i+1} &= \top_{\mathrm{nr}} \\
R_1^{\sharp\,i+1} &= R_0^{\sharp\,i+1}\,[x \mapsto \geq 0] \\
R_2^{\sharp\,i+1} &= R_1^{\sharp\,i+1}\,[y \mapsto \geq 0] \sqcup^\sharp_{\mathrm{nr}} \\
&\qquad R_4^{\sharp\,i}\left[y \mapsto R_4^{\sharp\,i}(y) +^\sharp (\geq 0)\right] \\
R_3^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \sqcap^\sharp \geq 0\right] \\
R_4^{\sharp\,i+1} &= R_3^{\sharp\,i+1}\left[x \mapsto R_3^{\sharp\,i+1}(x) -^\sharp (\geq 0)\right] \\
R_5^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1} \sqcap^\sharp \leq 0\right]
\end{aligned}
$$

k	$R_k^{\sharp\,0}$	$R_k^{\sharp\,1}$	$R_k^{\sharp\,2}$	$R_k^{\sharp\,3}$
0	$(\bot, \bot)$	$(\top, \top)$	$(\top, \top)$	$(\top, \top)$
1	$(\bot, \bot)$	$(\geq 0, \top)$	$(\geq 0, \top)$	$(\geq 0, \top)$
2	$(\bot, \bot)$	$(\geq 0, \geq 0)$	$(\top, \geq 0)$	$(\top, \geq 0)$
3	$(\bot, \bot)$	$(\geq 0, \geq 0)$	$(\geq 0, \geq 0)$	
4	$(\bot, \bot)$	$(\top, \geq 0)$	$(\top, \geq 0)$	
5	$(\bot, \bot)$	$(0, \geq 0)$	$(\leq 0, \geq 0)$	

$$(\geq 0, \geq 0) \sqcup^\sharp_{\mathrm{nr}} (\top, \geq 0)$$

Example of computation of the abstract fixpoint (Sign Domain) 2/2

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

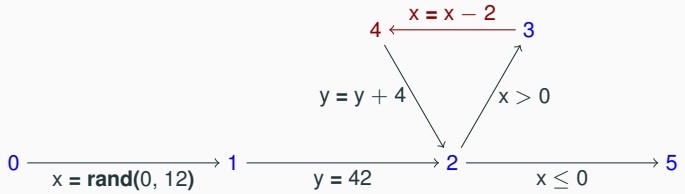


$$
\begin{aligned}
R_0^{\sharp\,i+1} &= \top_{\mathrm{nr}} \\
R_1^{\sharp\,i+1} &= R_0^{\sharp\,i+1}\,[x \mapsto \geq 0] \\
R_2^{\sharp\,i+1} &= R_1^{\sharp\,i+1}\,[y \mapsto \geq 0] \sqcup^{\sharp}_{\mathrm{nr}} \\
&\qquad R_4^{\sharp\,i}\left[y \mapsto R_4^{\sharp\,i}(y) +^{\sharp} (\geq 0)\right] \\
R_3^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \sqcap^{\sharp} \geq 0\right] \\
R_4^{\sharp\,i+1} &= R_3^{\sharp\,i+1}\left[x \mapsto R_3^{\sharp\,i+1}(x) -^{\sharp} (\geq 0)\right] \\
R_5^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1} \sqcap^{\sharp} \leq 0\right]
\end{aligned}
$$

k	$R_k^{\sharp\,0}$	$R_k^{\sharp\,1}$	$R_k^{\sharp\,2}$	$R_k^{\sharp\,3}$
0	$(\bot, \bot)$	$(\top, \top)$	$(\top, \top)$	$(\top, \top)$
1	$(\bot, \bot)$	$(\geq 0, \top)$	$(\geq 0, \top)$	$(\geq 0, \top)$
2	$(\bot, \bot)$	$(\geq 0, \geq 0)$	$(\top, \geq 0)$	$(\top, \geq 0)$
3	$(\bot, \bot)$	$(\geq 0, \geq 0)$	$(\geq 0, \geq 0)$	$(\geq 0, \geq 0)$
4	$(\bot, \bot)$	$(\top, \geq 0)$	$(\top, \geq 0)$	
5	$(\bot, \bot)$	$(0, \geq 0)$	$(\leq 0, \geq 0)$	

