Organization

Part 1: Constraint Systems

Part 2: Widening and Narrowing

Widening and Narrowing

Observation 1

Cousot, Cousot 1977

- Program invariants are post-solutions of systems of equations over suitable lattices.

Observation 1

Cousot, Cousot 1977

- Program invariants are post-solutions of systems of equations over suitable lattices.
- Non-trivial program invariants require lattices with infinite ascending chains.

Observation 1

Cousot, Cousot 1977

- Program invariants are post-solutions of systems of equations over suitable lattices.
- Non-trivial program invariants require lattices with infinite ascending chains.
- Kleene fixpoint iteration will often not terminate.

Observation 1

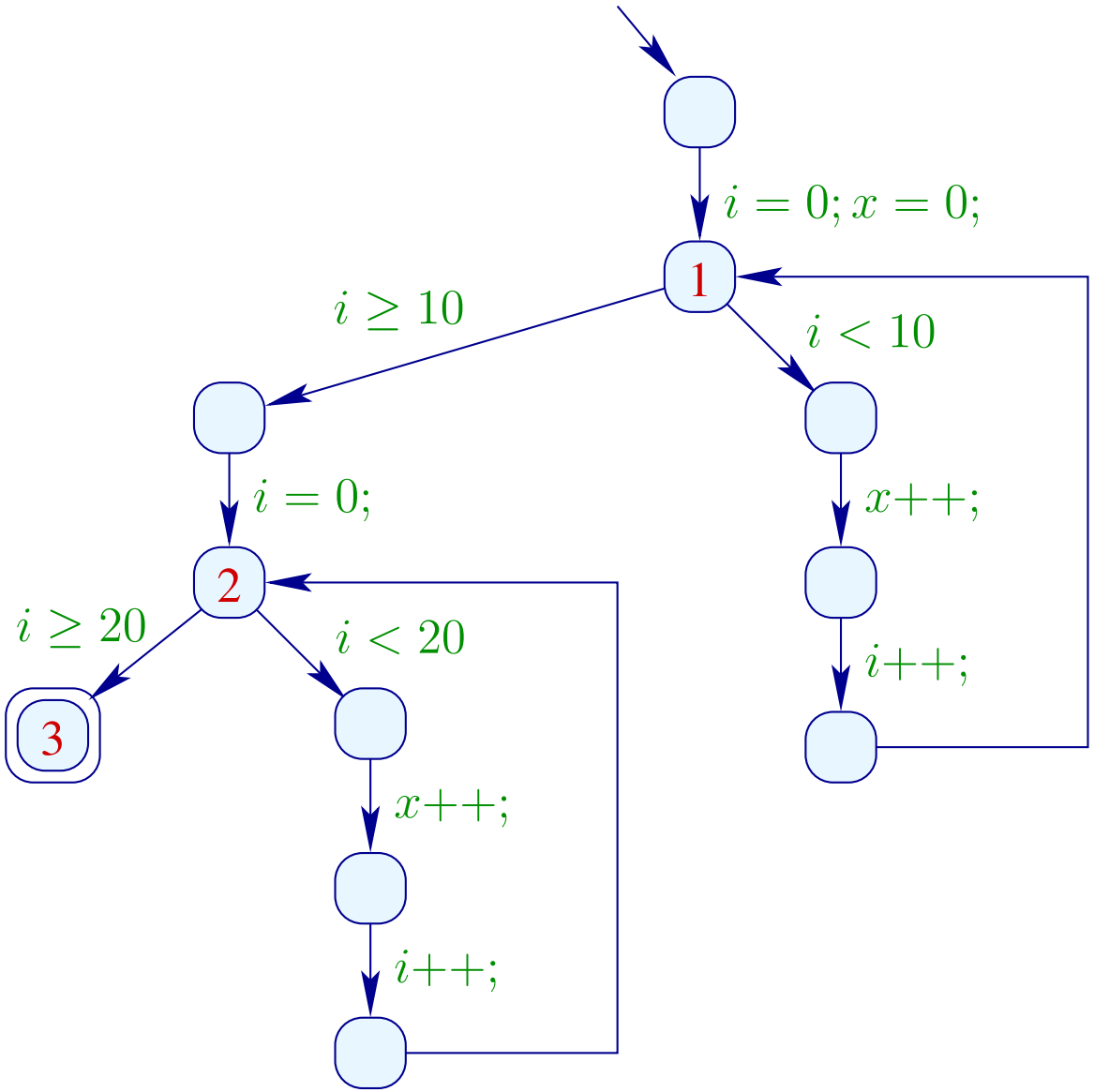
Cousot, Cousot 1977

- Program invariants are post-solutions of systems of equations over suitable lattices.
- Non-trivial program invariants require lattices with infinite ascending chains.
- Kleene fixpoint iteration will often not terminate.
- Widening allows to enforce termination—at the price of giving up precision.

Observation 1

Cousot, Cousot 1977

- Program invariants are post-solutions of systems of equations over suitable lattices.
- Non-trivial program invariants require lattices with infinite ascending chains.
- Kleene fixpoint iteration will often not terminate.
- Widening allows to enforce termination—at the price of giving up precision.
- Some of the precision subsequently can may be recovered through narrowing ...


$i = 0; x = 0;$
1
$i \geq 10$
$i < 10$
$i = 0;$
$x{+}{+};$
2
$i \geq 20$
$i < 20$
$i{+}{+};$
3
$x{+}{+};$
$i{+}{+};$

