Narrowing operators on template abstract domains

Gianluca Amato

Università di Chieti–Pescara
Pescara, Italy

20th International Symposium on Formal Methods
FM 2015

(joint work with S. Di Nardo Di Maio, M. C. Meo and F. Scozzari)

Overview

Context

Data-flow analysis
Abstract interpretation
Analysis of numerical properties
Interval domain (and other template domains)

Topic

Under which conditions *narrowing* may be avoided.

Plan of the talk

1. Two-phase (widening/narrowing based) analysis.
2. Narrowing with integer bounds.
3. Narrowing with rational bounds.
4. Conclusions and future work.

Overview

Context

Data-flow analysis
Abstract interpretation
Analysis of numerical properties
Interval domain (and other template domains)

Topic

Under which conditions *narrowing* may be avoided.

Plan of the talk

1. Two-phase (widening/narrowing based) analysis.
2. Narrowing with integer bounds.
3. Narrowing with rational bounds.
4. Conclusions and future work.

Overview

Context

Data-flow analysis
Abstract interpretation
Analysis of numerical properties
Interval domain (and other template domains)

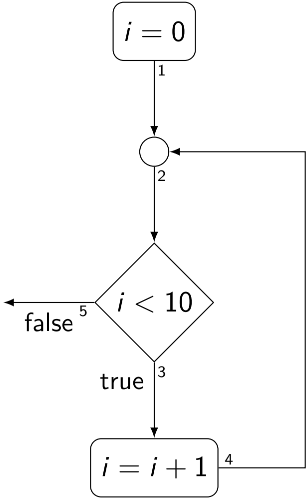
Topic

Under which conditions *narrowing* may be avoided.

Plan of the talk

1. Two-phase (widening/narrowing based) analysis.
2. Narrowing with integer bounds.
3. Narrowing with rational bounds.
4. Conclusions and future work.

An example: interval analysis



$$x_1 = [0, 0]$$
$$x_2 = x_1 \vee x_4$$
$$x_3 = x_2 \wedge [-\infty, 9]$$
$$x_4 = x_3 + [1, 1]$$
$$x_5 = x_3 \wedge [10, +\infty]$$

where

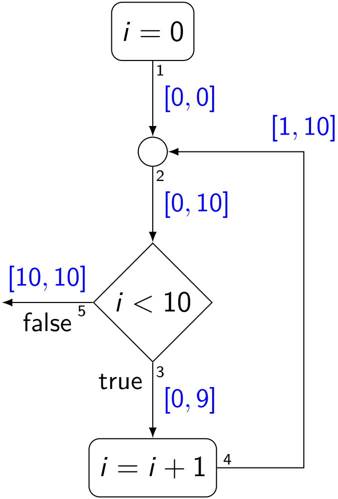
$[l, r]$ is an interval in $\mathbb{Z}$

$\vee$ is convex hull

$\wedge$ is intersection

$+$ is pointwise sum

An example: interval analysis



$$x_1 = [0, 0]$$
$$x_2 = x_1 \vee x_4$$
$$x_3 = x_2 \wedge [-\infty, 9]$$
$$x_4 = x_3 + [1, 1]$$
$$x_5 = x_3 \wedge [10, +\infty]$$

where

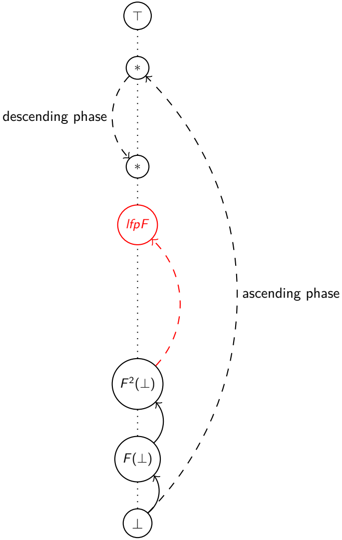
$[l, r]$ is an interval in $\mathbb{Z}$

$\vee$ is convex hull

$\wedge$ is intersection

$+$ is pointwise sum

Two phase analysis



Widening

- A standard chaotic iteration might non terminate
- Introduce widening: accelerate convergence ensuring termination. Replace

$$x_i = \text{expr}$$

with

$$x_i = x_i \nabla \text{expr}$$

- Standard widening on intervals:
  - When a bound is increased, it goes to infinite
  - $\emptyset \nabla [l, r] = [l, r]$
  - $[1, 2] \nabla [1, 3] = [1, +\infty]$

Widening

- A standard chaotic iteration might non terminate
- Introduce widening: accelerate convergence ensuring termination. Replace

$$x_i = \text{expr}$$

  with

$$x_i = x_i \triangledown \text{expr}$$

- Standard widening on intervals:
  - When a bound is increased, it goes to infinite
  - $\emptyset \triangledown [l, r] = [l, r]$
  - $[1, 2] \triangledown [1, 3] = [1, +\infty]$

Widening

- A standard chaotic iteration might non terminate
- Introduce widening: accelerate convergence ensuring termination. Replace

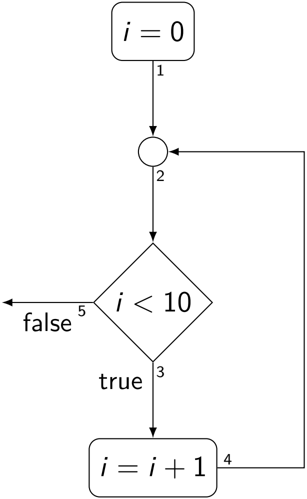
  $$x_i = \mathrm{expr}$$

  with

  $$x_i = x_i \nabla \mathrm{expr}$$

- Standard widening on intervals:
  - When a bound is increased, it goes to infinite
  - $\emptyset \nabla [l, r] = [l, r]$
  - $[1, 2] \nabla [1, 3] = [1, +\infty]$