Example of computation of the abstract fixpoint (Sign Domain) 2/2

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

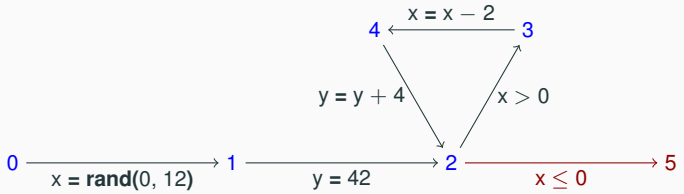


$$
\begin{aligned}
R_0^{\sharp\,i+1} &= \top_{\mathrm{nr}} \\
R_1^{\sharp\,i+1} &= R_0^{\sharp\,i+1}\,[x \mapsto \geq 0] \\
R_2^{\sharp\,i+1} &= R_1^{\sharp\,i+1}\,[y \mapsto \geq 0] \sqcup^{\sharp}_{\mathrm{nr}} \\
&\qquad R_4^{\sharp\,i}\left[y \mapsto R_4^{\sharp\,i}(y) +^{\sharp} (\geq 0)\right] \\
R_3^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \sqcap^{\sharp} \geq 0\right] \\
R_4^{\sharp\,i+1} &= R_3^{\sharp\,i+1}\left[x \mapsto R_3^{\sharp\,i+1}(x) -^{\sharp} (\geq 0)\right] \\
R_5^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1} \sqcap^{\sharp} \leq 0\right]
\end{aligned}
$$

k	$R_k^{\sharp\,0}$	$R_k^{\sharp\,1}$	$R_k^{\sharp\,2}$	$R_k^{\sharp\,3}$
0	$(\bot, \bot)$	$(\top, \top)$	$(\top, \top)$	$(\top, \top)$
1	$(\bot, \bot)$	$(\geq 0, \top)$	$(\geq 0, \top)$	$(\geq 0, \top)$
2	$(\bot, \bot)$	$(\geq 0, \geq 0)$	$(\top, \geq 0)$	$(\top, \geq 0)$
3	$(\bot, \bot)$	$(\geq 0, \geq 0)$	$(\geq 0, \geq 0)$	$(\geq 0, \geq 0)$
4	$(\bot, \bot)$	$(\top, \geq 0)$	$(\top, \geq 0)$	$(\top, \geq 0)$
5	$(\bot, \bot)$	$(0, \geq 0)$	$(\leq 0, \geq 0)$	

Example of computation of the abstract fixpoint (Sign Domain) 2/2

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

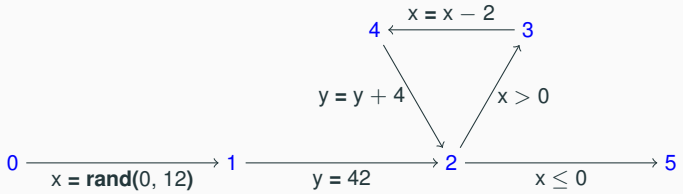


$$
\begin{aligned}
R_0^{\sharp\,i+1} &= \top_{\mathrm{nr}} \\
R_1^{\sharp\,i+1} &= R_0^{\sharp\,i+1}\,[x \mapsto \geq 0] \\
R_2^{\sharp\,i+1} &= R_1^{\sharp\,i+1}\,[y \mapsto \geq 0] \sqcup^\sharp_{\mathrm{nr}} \\
&\qquad R_4^{\sharp\,i}\left[y \mapsto R_4^{\sharp\,i}(y) +^\sharp (\geq 0)\right] \\
R_3^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \sqcap^\sharp \geq 0\right] \\
R_4^{\sharp\,i+1} &= R_3^{\sharp\,i+1}\left[x \mapsto R_3^{\sharp\,i+1}(x) -^\sharp (\geq 0)\right] \\
R_5^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1} \sqcap^\sharp \leq 0\right]
\end{aligned}
$$