Example Analysis

reduced product of

- intervals
- linear equalities

$\Longrightarrow$ infinite ascending chains

$\Longrightarrow$ widening/narrowing required

Widening

Equations

$$x_i = f_i(x_1, \ldots, x_n) \ , \quad i = 1, \ldots, n$$

Widening

Equations

$$x_i = x_i \sqcup f_i(x_1, \ldots, x_n) \;, \quad i = 1, \ldots, n$$

Widening

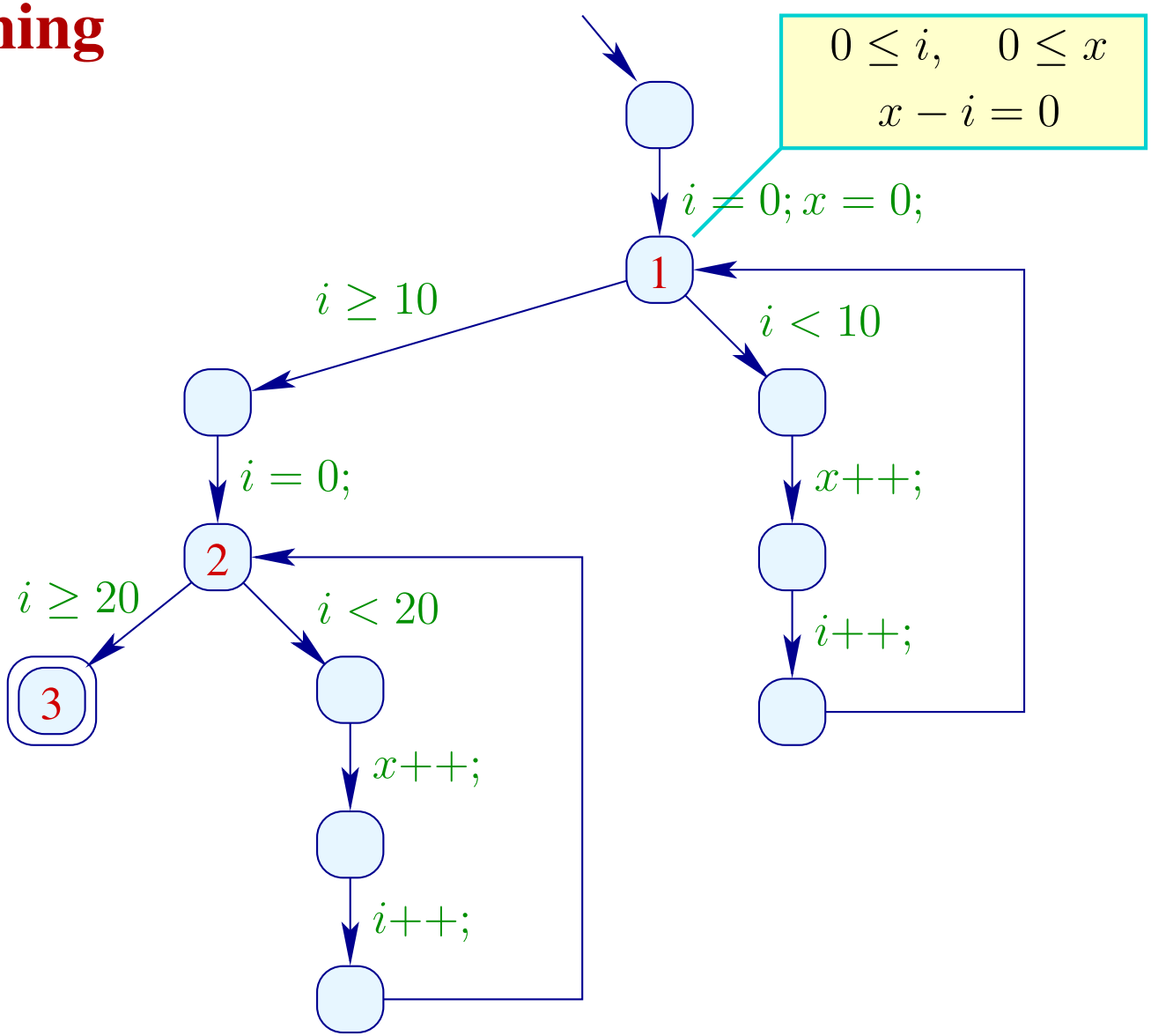
Equations

$$x_i = x_i \uplus f_i(x_1, \ldots, x_n) \;, \quad i = 1, \ldots, n$$

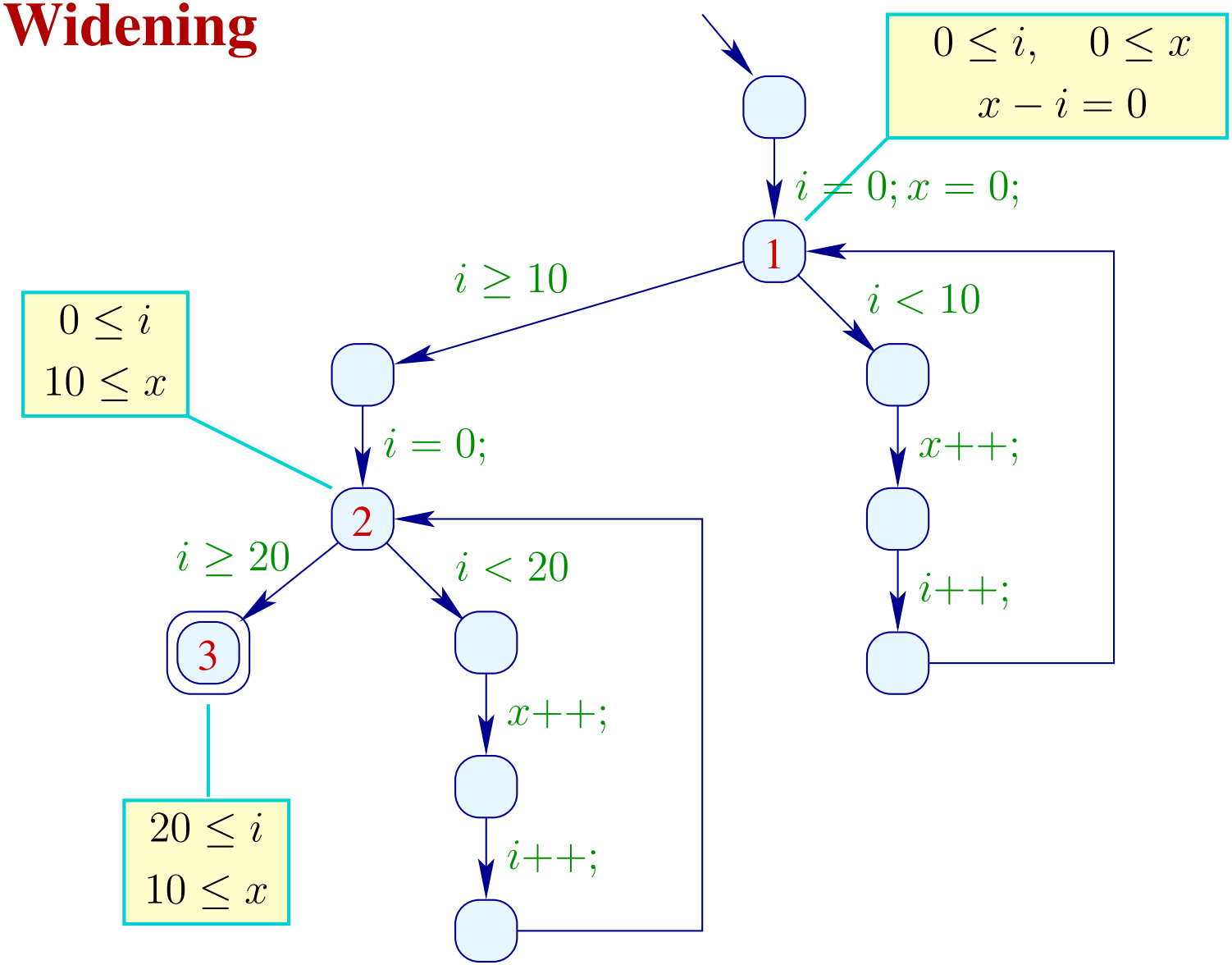
Widening operator

$$a \sqcup b \sqsubseteq a \uplus b$$

Widening



Widening


$0 \le i, \quad 0 \le x$
$x - i = 0$
$i = 0; x = 0;$
1
$i \ge 10$
$i < 10$
$0 \le i$
$10 \le x$
$i = 0;$
$x{+}{+};$
2
$i{+}{+};$
$i \ge 20$
$i < 20$
3
$x{+}{+};$
$20 \le i$
$10 \le x$
$i{+}{+};$