An example: introducing widening



$$x_1 = [0, 0]$$
$$x_2 = x_2 \nabla (x_1 \vee x_4)$$
$$x_3 = x_2 \wedge [-\infty, 9]$$
$$x_4 = x_3 + [1, 1]$$
$$x_5 = x_3 \wedge [10, +\infty]$$

where

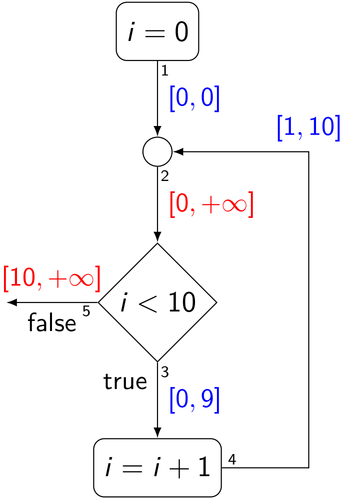
$[l, r]$ is an interval in $\mathbb{Z}$

$\vee$ is convex hull

$\wedge$ is intersection

$+$ is pointwise sum

An example: introducing widening



$$x_1 = [0, 0]$$
$$x_2 = x_2 \triangledown (x_1 \vee x_4)$$
$$x_3 = x_2 \wedge [-\infty, 9]$$
$$x_4 = x_3 + [1, 1]$$
$$x_5 = x_3 \wedge [10, +\infty]$$

where

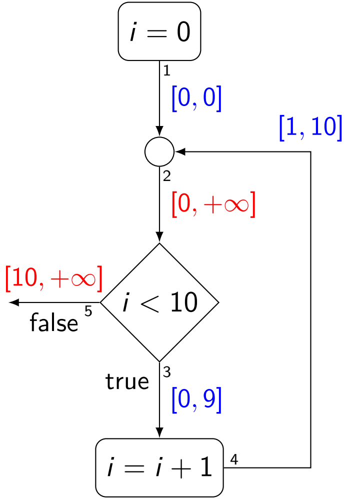
$[l, r]$ is an interval in $\mathbb{Z}$

$\vee$ is convex hull

$\wedge$ is intersection

$+$ is pointwise sum

An example: descending chain



$$
\begin{aligned}
x_1 &= [0, 0] \\
x_2 &= x_1 \vee x_4 \\
x_3 &= x_2 \wedge [-\infty, 9] \\
x_4 &= x_3 + [1, 1] \\
x_5 &= x_3 \wedge [10, +\infty]
\end{aligned}
$$

where

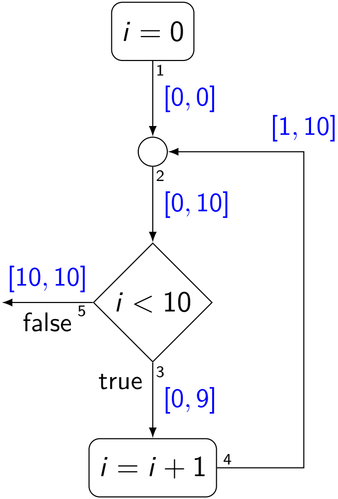
$[l, r]$ is an interval in $\mathbb{Z}$

$\vee$ is convex hull

$\wedge$ is intersection

$+$ is pointwise sum

An example: descending chain



$$x_1 = [0, 0]$$
$$x_2 = x_1 \vee x_4$$
$$x_3 = x_2 \wedge [-\infty, 9]$$
$$x_4 = x_3 + [1, 1]$$
$$x_5 = x_3 \wedge [10, +\infty]$$

where

$[l, r]$ is an interval in $\mathbb{Z}$

$\vee$ is convex hull

$\wedge$ is intersection

$+$ is pointwise sum

Narrowing

- A standard downward chaotic iteration might non terminate
- Either perform a few steps and terminate...
- ...or introduce narrowing: kind of dual of widening

$$x_i = \text{expr}$$

with

$$x_i = x_i \bigtriangleup \text{expr}$$

- Standard narrowing on intervals:
  - only improves infinite bounds
  - $[-\infty, +\infty] \bigtriangleup [l, r] = [l, r]$
  - $[1, +\infty] \bigtriangleup [2, 4] = [1, 4]$

Narrowing

- A standard downward chaotic iteration might non terminate
- Either perform a few steps and terminate...
- ...or introduce narrowing: kind of dual of widening

$$x_i = \text{expr}$$

with

$$x_i = x_i \bigtriangleup \text{expr}$$

- Standard narrowing on intervals:
  - only improves infinite bounds
  - $[-\infty, +\infty] \bigtriangleup [l, r] = [l, r]$
  - $[1, +\infty] \bigtriangleup [2, 4] = [1, 4]$

Narrowing

- A standard downward chaotic iteration might non terminate
- Either perform a few steps and terminate...
- ...or introduce narrowing: kind of dual of widening

$$x_i = \text{expr}$$

  with

$$x_i = x_i \bigtriangleup \text{expr}$$

- Standard narrowing on intervals:
  - only improves infinite bounds
  - $[-\infty, +\infty] \bigtriangleup [l, r] = [l, r]$
  - $[1, +\infty] \bigtriangleup [2, 4] = [1, 4]$

Narrowing

- A standard downward chaotic iteration might non terminate
- Either perform a few steps and terminate...
- ...or introduce narrowing: kind of dual of widening

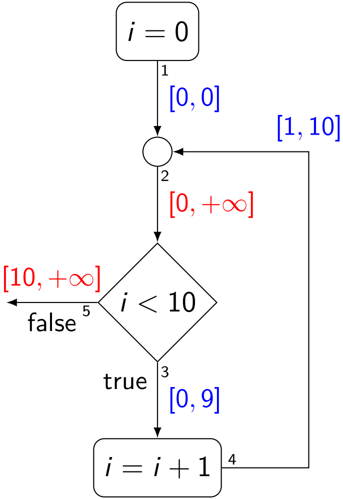
$$x_i = \text{expr}$$

  with

$$x_i = x_i \,\triangle\, \text{expr}$$

- Standard narrowing on intervals:
  - only improves infinite bounds
  - $[-\infty, +\infty] \triangle [l, r] = [l, r]$
  - $[1, +\infty] \triangle [2, 4] = [1, 4]$