k	$R_k^{\sharp\,0}$	$R_k^{\sharp\,1}$	$R_k^{\sharp\,2}$	$R_k^{\sharp\,3}$
0	$(\bot, \bot)$	$(\top, \top)$	$(\top, \top)$	$(\top, \top)$
1	$(\bot, \bot)$	$(\geq 0, \top)$	$(\geq 0, \top)$	$(\geq 0, \top)$
2	$(\bot, \bot)$	$(\geq 0, \geq 0)$	$(\top, \geq 0)$	$(\top, \geq 0)$
3	$(\bot, \bot)$	$(\geq 0, \geq 0)$	$(\geq 0, \geq 0)$	$(\geq 0, \geq 0)$
4	$(\bot, \bot)$	$(\top, \geq 0)$	$(\top, \geq 0)$	$(\top, \geq 0)$
5	$(\bot, \bot)$	$(0, \geq 0)$	$(\leq 0, \geq 0)$	$(\leq 0, \geq 0)$

Example of computation of the abstract fixpoint (Sign Domain) 2/2

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



$$
\begin{aligned}
R_0^{\sharp\,i+1} &= \top_{\mathrm{nr}} \\
R_1^{\sharp\,i+1} &= R_0^{\sharp\,i+1}\,[x \mapsto \geq 0] \\
R_2^{\sharp\,i+1} &= R_1^{\sharp\,i+1}\,[y \mapsto \geq 0] \sqcup^\sharp_{\mathrm{nr}} \\
&\qquad R_4^{\sharp\,i}\left[y \mapsto R_4^{\sharp\,i}(y) +^\sharp (\geq 0)\right] \\
R_3^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1}(x) \sqcap^\sharp \geq 0\right] \\
R_4^{\sharp\,i+1} &= R_3^{\sharp\,i+1}\left[x \mapsto R_3^{\sharp\,i+1}(x) -^\sharp (\geq 0)\right] \\
R_5^{\sharp\,i+1} &= R_2^{\sharp\,i+1}\left[x \mapsto R_2^{\sharp\,i+1} \sqcap^\sharp \leq 0\right]
\end{aligned}
$$

k	$R_k^{\sharp\,0}$	$R_k^{\sharp\,1}$	$R_k^{\sharp\,2}$	$R_k^{\sharp\,3}$
0	$(\bot, \bot)$	$(\top, \top)$	$(\top, \top)$	$(\top, \top)$
1	$(\bot, \bot)$	$(\geq 0, \top)$	$(\geq 0, \top)$	$(\geq 0, \top)$
2	$(\bot, \bot)$	$(\geq 0, \geq 0)$	$(\top, \geq 0)$	$(\top, \geq 0)$
3	$(\bot, \bot)$	$(\geq 0, \geq 0)$	$(\geq 0, \geq 0)$	$(\geq 0, \geq 0)$
4	$(\bot, \bot)$	$(\top, \geq 0)$	$(\top, \geq 0)$	$(\top, \geq 0)$
5	$(\bot, \bot)$	$(0, \geq 0)$	$(\leq 0, \geq 0)$	$(\leq 0, \geq 0)$

We've reached the fixpoint !

Abstract interpretation framework

Theorem we have

If we use any (fair) iterative worklist algorithm to solve the set of abstract equations constructed **according to the framework**, then it terminates and is correct.

Of course :

- The underlying abstract domain should have good algebraic properties (lattice, galois connection, . . .)
- There are some special operations to guarantee termination if the structure is not **good enough**.

A taste of widening

An example with intervals

```
int x=0;
while (x<1000) {
  x=x+1;
}
```

Loop iterations $[0,0]$, $[0,1]$, $[0,2]$, $[0,3]$,...

- Stricly growing interval during 1000 iterations, then stabilizes : $[0,1000]$ is an **invariant**.

Soundness and termination

We still have soundness, but :

Bad news !