Narrowing

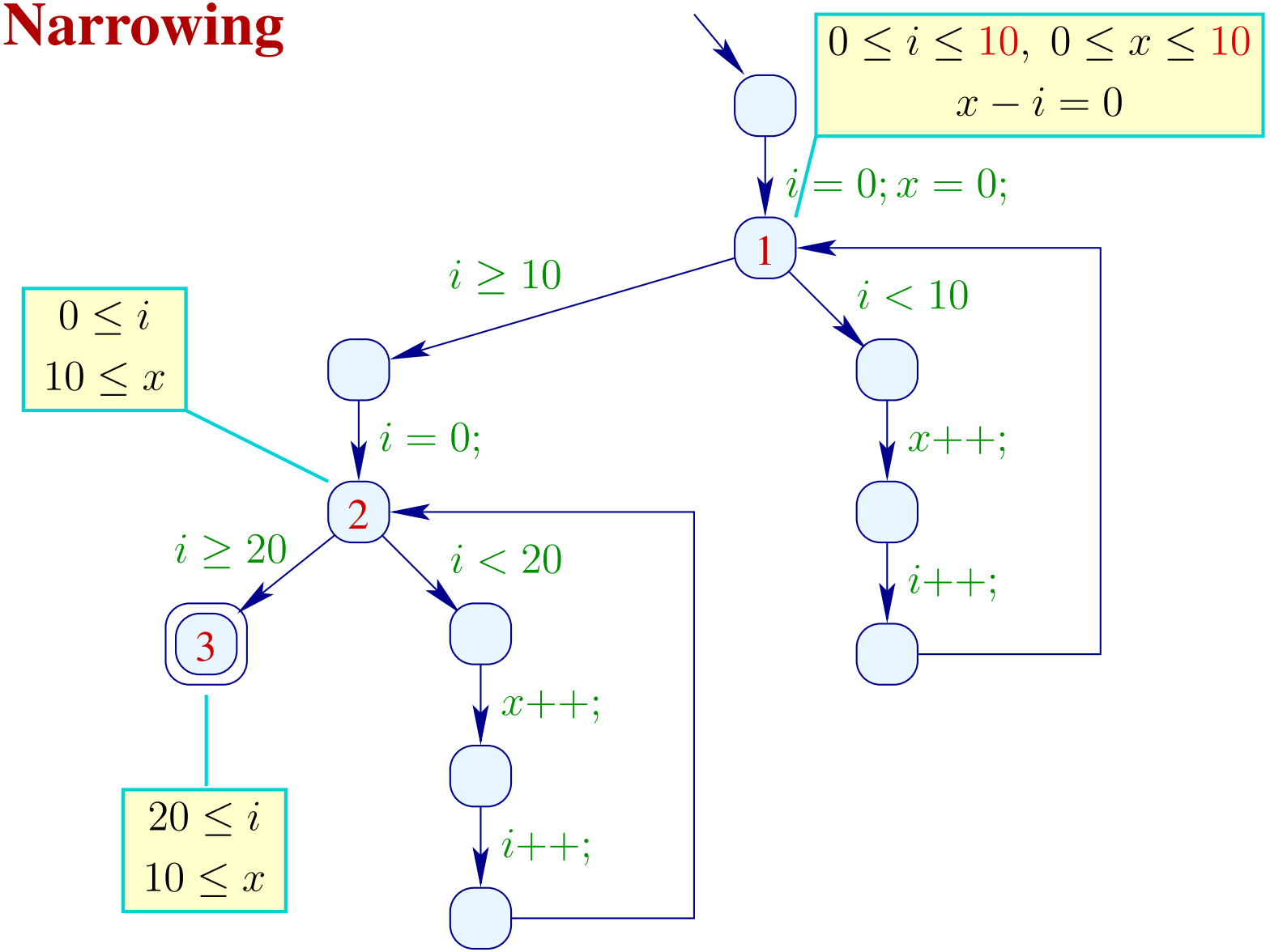
Equations

$$x_i = x_i \sqcap f_i(x_1, \ldots, x_n) \;, \quad i = 1, \ldots, n$$

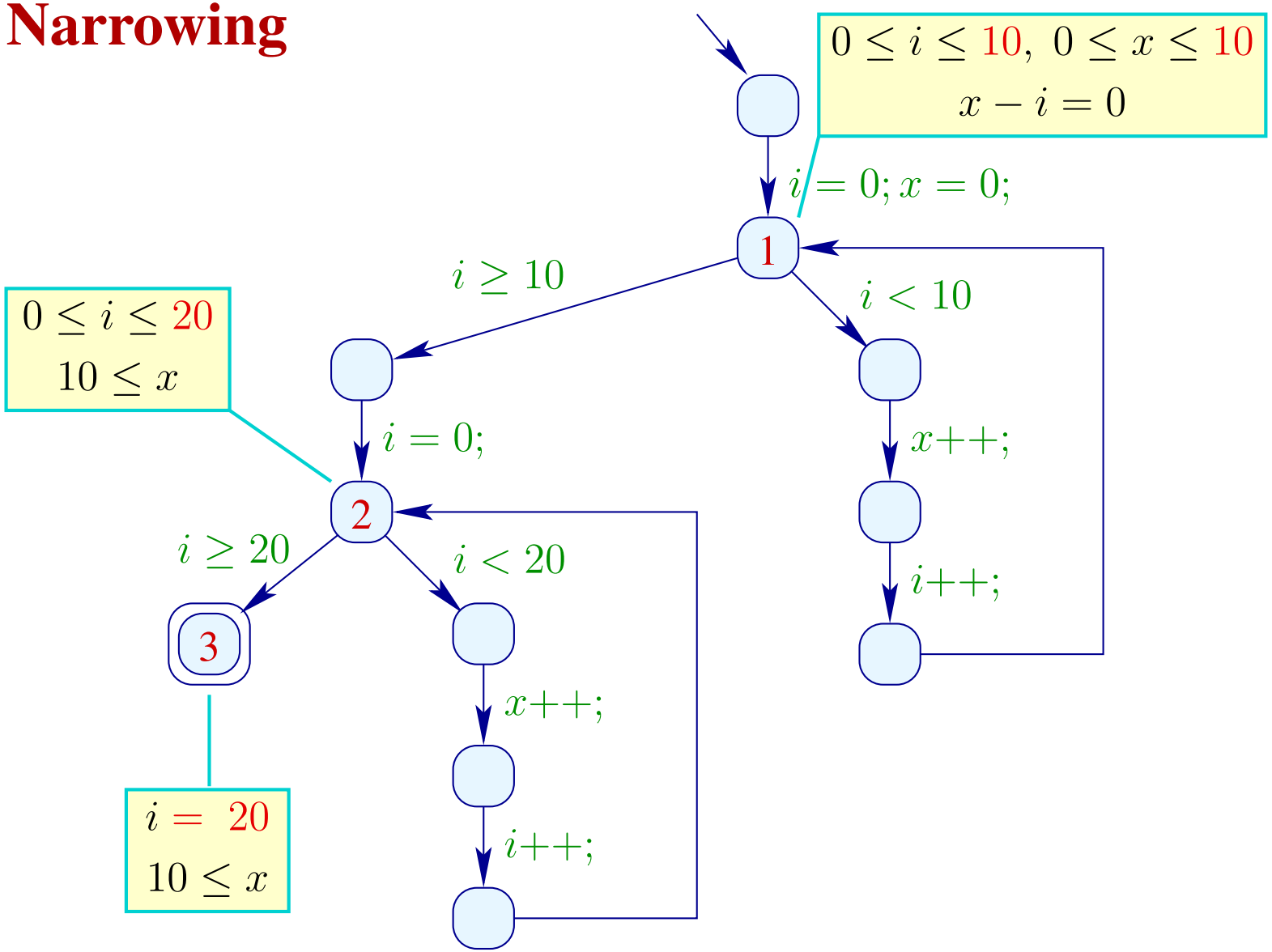
Narrowing operator

$$b \sqsubseteq a \quad \Longrightarrow \quad b \sqsubseteq a \sqcap b \sqsubseteq a$$

Narrowing


$0 \le i \le 10,\ 0 \le x \le 10$
$x - i = 0$
$i = 0; x = 0;$
1
$i \ge 10$
$i < 10$
$0 \le i$
$10 \le x$
$i = 0;$
$x{+}{+};$
2
$i{+}{+};$
$i \ge 20$
$i < 20$
3
$x{+}{+};$
$i{+}{+};$
$20 \le i$
$10 \le x$

Narrowing


$0 \le i \le 10,\ 0 \le x \le 10$
$x - i = 0$
$i = 0; x = 0;$
1
$i \ge 10$
$i < 10$
$0 \le i \le 20$
$10 \le x$
$i = 0;$
$x{+}{+};$
2
$i{+}{+};$
$i \ge 20$
$i < 20$
3
$x{+}{+};$
$i{+}{+};$
$i = 20$
$10 \le x$

Problem

- Precision, once abandoned, is difficult to recover.
- Narrowing is only partially defined and requires:
  - ... equations
  - ... monotonic right-hand sides

Problem

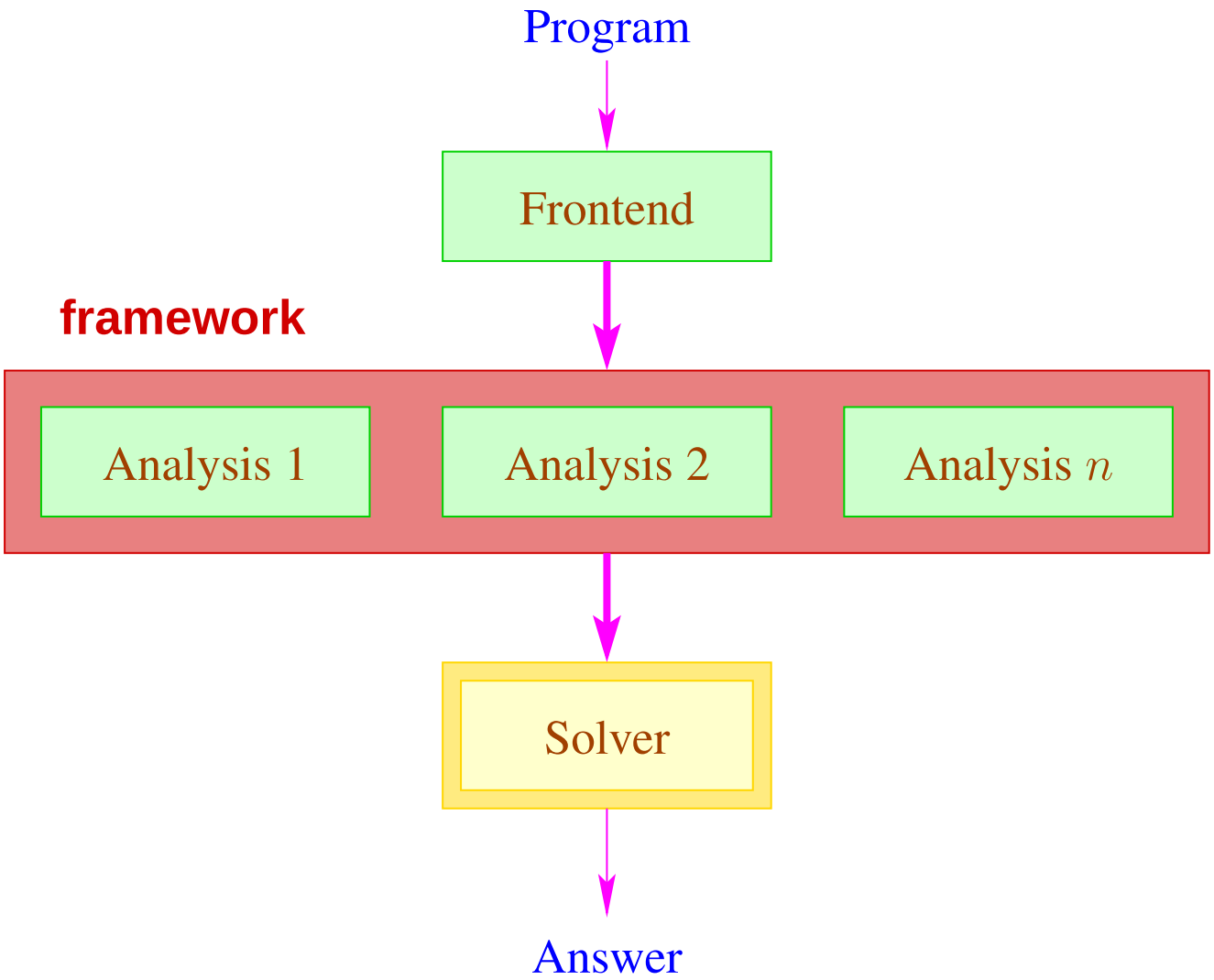
- Precision, once abandoned, is difficult to recover.
- Narrowing is only partially defined and requires:

  ... equations

  ... monotonic right-hand sides

  $\Longrightarrow$ not met by interprocedural analysis ...

Goblint:


Program
Frontend
framework
Analysis 1
Analysis 2
Analysis n
Solver
Answer

Observation 2

Analysis	Constraint system
no context	finite monotonic finite dependences
context	infinite non-monotonic finite dependences
partial context	infinite non-monotonic infinite dependences

Observation 2

Analysis	Constraint system
no context	finite monotonic finite dependences
context	infinite non-monotonic finite dependences
partial context	infinite non-monotonic infinite dependences

... 2-phase algorithm is unsound !

Overview

- Combining widening and narrowing
- Termination
- Evaluation
- Enhancements

Combining Widening and Narrrowing

Equations

$$x_i = x_i \boxplus f_i(x_1, \ldots, x_n) \ , \quad i = 1, \ldots, n$$

Combining Widening and Narrrowing

Equations

$$x_i = x_i \boxplus f_i(x_1, \ldots, x_n) \;, \quad i = 1, \ldots, n$$

Combined operator

$$a \boxplus b = \begin{array}{l} \textbf{if } b \sqsubseteq a \textbf{ then } a \boxplus b \\ \textbf{else } a \sqcup b \end{array}$$

Idea

- Perform one joint iteration !
- Iterate until stabilization !