An example: introducing narrowing



$$
\begin{aligned}
x_1 &= [0, 0] \\
x_2 &= x_2 \triangle (x_1 \vee x_4) \\
x_3 &= x_2 \wedge [-\infty, 9] \\
x_4 &= x_3 + [1, 1] \\
x_5 &= x_3 \wedge [10, +\infty]
\end{aligned}
$$

where

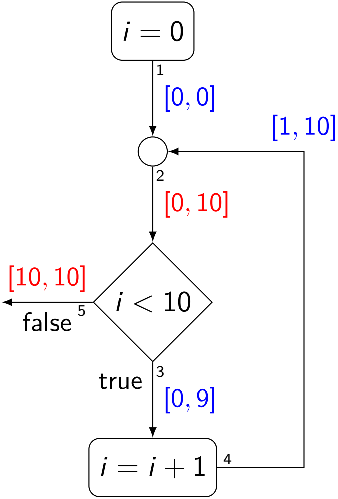
$[l, r]$ is an interval in $\mathbb{Z}$

$\vee$ is convex hull

$\wedge$ is intersection

$+$ is pointwise sum

An example: introducing narrowing



$$x_1 = [0, 0]$$
$$x_2 = x_2 \bigtriangleup (x_1 \vee x_4)$$
$$x_3 = x_2 \wedge [-\infty, 9]$$
$$x_4 = x_3 + [1, 1]$$
$$x_5 = x_3 \wedge [10, +\infty]$$

where

$[l, r]$ is an interval in $\mathbb{Z}$

$\vee$ is convex hull

$\wedge$ is intersection

$+$ is pointwise sum

Is narrowing really needed for termination?

- This is the classical example used to explain widening and narrowing
- ...nonetheless
  - narrowing is not needed in this case
  - descending chain terminates after a single iteration
- **how common is this scenario?**
- We try to answer this question using
  - *Alice benchmarks*
    collection of linear transition systems (called models)
    [`http://alice.cri.ensmp.fr/`]
  - *Jandom*
    abstract interpretation based analyzer we are developing
    [`https://github.com/jandom-devel/Jandom`]

Is narrowing really needed for termination?

- This is the classical example used to explain widening and narrowing
- ...nonetheless
  - narrowing is not needed in this case
  - descending chain terminates after a single iteration
- **how common is this scenario?**
- We try to answer this question using
  - *Alice benchmarks*
    collection of linear transition systems (called models)
    [`http://alice.cri.ensmp.fr/`]
  - *Jandom*
    abstract interpretation based analyzer we are developing
    [`https://github.com/jandom-devel/Jandom`]

Is narrowing really needed for termination?

- This is the classical example used to explain widening and narrowing
- ...nonetheless
  - narrowing is not needed in this case
  - descending chain terminates after a single iteration
- **how common is this scenario?**
- We try to answer this question using
  - *Alice benchmarks*
    collection of linear transition systems (called models)
    [http://alice.cri.ensmp.fr/]
  - *Jandom*
    abstract interpretation based analyzer we are developing
    [https://github.com/jandom-devel/Jandom]

Is narrowing really needed for termination?

- This is the classical example used to explain widening and narrowing
- ...nonetheless
  - narrowing is not needed in this case
  - descending chain terminates after a single iteration
- **how common is this scenario?**
- We try to answer this question using
  - *Alice benchmarks*
    collection of linear transition systems (called models)
    [`http://alice.cri.ensmp.fr/`]
  - *Jandom*
    abstract interpretation based analyzer we are developing
    [`https://github.com/jandom-devel/Jandom`]

Narrowing and the Alice benchmarks

- The two phase analysis with no narrowing:
  - terminates for all the models;
  - during the descending phase, each loop head is evaluated at most 3 times.
- The same is true for the Octagon domain ...
  - octagons are generalization of intervals with constraint of the kind $\pm x \pm y \leq c$ instead of $x \leq c$;
  - octagons and intervals are example of template domains.
- ... but not for the Polyhedra domain
  - the analysis of some models does not terminate;
  - the analysis of some models terminates but gives origin to very long descending chains (more than 100 iterations).

Narrowing and the Alice benchmarks

- The two phase analysis with no narrowing:
  - terminates for all the models;
  - during the descending phase, each loop head is evaluated at most 3 times.
- The same is true for the Octagon domain . . .
  - octagons are generalization of intervals with constraint of the kind $\pm x \pm y \leq c$ instead of $x \leq c$;
  - octagons and intervals are example of template domains.
- . . . but not for the Polyhedra domain
  - the analysis of some models does not terminate;
  - the analysis of some models terminates but gives origin to very long descending chains (more than 100 iterations).

Narrowing and the Alice benchmarks

- The two phase analysis with no narrowing:
  - terminates for all the models;
  - during the descending phase, each loop head is evaluated at most 3 times.
- The same is true for the Octagon domain ...
  - octagons are generalization of intervals with constraint of the kind $\pm x \pm y \leq c$ instead of $x \leq c$;
  - octagons and intervals are example of template domains.
- ... but not for the Polyhedra domain
  - the analysis of some models does not terminate;
  - the analysis of some models terminates but gives origin to very long descending chains (more than 100 iterations).

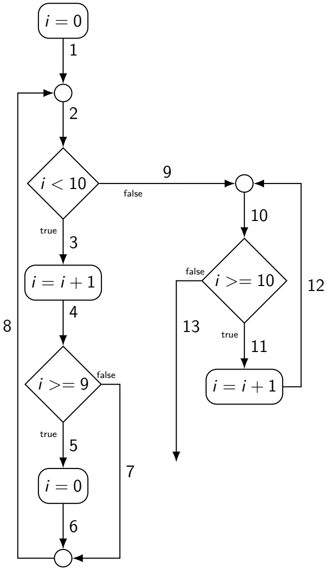
Narrowing and the Alice benchmarks

- The two phase analysis with no narrowing:
  - terminates for all the models;
  - during the descending phase, each loop head is evaluated at most 3 times.
- The same is true for the Octagon domain ...
  - octagons are generalization of intervals with constraint of the kind $\pm x \pm y \leq c$ instead of $x \leq c$;
  - octagons and intervals are example of template domains.
- ... but not for the Polyhedra domain
  - the analysis of some models does not terminate;
  - the analysis of some models terminates but gives origin to very long descending chains (more than 100 iterations).