- In all generality, it does not terminate !
  Because the lattice has infinite ascending chains
  (e.g., $(\llbracket 0, n \rrbracket)_{n \in \mathbf{N}}$).
- And even if it terminates, it can be long...

Third problem to cope with : **stopping the computation** :

- Too many computations
- unbounded loops

Toward a solution : extrapolation !

```
int x=0;
while (x<1000) {
  x=x+1;
}
```

$[0,0]$, $[0,1]$, $[0,2]$, $[0,3] \rightarrow [0,+\infty)$

Yes ! $[0,\infty[$ is stable !

- interpose between each iteration a **widening** that will prevent following infinite ascending chains by “jumping” higher

Widening on intervals

Idea : when the bounds of an interval are stables, we keep them otherwise we replace them by ∞.

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

Examples

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

Remarks :

- $\triangledown$ is not symmetric
- $x \triangledown y$ often defined with $x \sqsubseteq y$

Definition (widening)

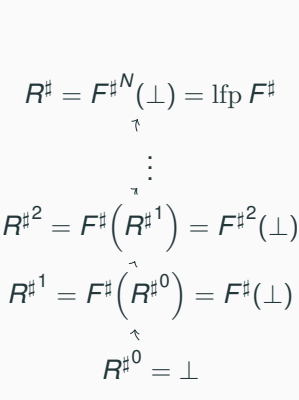
A widening $\triangledown$ is a binary operation ($\triangledown : \mathcal{D}^\sharp \times \mathcal{D}^\sharp \to \mathcal{D}^\sharp$) satisfying

- $\forall x^\sharp, y^\sharp, \quad x^\sharp \sqcup^\sharp y^\sharp \ \sqsubseteq^\sharp \ x^\sharp \triangledown y^\sharp$;
- for all sequence $\left(x_n^\sharp\right)_{n\in\mathbf{N}}$, the increasing sequence

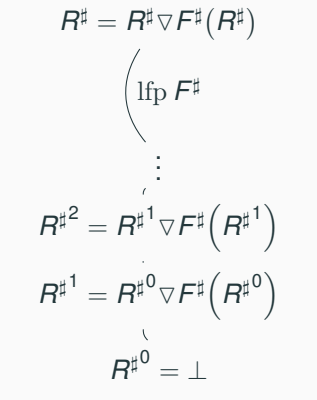
$$\begin{cases} y_0^\sharp &= x_0^\sharp \\ y_{i+1}^\sharp &= y_i^\sharp \triangledown x_{i+1}^\sharp \end{cases}$$

is stationary.

Widening, Illustration



$F^\sharp$ stationary



$F^\sharp$ not stationary, widening

Widening still guarantee soundness !

Cousot/Cousot 77

Iteratively computing the reachable states from the entry point with the interval operators and applying the interval widening at entry nodes of loops converges in **a finite** number of steps to a overapproximation of the least invariant (aka **postfixpoint**).

See “introduction to static analysis”, by X. Rival and K. Yi. for a clean proof.

Demo time : PAGAI

https://pagai.gricad-pages.univ-grenoble-alpes.fr

```c
int main() {
    int x = 0;
    int y = 0;
    while (1) {
        if (x <= 50) {
            y++;
        } else y--;
        if (y < 0) break;
        x++;
    }
    assert(x+y<=101);
    assert(x <= 102);
}
```


```
# Generate llvm-ssa
clang -emit-llvm -g -c gopan.c -o gopan.bc
# Analyze!
pagai -i gopan.bc
```

Demo time : PAGAI 2/2 : final result

```
int main() {
    int x = 0;
    int y = 0;

    /* reachable */

    while (1) {
        /* invariant:
        102-x-y >= 0
        y >= 0
        x-y >= 0 */
        if (x <= 50) { // safe
            y++;
        } else // safe
            y--;
        if (y < 0) break;
        // safe
        x++;
    }

    // safe
    /* assert OK */
    assert(x+y<=101);
    /* assert OK */
    assert(x <= 102);
    /* reachable */
}
```

Conclusion part 1

- Abstract Interpretation to the rescue of dataflow analyses
- Expressive framework . But is it **effective ?**