Idea

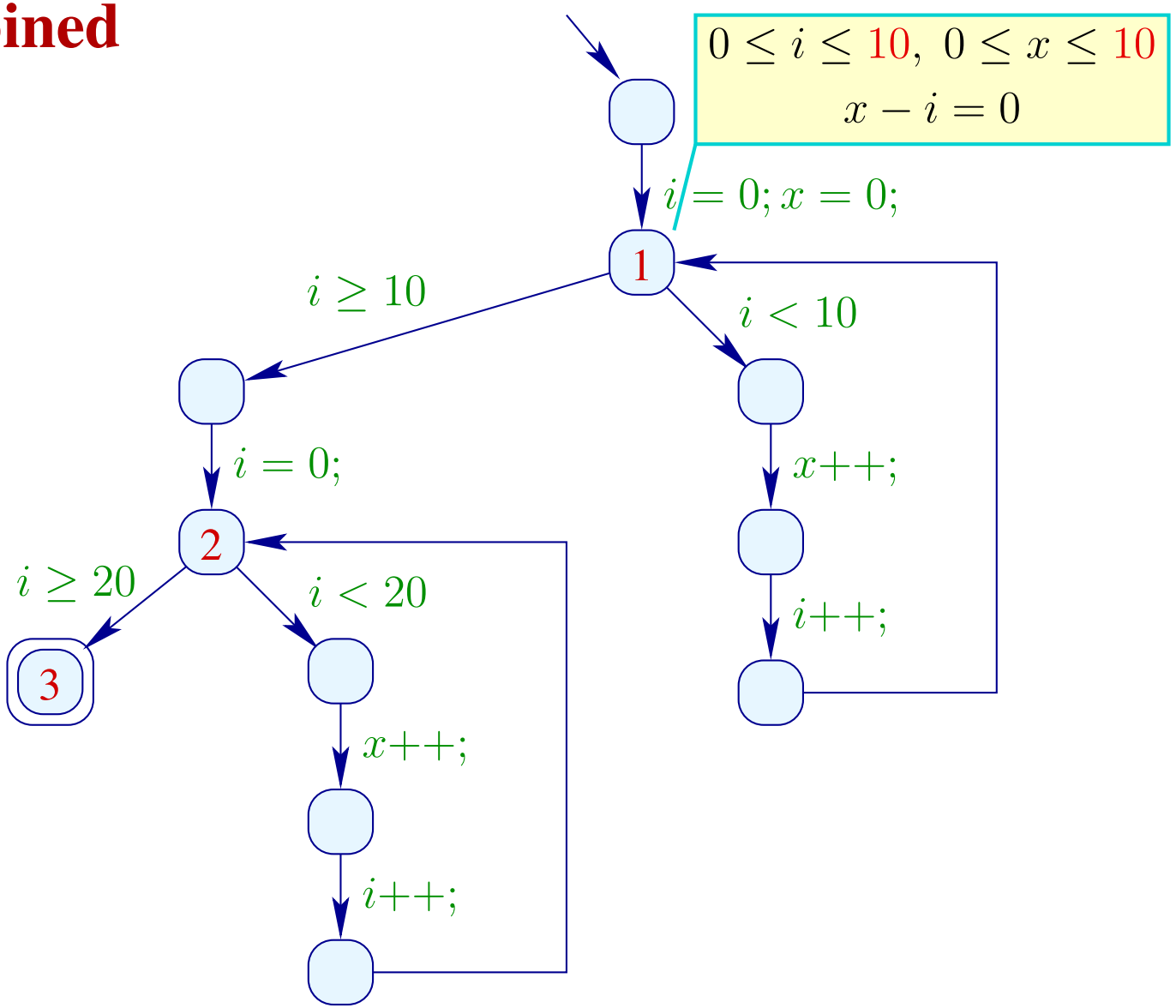
- Perform one joint iteration !
- Iterate until stabilization !

Theorem

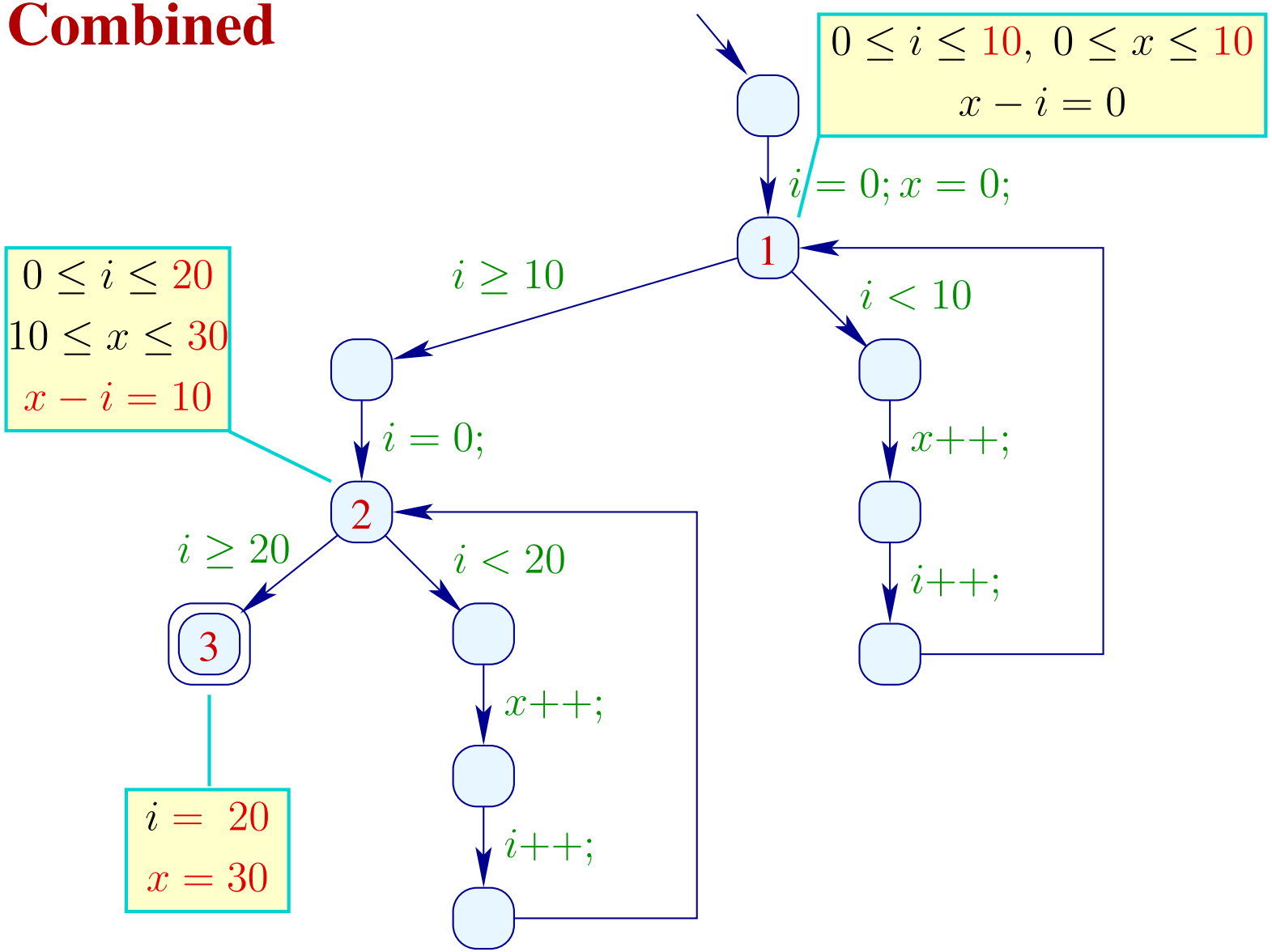
Assume that the fixpoint algorithm performs a sequence of atomic evaluations of right-hand sides. Then

- Upon termination, a post-solution is attained ...
- This holds independent of the start values !

Combined



Combined


$0 \le i \le 10,\ 0 \le x \le 10$
$x - i = 0$
$i = 0; x = 0;$
1
$i \ge 10$
$i < 10$
$0 \le i \le 20$
$10 \le x \le 30$
$x - i = 10$
$i = 0;$
$x{+}{+};$
2
$i \ge 20$
$i < 20$
$i{+}{+};$
3
$x{+}{+};$
$i{+}{+};$
$i = 20$
$x = 30$

Termination

- Any solver performing chaotic fixpoint iteration can be enhanced to a ⊟-solver.
- Termination, though, can no longer be guaranteed—even if right-hand sides are monotonic ...