Narrowing and the Alice benchmarks

- The two phase analysis with no narrowing:
  - terminates for all the models;
  - during the descending phase, each loop head is evaluated at most 3 times.
- The same is true for the Octagon domain ...
  - octagons are generalization of intervals with constraint of the kind $\pm x \pm y \leq c$ instead of $x \leq c$;
  - octagons and intervals are example of template domains.
- ... but not for the Polyhedra domain
  - the analysis of some models does not terminate;
  - the analysis of some models terminates but gives origin to very long descending chains (more than 100 iterations).

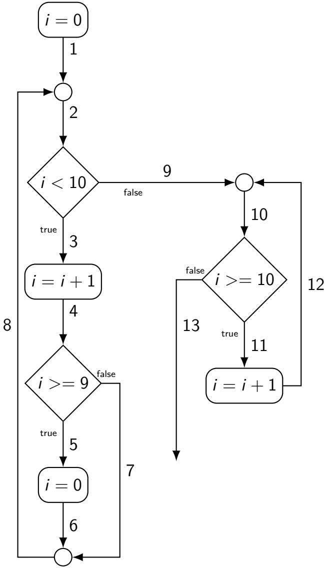
When narrowing is needed
Example program




```
i=0

while   (i<10) {

   i=i+1
   if (i>=9) i=0

}

while   (i>=10) {

   i=i+1

}
```

When narrowing is needed

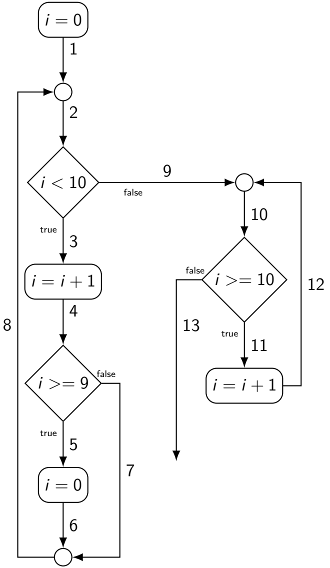
Example program




```
i=0
[x1]
while [x2] (i<10) {
   [x3]
   i=i+1
   if (i>=9) i=0
   [x8]
}
[x9]
while [x10] (i>=10) {
   [x11]
   i=i+1
   [x12]
}
```