Round robin iteration

$$\begin{aligned} x_1 &= x_2 \\ x_2 &= x_3 + 1 \\ x_3 &= x_1 \end{aligned}$$

for $\mathtt{N} \cup \{\infty\}$ with $\leq$

Round robin iteration

$$\begin{aligned} x_1 &= x_2 \\ x_2 &= x_3 + 1 \\ x_3 &= x_1 \end{aligned}$$

for $\mathbb{N} \cup \{\infty\}$ with $\leq$ results in:

	0	1	2	3	4	5	6
x_1	0	0	∞	1	∞	2	$\ldots$
x_2	0	∞	1	∞	2	∞	$\ldots$
x_3	0	0	∞	1	∞	2	$\ldots$

Work list iteration

$$
\begin{aligned}
x_1 &= x_1 + 1 \wedge x_2 + 1 \\
x_2 &= x_2 + 1 \wedge x_1 + 1
\end{aligned}
$$

Work list iteration

$$\begin{aligned} x_1 &= x_1 + 1 \wedge x_2 + 1 \\ x_2 &= x_2 + 1 \wedge x_1 + 1 \end{aligned}$$

results in:

	1	2	3	4	5	6	7	8	
W	$[x_1, x_2]$	$[x_1, x_2]$	$[x_1, x_2]$	$[x_2]$	$[x_2, x_1]$	$[x_2, x_1]$	$[x_1]$	$[x_1, x_2]$	
x_1	0	∞	1	1	1	1	1	∞	...
x_2	0	0	0	0	∞	2	2	2	...

Bad News

Standard solvers fail to terminate already for trivial systems of equations.

Bad News

Standard solvers fail to terminate already for trivial systems of equations.

Good News

Variations of standard solvers are guaranteed to terminate—whenever right-hand sides are monotonic ...

Structured RR

```
void solve i {
    if (i = 0) return;
    solve (i − 1);
    new ← ρ(x_i) ⊟ f_i ρ;
    if (ρ(x_i) ≠ new) {
        ρ(x_i) ← new;
        solve i;
    }
}
```

Structured RR (cont.)

$$\begin{aligned} x_1 &= x_2 \\ x_2 &= x_3 + 1 \\ x_3 &= x_1 \end{aligned}$$

results in:

	0	1	2	3	4	5	6	7
x_1	0	0	∞	∞	1	1	1	∞
x_2	0	∞	∞	1	1	1	∞	∞
x_3	0	0	0	0	0	∞	∞	∞

// SRR is a mild generalization of iterating

// according to the syntax

// SRR is a mild generalization of iterating

// according to the syntax

Theorem

(1) If SRR terminates, a post-solution has been found.

(2) SRR terminates, whenever all right-hand sides are monotonic.

Complexity Issues

- If $\sqcup\!\!\!\!-\!\!\!\!\sqcup = \sqcup$ and $\sqcap\!\!\!\!-\!\!\!\!\sqcap = \pi_1$, then SRR may perform in the worst-case, only half as many iterations as RR.

Complexity Issues

- If $\sqcup\!\!\!-\!\!\!\sqcup = \sqcup$ and $\sqcap\!\!\!-\!\!\!\sqcap = \pi_1$, then SRR may perform in the worst-case, only half as many iterations as RR.
- For arbitrary W/N operators and monotonic systems, it may perform exponentially many evaluations

  // although we never observed that

Complexity Issues

- If $\sqcup\!\!\!\!\sqcup = \sqcup$ and $\sqcap\!\!\!\!\sqcap = \pi_1$, then SRR may perform in the worst-case, only half as many iterations as RR.
- For arbitrary W/N operators and monotonic systems, it may perform exponentially many evaluations

  // although we never observed that
- For non-monotonic systems, it still may not terminate

  // which we also have not observed.

Structured W

$Q \leftarrow \emptyset$; **for** $(i \leftarrow 1; i \leq n; i{+}{+})$ add Q x_i;
while $(Q \neq \emptyset)$ {
  $x_i \leftarrow$ extract_min (Q);
  $new \leftarrow \rho(x_i)$ ▤ $f_i\,\rho$;
  if $(\rho(x_i) \neq new)$ {
    $\rho(x_i) \leftarrow new$; add Q x_i;
    forall $(x_j \in \mathsf{infl}_i)$ add Q x_j;
  }
}

Structured W

$Q \leftarrow \emptyset$; **for** $(i \leftarrow 1; i \leq n; i{+}{+})$ add Q x_i;

while $(Q \neq \emptyset)$ { // Q priority queue

  $x_i \leftarrow$ extract_min (Q);

  $new \leftarrow \rho(x_i)$ ▤ $f_i\,\rho$;

  if $(\rho(x_i) \neq new)$ {

    $\rho(x_i) \leftarrow new$; add Q x_i;

    forall $(x_j \in \mathsf{infl}_i)$ add Q x_j;

  }

}

Structured W (cont.)

$$\begin{aligned} x_1 &= x_1 + 1 \wedge x_2 + 1 \\ x_2 &= x_2 + 1 \wedge x_1 + 1 \end{aligned}$$

results in:

	1	2	3	4	5	6	7	8
Q	$[x_1, x_2]$	$[x_1, x_2]$	$[x_1, x_2]$	$[x_2]$	$[x_1, x_2]$	$[x_1, x_2]$	$[x_2]$	[]
x_1	0	∞	1	1	1	∞	∞	∞
x_2	0	0	0	0	∞	∞	∞	∞

Theorem

(1) If SW terminates, a post-solution has been found.

(2) SW terminates, whenever all right-hand sides are monotonic.

Complexity Issues

- If $\sqcup\!\!\!\!-\!\!\!\!\sqcup = \sqcup$ and $\sqcap\!\!\!\!-\!\!\!\!\sqcap = \pi_1$, then SW may perform as many iterations as SW.

Complexity Issues

- If $\sqcup\!\!\!-\!\!\!\sqcup = \sqcup$ and $\sqcap\!\!\!-\!\!\!\sqcap = \pi_1$, then SW may perform as many iterations as SW.
- For arbitrary W/N operators and monotonic systems, it may perform exponentially many evaluations

  // although we never observed that in practice
- For non-monotonic systems, it still may not terminate

  // which we also have not observed.

Local Solving

- Context-sensitive interprocedural analysis for complex invariants gives rise to infinite equation systems with dynamic variable dependences

  $\Longrightarrow$ local fixpoint iteration

Local Solving

- Context-sensitive interprocedural analysis for complex invariants gives rise to infinite equation systems with dynamic variable dependences

  $\Longrightarrow$ local fixpoint iteration

- Partial contexts and the combination with flow-insensitive analysis can conveniently be handled using side-effecting ...

Local Solving (cont.)

- Two-phase widening/narrowing neither works well with local solving nor with side-effecting.

Local Solving (cont.)

- Two-phase widening/narrowing neither works well with local solving nor with side-effecting.
- The local side-effecting solver, e.g., in Goblint does not perform chaotic fixpoint iteration.

Local Solving (cont.)

- Two-phase widening/narrowing neither works well with local solving nor with side-effecting.
- The local side-effecting solver, e.g., in Goblint does not perform chaotic fixpoint iteration

$\Longrightarrow$ new ⊟-solver SLR^+

Issues

Local solver SLR$^+$ does not perform any preprocessing of system of equations.

$\Rightarrow$ Variable dependences are detected on the fly.

Issues

Local solver SLR$^+$ does not perform any preprocessing of system of equations.

$\Rightarrow$ Variable dependences are detected on the fly.

$\Rightarrow$ Priorities must be assigned on the fly.

Issues

Local solver SLR$^+$ does not perform any preprocessing of system of equations.

$\Rightarrow$ Variable dependences are detected on the fly.

$\Rightarrow$ Priorities must be assigned on the fly.

$\Rightarrow$ Loop heads for placing W/N also must be detected on the fly.

Issues

Local solver SLR^+ does not perform any preprocessing of system of equations.

$\Rightarrow$ Variable dependences are detected on the fly.

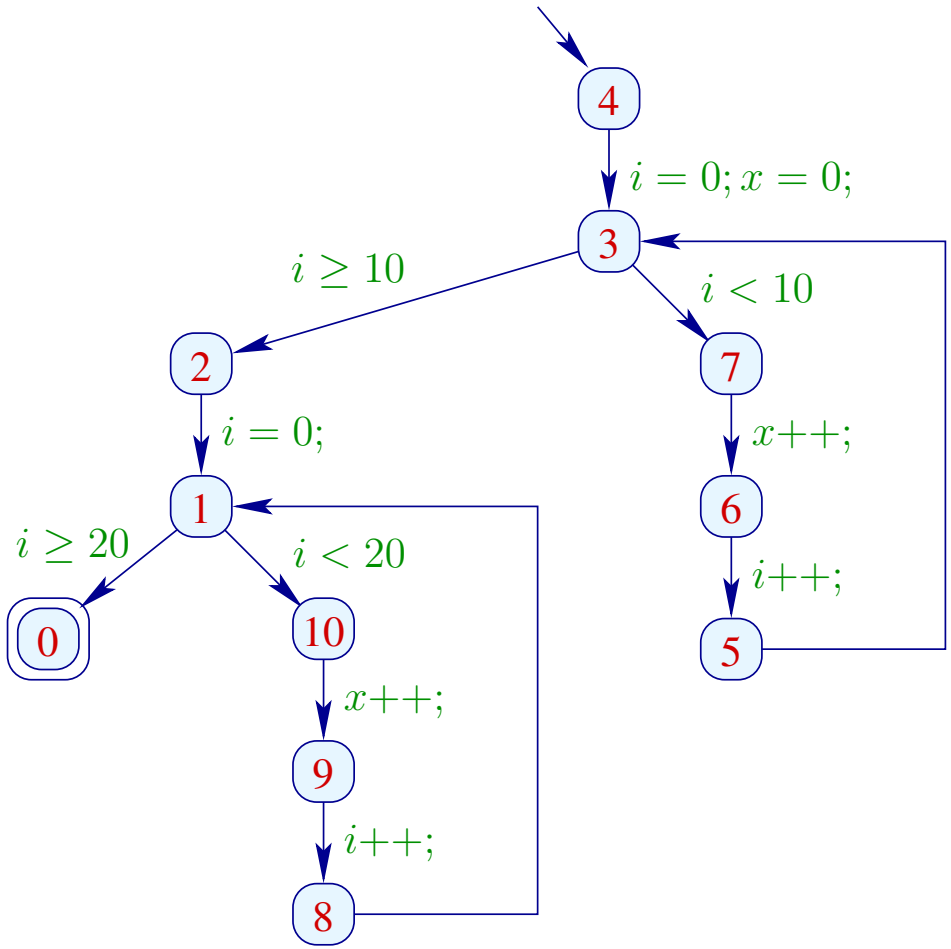
$\Rightarrow$ Priorities must be assigned on the fly.