When narrowing is needed

Ascending chain



i=0

$[x_1 \rightarrow i = 0]$

while $[x_2 \rightarrow 0 \leq i]$ (i<10) {

  $[x_3 \rightarrow 0 \leq i \leq 9]$

  i=i+1

  if (i>=9) i=0

  $[x_8 \rightarrow 1 \leq i \leq 9]$

}

$[x_9 \rightarrow 10 \leq i]$

while $[x_{10} \rightarrow 10 \leq i\ ]$ (i>=10) {

  $[x_{11} \rightarrow 10 \leq i]$

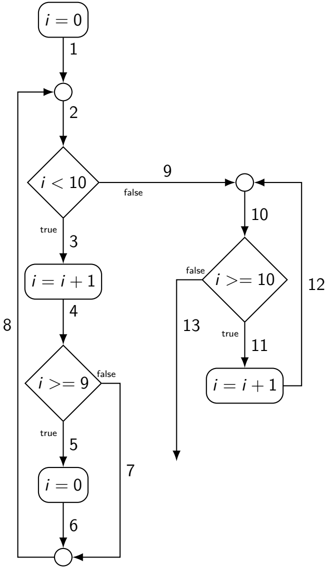
  i=i+1

  $[x_{12} \rightarrow 11 \leq i]$

}

When narrowing is needed

Descending chain 1st loop



i=0

$[x_1 \rightarrow i = 0]$

while $[x_2 \rightarrow 0 \leq i \leq 9]$ (i<10) {

  $[x_3 \rightarrow 0 \leq i \leq 9]$

  i=i+1

  if (i>=9) i=0

  $[x_8 \rightarrow 1 \leq i \leq 9]$

}

$[x_9 \rightarrow \emptyset]$

while $[x_{10} \rightarrow 10 \leq i\,]$ (i>=10) {

  $[x_{11} \rightarrow 10 \leq i]$

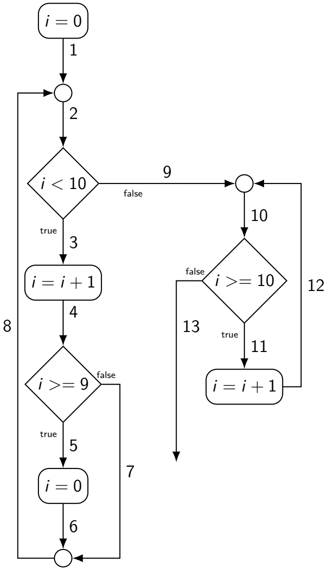
  i=i+1

  $[x_{12} \rightarrow 11 \leq i]$

}

When narrowing is needed

Descending chain 1st loop



i=0

$[x_1 \rightarrow i = 0]$

while $[x_2 \rightarrow 0 \leq i \leq 9]$ (i<10) {

  $[x_3 \rightarrow 0 \leq i \leq 9]$

  i=i+1

  if (i>=9) i=0

With narrowing we would stop here

}

$[x_9 \rightarrow \emptyset]$

while $[x_{10} \rightarrow 10 \leq i\ ]$ (i>=10) {

  $[x_{11} \rightarrow 10 \leq i]$

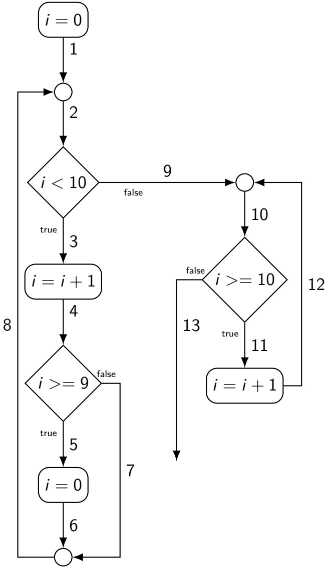
  i=i+1

  $[x_{12} \rightarrow 11 \leq i]$

}

When narrowing is needed

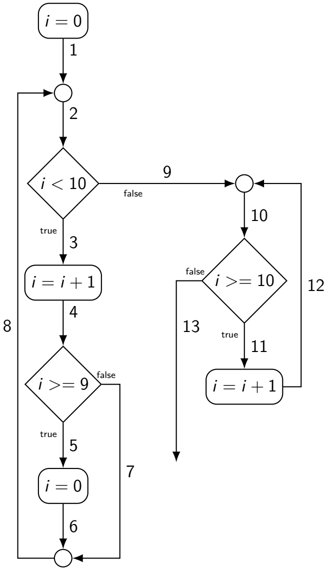
Descending chain 2nd loop 1st iteration



i=0
$[x_1 \rightarrow i = 0]$
while $[x_2 \rightarrow 0 \leq i \leq 9]$ (i<10) {
  $[x_3 \rightarrow 0 \leq i \leq 9]$
  i=i+1
  if (i>=9) i=0
  $[x_8 \rightarrow 1 \leq i \leq 9]$
}
$[x_9 \rightarrow \emptyset]$
while $[x_{10} \rightarrow 11 \leq i]$ (i>=10) {
  $[x_{11} \rightarrow 11 \leq i]$
  i=i+1
  $[x_{12} \rightarrow 12 \leq i]$
}

When narrowing is needed

Descending chain 2nd loop 2nd iteration



i=0
$[x_1 \rightarrow i = 0]$
while $[x_2 \rightarrow 0 \leq i \leq 9]$ (i<10) {
  $[x_3 \rightarrow 0 \leq i \leq 9]$
  i=i+1
  if (i>=9) i=0
  $[x_8 \rightarrow 1 \leq i \leq 9]$
}
$[x_9 \rightarrow \emptyset]$
while $[x_{10} \rightarrow 12 \leq i]$ (i>=10) {
  $[x_{11} \rightarrow 12 \leq i]$
  i=i+1
  $[x_{12} \rightarrow 13 \leq i]$
}

Narrowing and infinite descending chains

- x_{10} follows an infinite descending chain, whose limit is $\emptyset$:

$$[10, +\infty], [11, +\infty], [12, +\infty], \ldots$$

- there are only two kinds of infinite descending chains for intervals with integer bounds:

$$[n_0, +\infty], [n_1, +\infty], [n_2, +\infty], \ldots$$

$$[-\infty, -n_0], [-\infty, -n_1], [-\infty, -n_2], \ldots$$

  with $n_0 < n_1 < n2 < \ldots$

- infinite descending chains may only happen:
  - in presence of unreachable code;
  - when unreachability is detected during the descending phase.

Narrowing and infinite descending chains

- x_{10} follows an infinite descending chain, whose limit is $\emptyset$:

$$[10, +\infty], [11, +\infty], [12, +\infty], \ldots$$

- there are only two kinds of infinite descending chains for intervals with integer bounds:

$$[n_0, +\infty], [n_1, +\infty], [n_2, +\infty], \ldots$$

$$[-\infty, -n_0], [-\infty, -n_1], [-\infty, -n_2], \ldots$$

with $n_0 < n_1 < n2 < \ldots$

- infinite descending chains may only happen:
  - in presence of unreachable code;
  - when unreachability is detected during the descending phase.

Narrowing and infinite descending chains

- x_{10} follows an infinite descending chain, whose limit is $\emptyset$:

$$[10, +\infty], [11, +\infty], [12, +\infty], \ldots$$

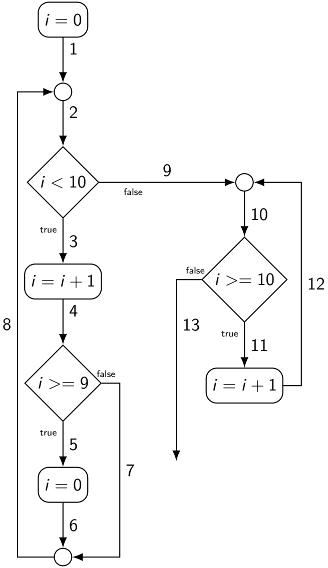
- there are only two kinds of infinite descending chains for intervals with integer bounds:

$$[n_0, +\infty], [n_1, +\infty], [n_2, +\infty], \ldots$$

$$[-\infty, -n_0], [-\infty, -n_1], [-\infty, -n_2], \ldots$$

  with $n_0 < n_1 < n2 < \ldots$
- infinite descending chains may only happen:
  - in presence of unreachable code;
  - when unreachability is detected during the descending phase.

Detecting infinite descending chains

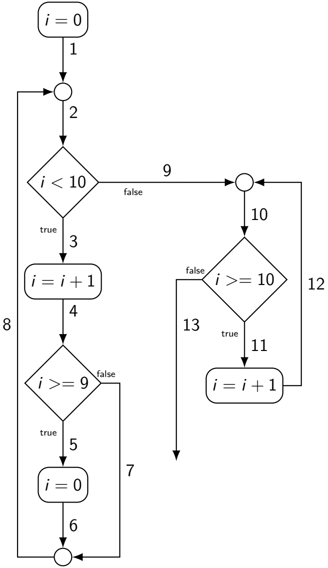


- edge 9 dominates the second loop
  - 9 unreachable $\rightarrow$ 10 unreachable
  - …but $x_{10} = x_9 \vee x_{12}$
  - …and $x_9 = \emptyset \not\rightarrow x_{10} = \emptyset$
- replace $\vee$ with a left-strict variant

$$l_1 \vee^{\emptyset} l_2 = \begin{cases} \emptyset & \text{if } l_1 = \emptyset, \\ l_1 \vee l_2 & \text{otherwise.} \end{cases}$$

- $x_{10} = x_9 \vee^{\emptyset} x_{12}$

Detecting infinite descending chains

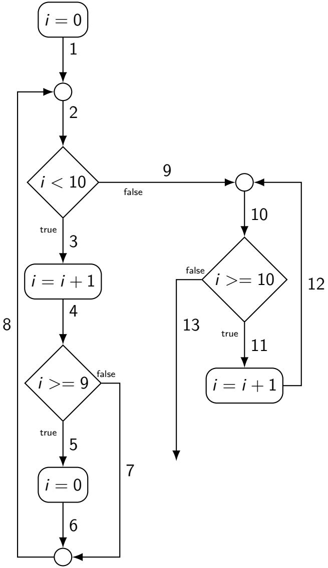


- edge 9 dominates the second loop
  - 9 unreachable $\rightarrow$ 10 unreachable
  - ... but $x_{10} = x_9 \vee x_{12}$
  - ... and $x_9 = \emptyset \not\rightarrow x_{10} = \emptyset$
- replace $\vee$ with a left-strict variant

$$l_1 \vee^{\emptyset} l_2 = \begin{cases} \emptyset & \text{if } l_1 = \emptyset, \\ l_1 \vee l_2 & \text{otherwise.} \end{cases}$$