$\Rightarrow$ Loop heads for placing W/N also must be detected on the fly.

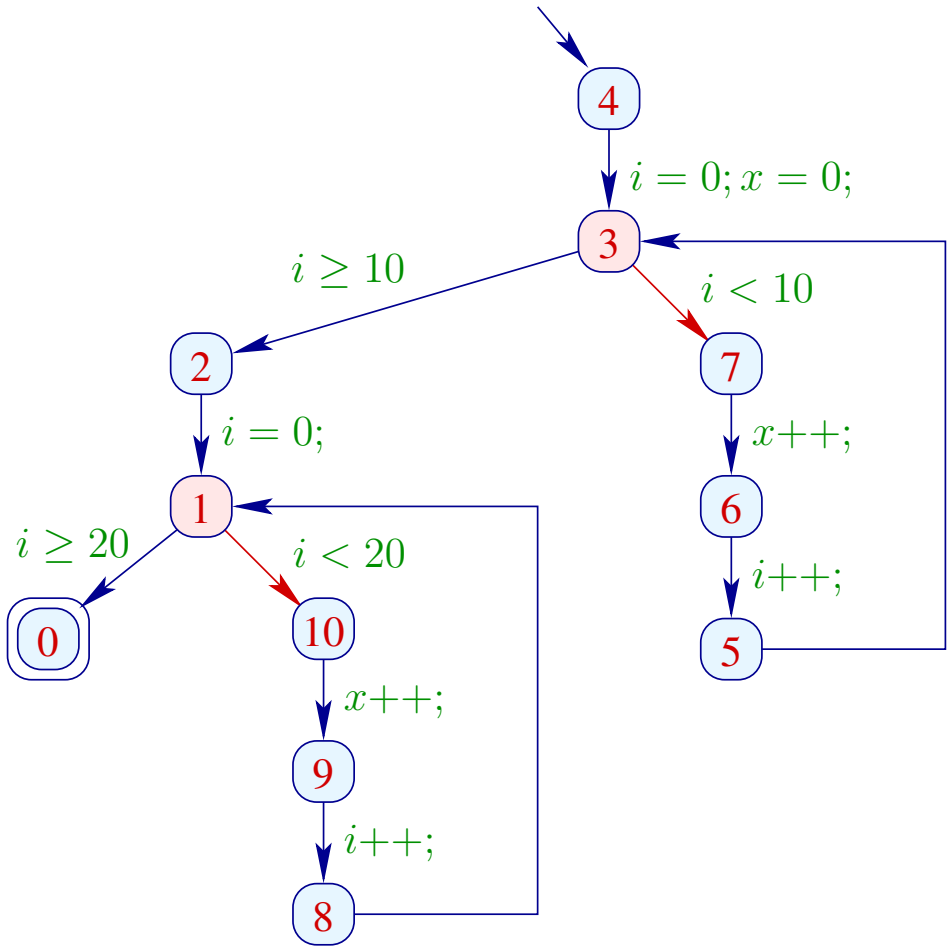
Idea

Exploit priorities ...

Assignment of priorities starts at end point of the program and then recursively descends into variable dependences ...



Assignment of priorities starts at end point of the program and then recursively descends into variable dependences ...



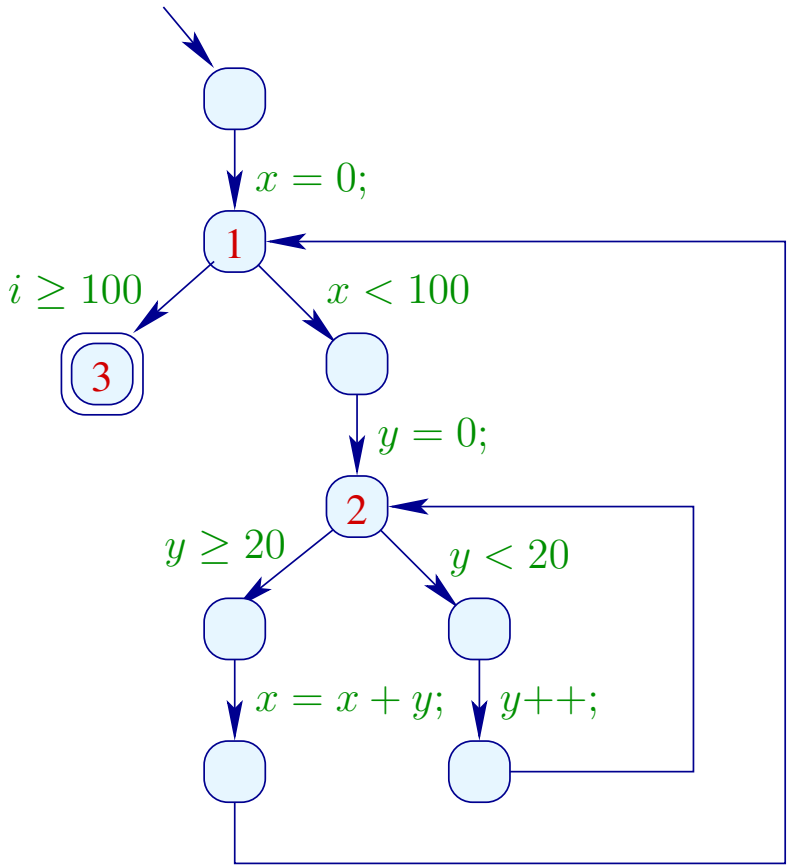
Theorem

(1) If SLR^+ terminates, a post-solution has been found.

(2) SLR^+ terminates, whenever all right-hand sides are monotonic and only finitely many unknowns are encountered.

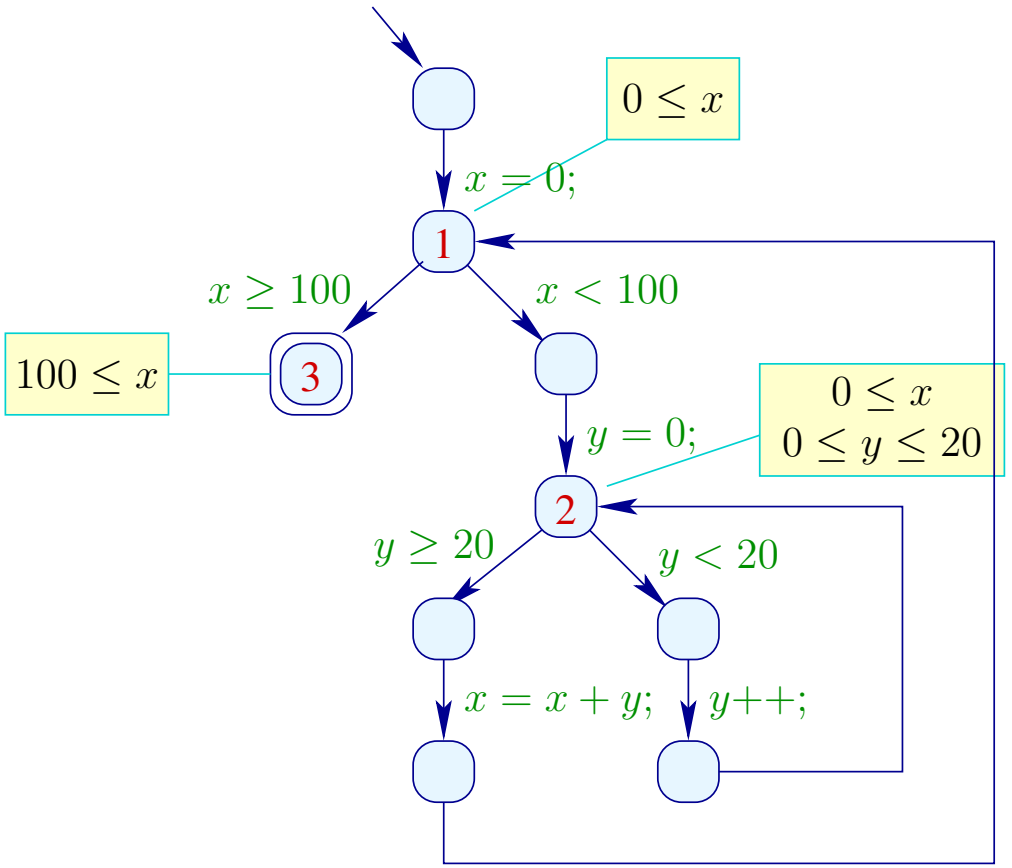
Back Edges

Combined W/N still may be doomed when loops are nested ...