- $x_{10} = x_9 \vee^{\emptyset} x_{12}$

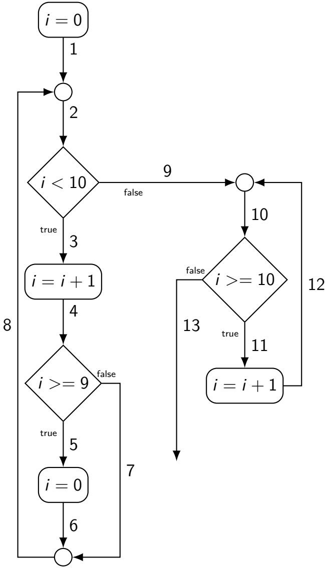
Detecting infinite descending chains



- edge 9 dominates the second loop
  - 9 unreachable $\rightarrow$ 10 unreachable
  - ...but $x_{10} = x_9 \vee x_{12}$
  - ...and $x_9 = \emptyset \not\rightarrow x_{10} = \emptyset$
- replace $\vee$ with a left-strict variant

$$l_1 \vee^{\emptyset} l_2 = \begin{cases} \emptyset & \text{if } l_1 = \emptyset, \\ l_1 \vee l_2 & \text{otherwise.} \end{cases}$$

- $x_{10} = x_9 \vee^{\emptyset} x_{12}$



i=0

$[x_1 \to i = 0]$

while $[x_2 \to 0 \leq i \leq 9]$ (i<10) {

$[x_3 \to 0 \leq i \leq 9]$

i=i+1

if (i>=9) i=0

$[x_8 \to 1 \leq i \leq 9]$

}

$[x_9 \to \emptyset]$

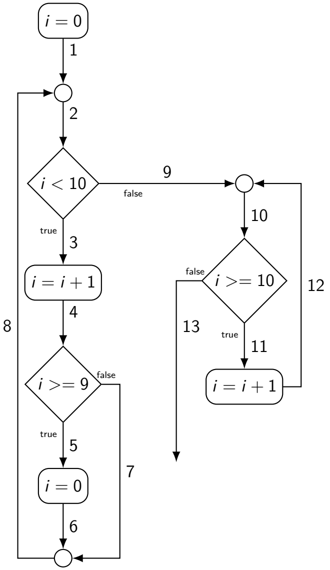
while $[x_{10} \to 10 \leq i\ ]$ (i>=10) {

$[x_{11} \to 10 \leq i]$

i=i+1

$[x_{12} \to 11 \leq i]$

}



i=0

$[x_1 \rightarrow i = 0]$

while $[x_2 \rightarrow 0 \leq i \leq 9]$ (i<10) {

  $[x_3 \rightarrow 0 \leq i \leq 9]$

  i=i+1

  if (i>=9) i=0

  $[x_8 \rightarrow 1 \leq i \leq 9]$

}

$[x_9 \rightarrow \emptyset]$

while $[x_{10} \rightarrow \emptyset]$ (i>=10) {

  $[x_{11} \rightarrow \emptyset]$

  i=i+1

  $[x_{12} \rightarrow \emptyset]$

}

Avoiding narrowing

Theorem

The set of data-flow equations corresponding to a reducible flow-chart may be analyzed without narrowing if we replace $\vee$ in join nodes with $\vee^{\emptyset}$.

- Are we proposing to replace narrowing for a non controlled descending chain?
- Not necessarily, because descending chains may be finite but very long
  - exponentially long in the size of the program

Avoiding narrowing

Theorem

The set of data-flow equations corresponding to a reducible flow-chart may be analyzed without narrowing if we replace $\vee$ in join nodes with $\vee^{\emptyset}$.

- Are we proposing to replace narrowing for a non controlled descending chain?
- Not necessarily, because descending chains may be finite but very long
  - exponentially long in the size of the program

Avoiding narrowing

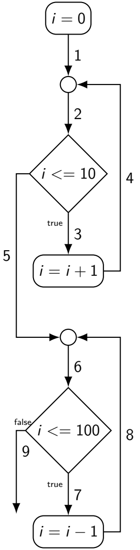
Theorem

The set of data-flow equations corresponding to a reducible flow-chart may be analyzed without narrowing if we replace $\vee$ *in join nodes with* $\vee^{\emptyset}$*.*

- Are we proposing to replace narrowing for a non controlled descending chain?
- Not necessarily, because descending chains may be finite but very long
  - exponentially long in the size of the program

Very long descending chains

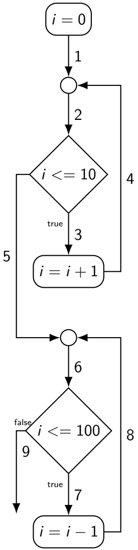
Example program



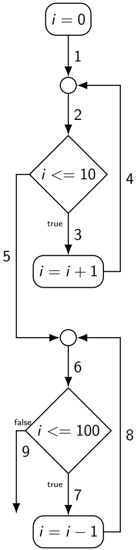

```
i=0
[x1]
while [x2] (i<=10) {
  [x3]
  i=i+1
  [x4]
}
[x5]
while [x6] (i<=100) {
  [x7]
  i=i-1
  [x8]
}
```