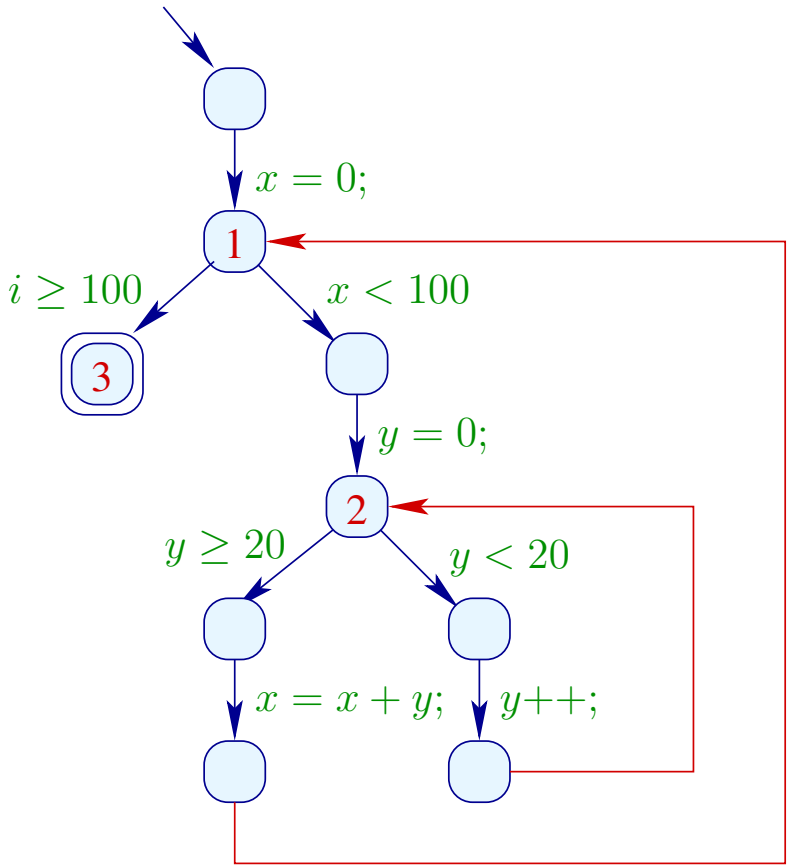
Back Edges

Combined W/N still may be doomed when loops are nested ...



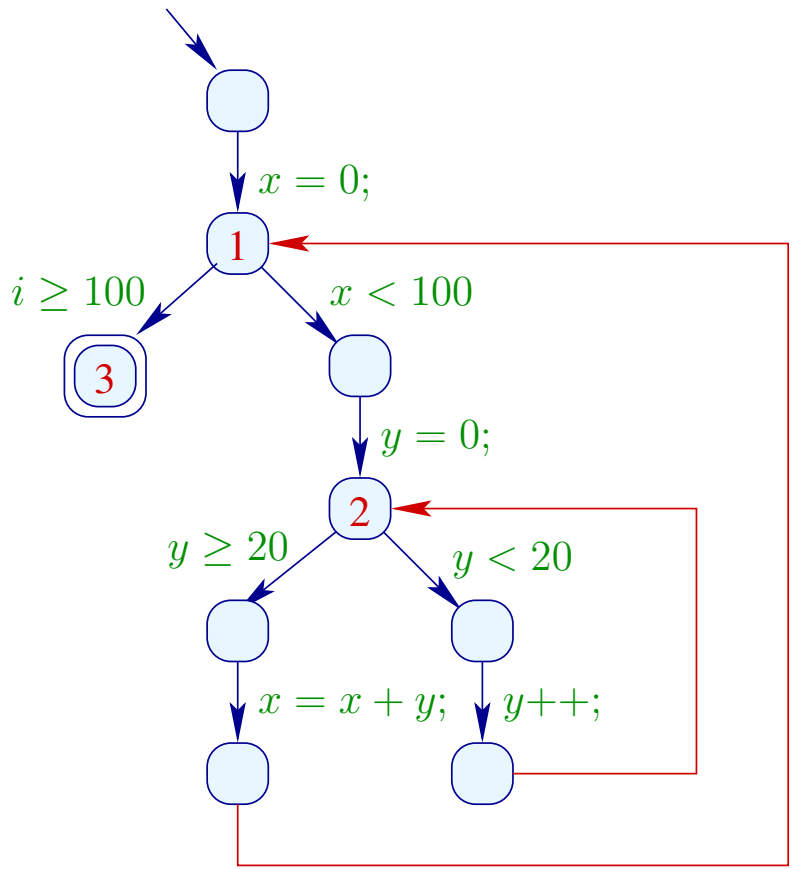
Restrict W/N to back edges: Amato, Scozzari 2013

$$z = z \boxplus (\mathsf{outside} \sqcup \mathsf{back})$$



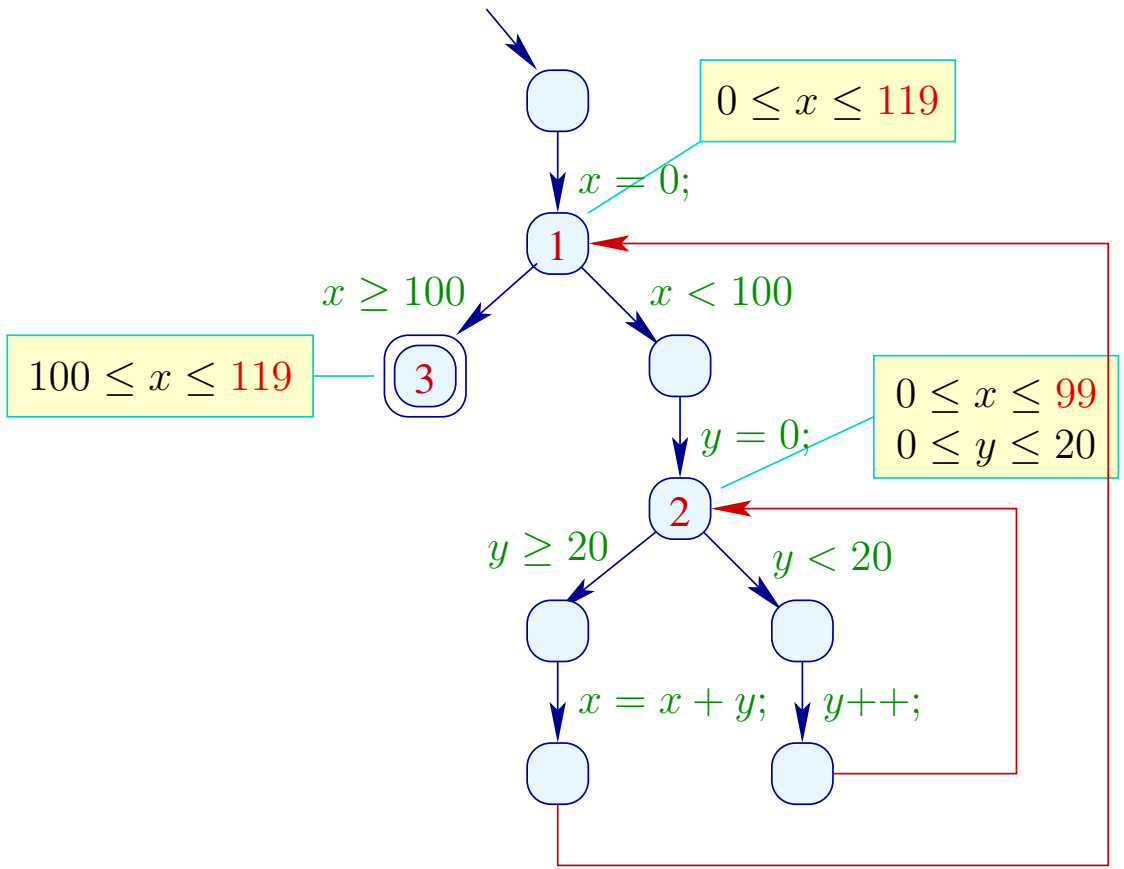
Restrict W/N to back edges: Amato, Scozzari 2013

$$z = \mathsf{outside} \sqcup (z \boxplus \mathsf{back})$$



Restrict W/N to back edges: Amato, Scozzari 2013

$$z = \mathsf{outside} \sqcup (z \boxplus \mathsf{back})$$



Discussion

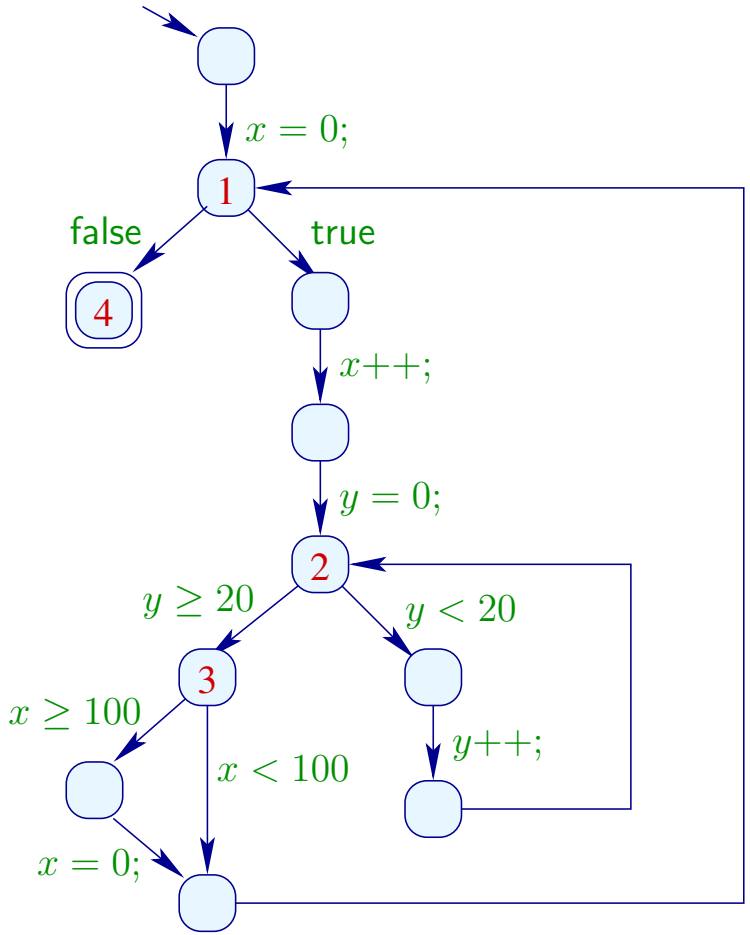
- Restricting $\exists$ to back edges, may increase precision significantly !
- It can be easily integrated into SRR, SW without compromising termination !!