Very long descending chains

Ascending chain



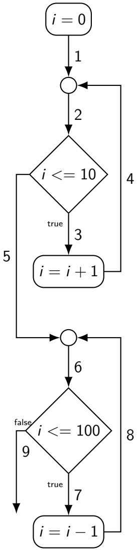
i=0
$[x_1 \rightarrow i = 0]$
while $[x_2 \rightarrow 0 \leq i]$ (i<=10) {
  $[x_3 \rightarrow 0 \leq i \leq 10]$
  i=i+1
  $[x_4 \rightarrow 1 \leq i \leq 11]$
}
$[x_5 \rightarrow 11 \leq i]$
while $[x_6 \rightarrow \mathbb{R}]$ (i<=100) {
  $[x_7 \rightarrow i \leq 100]$
  i=i−1
  $[x_8 \rightarrow i \leq 99]$
}



i=0
$[x_1 \rightarrow i = 0]$
while $[x_2 \rightarrow 0 \leq i \leq 10]$ (i<=10) {
  $[x_3 \rightarrow 0 \leq i \leq 10]$
  i=i+1
  $[x_4 \rightarrow 1 \leq i \leq 11]$
}
$[x_5 \rightarrow i = 11]$
while $[x_6 \rightarrow \mathbb{R}]$ (i<=100) {
  $[x_7 \rightarrow i \leq 100]$
  i=i−1
  $[x_8 \rightarrow i \leq 99]$
}

Very long descending chains

Descending chain 2nd loop 1st iteration



i=0

$[x_1 \to i = 0]$

while $[x_2 \to 0 \leq i \leq 10]$ (i<=10) {

  $[x_3 \to 0 \leq i \leq 10]$

  i=i+1

  $[x_4 \to 1 \leq i \leq 11]$

}

$[x_5 \to i = 11]$

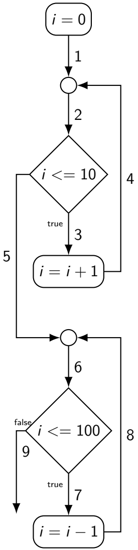
while $[x_6 \to i \leq 99]$ (i<=100) {

  $[x_7 \to i \leq 99]$

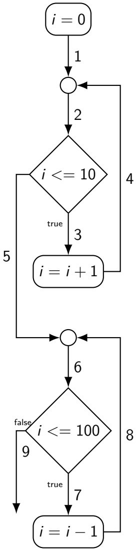
  i=i−1

  $[x_8 \to i \leq 98]$

}




```
i=0
[x1→ i = 0]
while [x2→ 0 ≤ i ≤ 10] (i<=10) {
  [x3→ 0 ≤ i ≤ 10]
  i=i+1
  [x4→ 1 ≤ i ≤ 11]
}
[x5→ i = 11]
while [x6→ i ≤ 98] (i<=100) {
  [x7→ i ≤ 98]
  i=i−1
  [x8→ i ≤ 97]
}
```




```
i=0
[x1→ i = 0]
while [x2→ 0 ≤ i ≤ 10] (i<=10) {
  [x3→ 0 ≤ i ≤ 10]
  i=i+1
  [x4→ 1 ≤ i ≤ 11]
}
[x5→ i = 11]
while [x6→ i ≤ 11] (i<=100) {
  [x7→ i ≤ 11]
  i=i−1
  [x8→ i ≤ 10]
}
```

Rational bounds

When rational bounds are allowed, infinite descending chains may be generated in different ways.

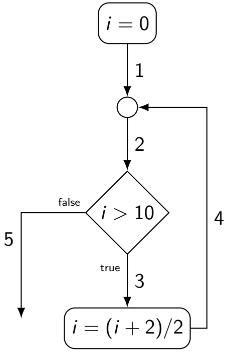
Rational bounds

When rational bounds are allowed, infinite descending chains may be generated in different ways.

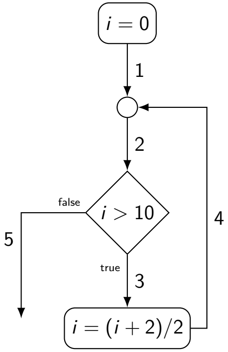
```
i=0
while(i <= 10) {
  i=(i+2)/2
}
```

Infinite descending chains with rational bounds

Example program



i=0
$[x_1]$
while $[x_2]$ (x<=10) {
    $[x_3]$
    i=(i+2)/2
    $[x_4]$
}




```
i=0
```
$[x_1 \rightarrow i = 0]$

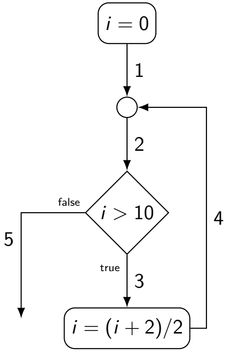
while $[x_2 \rightarrow 0 \leq i]$ (x<=10) {

  $[x_3 \rightarrow 0 \leq i \leq 10]$

  i=(i+2)/2

  $[x_4 \rightarrow 1 \leq i \leq 6]$

}



i=0

$[x_1 \rightarrow i = 0]$

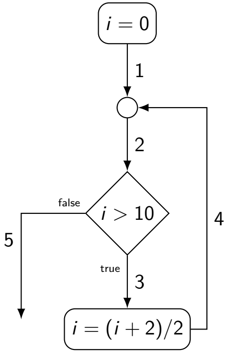
while $[x_2 \rightarrow 0 \leq i \leq 6]$ (x<=10) {

$[x_3 \rightarrow 0 \leq i \leq 6]$

i=(i+2)/2

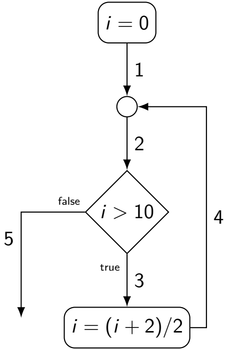
$[x_4 \rightarrow 1 \leq i \leq 4]$

}



i=0
$[x_1 \rightarrow i = 0]$
while $[x_2 \rightarrow 0 \leq i \leq 4]$ (x<=10) {
  $[x_3 \rightarrow 0 \leq i \leq 4]$
  i=(i+2)/2
  $[x_4 \rightarrow 1 \leq i \leq 3]$
}

Infinite descending chains with rational bounds



i=0

$[x_1 \to i = 0]$

while $[x_2 \to 0 \leq i \leq 4]$ (x<=10) {

$[x_3 \to 0 \leq i \leq 4]$

i=(i+2)/2

$[x_4 \to 1 \leq i \leq 3]$

}

x_2 descending chain:

$[0, 6], [0, 4], [0, 3], [0, 5/2], [0, 9/4], \ldots$

whose limit is $[0, 2]$

New narrowings on rationals

- We may define new narrowings which are much more precise by exploiting what we have seen for integer bounds.
- For example $\triangle^1$ is like intersection but we replace bounds with their integer values:

$$I \triangle^1 \emptyset = \emptyset$$

$$[l_1, u_1] \triangle^1 [l_2, u_2] = [l_1, u_1] \wedge [\lfloor l_2 \rfloor, \lceil r_2 \rceil]$$

- $[0, 2] \triangle^1 [0, 1.75] = [0, 2]$, but $[0, 2] \triangle^1 [0, 0.75] = [0, 1]$.
- With $\triangle^1$ narrowing, in the example before we stop at $[0, 3]$.
- Other narrowings are defined in the paper.

New narrowings on rationals

- We may define new narrowings which are much more precise by exploiting what we have seen for integer bounds.
- For example $\triangle^1$ is like intersection but we replace bounds with their integer values:

$$I \triangle^1 \emptyset = \emptyset$$
$$[l_1, u_1] \triangle^1 [l_2, u_2] = [l_1, u_1] \wedge [\lfloor l_2 \rfloor, \lceil r_2 \rceil]$$

- $[0, 2] \triangle^1 [0, 1.75] = [0, 2]$, but $[0, 2] \triangle^1 [0, 0.75] = [0, 1]$.
- With $\triangle^1$ narrowing, in the example before we stop at $[0, 3]$.
- Other narrowings are defined in the paper.

New narrowings on rationals

- We may define new narrowings which are much more precise by exploiting what we have seen for integer bounds.
- For example $\triangle^1$ is like intersection but we replace bounds with their integer values:

$$I \triangle^1 \emptyset = \emptyset$$

$$[l_1, u_1] \triangle^1 [l_2, u_2] = [l_1, u_1] \wedge [\lfloor l_2 \rfloor, \lceil r_2 \rceil]$$

- $[0, 2] \triangle^1 [0, 1.75] = [0, 2]$, but $[0, 2] \triangle^1 [0, 0.75] = [0, 1]$.
- With $\triangle^1$ narrowing, in the example before we stop at $[0, 3]$.
- Other narrowings are defined in the paper.

New narrowings on rationals

- We may define new narrowings which are much more precise by exploiting what we have seen for integer bounds.
- For example $\triangle^1$ is like intersection but we replace bounds with their integer values:

$$I \triangle^1 \emptyset = \emptyset$$

$$[l_1, u_1] \triangle^1 [l_2, u_2] = [l_1, u_1] \wedge [\lfloor l_2 \rfloor, \lceil r_2 \rceil]$$

- $[0, 2] \triangle^1 [0, 1.75] = [0, 2]$, but $[0, 2] \triangle^1 [0, 0.75] = [0, 1]$.
- With $\triangle^1$ narrowing, in the example before we stop at $[0, 3]$.
- Other narrowings are defined in the paper.

New narrowings on rationals

- We may define new narrowings which are much more precise by exploiting what we have seen for integer bounds.
- For example $\triangle^1$ is like intersection but we replace bounds with their integer values:

$$I \triangle^1 \emptyset = \emptyset$$

$$[l_1, u_1] \triangle^1 [l_2, u_2] = [l_1, u_1] \wedge [\lfloor l_2 \rfloor, \lceil r_2 \rceil]$$

- $[0, 2] \triangle^1 [0, 1.75] = [0, 2]$, but $[0, 2] \triangle^1 [0, 0.75] = [0, 1]$.
- With $\triangle^1$ narrowing, in the example before we stop at $[0, 3]$.
- Other narrowings are defined in the paper.

Conclusions

For all template domains:

- infinite descending chains exist but are rare
- very long descending chains exist but are rare
- if you only care about termination and you can afford very long descending chains:
  - use $\nabla^{\emptyset}$ on loops;
  - use $\triangle^{1}$ when bounds are rational numbers.
- if you want to avoid long descending chains, fix maximum number of descending steps

Conclusions

For all template domains:

- infinite descending chains exist but are rare
- very long descending chains exist but are rare
- if you only care about termination and you can afford very long descending chains:
  - use $\triangledown^{\emptyset}$ on loops;
  - use $\triangle^1$ when bounds are rational numbers.
- if you want to avoid long descending chains, fix maximum number of descending steps

Conclusions

For all template domains:

- infinite descending chains exist but are rare
- very long descending chains exist but are rare
- if you only care about termination and you can afford very long descending chains:
  - use $\triangledown^{\emptyset}$ on loops;
  - use $\triangle^{1}$ when bounds are rational numbers.
- if you want to avoid long descending chains, fix maximum number of descending steps

Conclusions

For all template domains:

- infinite descending chains exist but are rare
- very long descending chains exist but are rare
- if you only care about termination and you can afford very long descending chains:
  - use $\vee^{\emptyset}$ on loops;
  - use $\triangle^1$ when bounds are rational numbers.
- if you want to avoid long descending chains, fix maximum number of descending steps

Future work

- Perform further experiments on real programs instead of linear transition systems.
- Not only non-terminating descending chains are rare, but also the number of descending steps is generally quite low. Why?
- What about backward analysis?

Future work

- Perform further experiments on real programs instead of linear transition systems.
- Not only non-terminating descending chains are rare, but also the number of descending steps is generally quite low. Why?
- What about backward analysis?

Future work

- Perform further experiments on real programs instead of linear transition systems.
- Not only non-terminating descending chains are rare, but also the number of descending steps is generally quite low. Why?
- What about backward analysis?

Thank You!

Related Work

- performing descending chains without narrowing is common
  - termination ensured by fixing a limit on the number of descending steps;
  - we prove this limit may be removed most of the time.
- the same effect of $\vee^{\emptyset}$ may be realized using *localized narrowing with restart* [SAS '13]
  - localized narrowing has a much greater impact on performance and precision;
  - requires many more changes to existing analyzers.

Related Work

- performing descending chains without narrowing is common
  - termination ensured by fixing a limit on the number of descending steps;
  - we prove this limit may be removed most of the time.
- the same effect of $\vee^{\emptyset}$ may be realized using *localized narrowing with restart* [SAS '13]
  - localized narrowing has a much greater impact on performance and precision;
  - requires many more changes to existing analyzers.