Discussion

- Restricting $\boxminus$ to back edges, may increase precision significantly !
- It can be easily integrated into SRR, SW without compromising termination !!
- It is not yet clear how it can be integrated into local solvers such as SLR^+ where no preprocessing is possible.

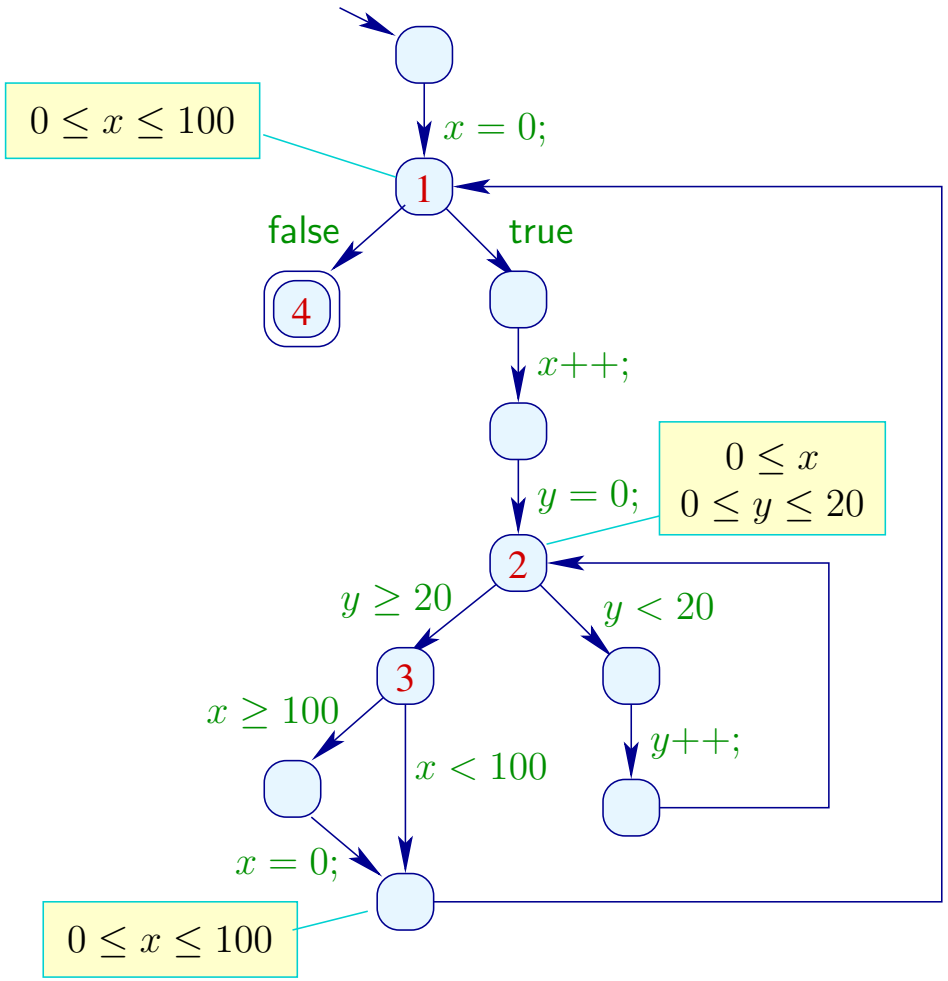
Restart

Information loss in inner loops can be quite insistent ...



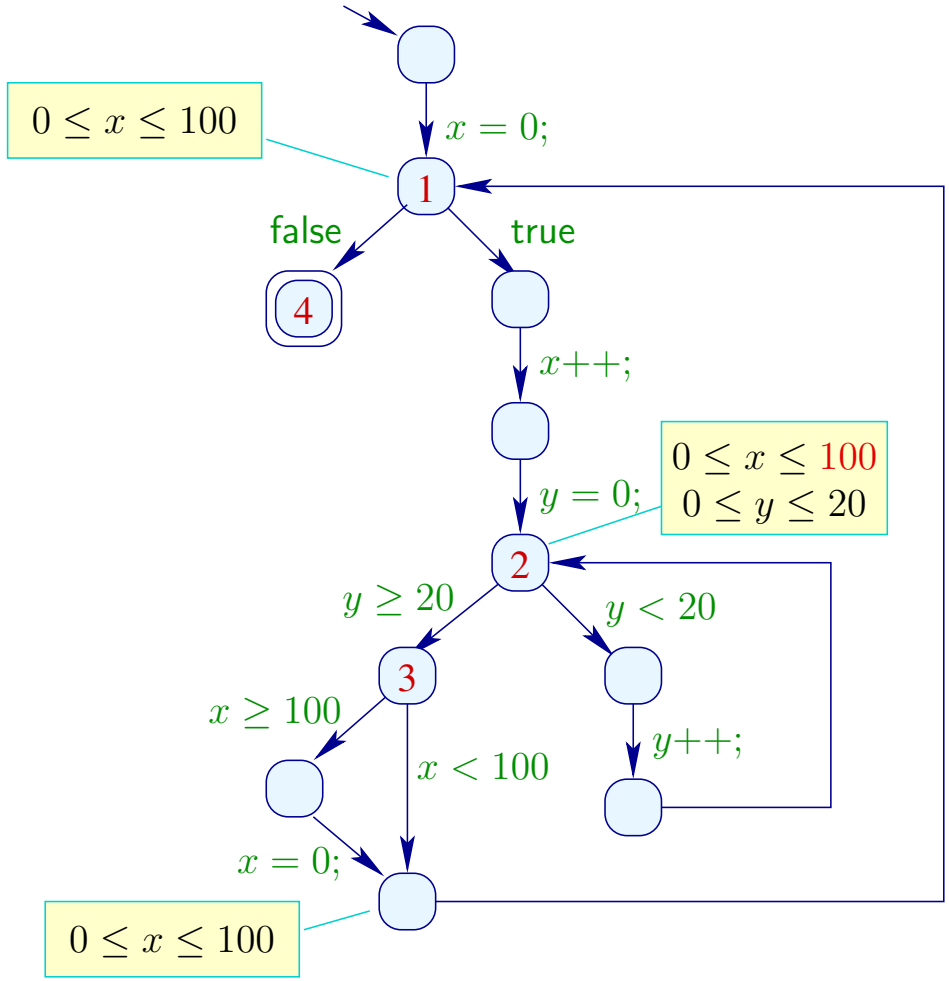
Restart

Information loss in inner loops can be quite insistent ...



Amato, Scozzari 2013

At decrease, restart iteration on lower-priority unknowns ...



Discussion

- After restarting of lower-priority unknowns, the value of x may be further decreased !

Discussion

- After restarting of lower-priority unknowns, the value of x may be further decreased !
- Restarting can be integrated into any structured iterator without compromising termination !!

Discussion

- After restarting of lower-priority unknowns, the value of x may be further decreased !
- Restarting can be integrated into any structured iterator without compromising termination !!
- In theory, it may drastically increase complexity.
- In practice, it seems to cause only a slowdown by a small factor.

Experiments

Precision 2-phase vs. 目-solving

Efficiency 目-solving against 凵-solving

Experiments

Precision 2-phase vs. ⊟-solving

benchmark suite of the Märdalen WCET research group

Efficiency ⊟-solving against ⊔-solving

C projects of the SpecCpu2006 benchmark suite that can be handled by CIL

Results

- A significant of amount of precision is retained by combining widening and narrowing !

Results

- A significant of amount of precision is retained by combining widening and narrowing !
- Back-edge aware iteration as well restarting may each provide further (small) improvements in precision.

Results

- A significant of amount of precision is retained by combining widening and narrowing !
- Back-edge aware iteration as well restarting may each provide further (small) improvements in precision.
- In absence of context, the $\boxplus$-solver is only marginally slower than the corresponding $\sqcup$-solver.
- Restarting is not completely impractical: in our experiments, the penalty was just a small factor !!

Summary

- A decent fixpoint engine should support lattices with infinite ascending chains.

  A generic idea which works for any lattice is W/N.

Summary

- A decent fixpoint engine should support lattices with infinite ascending chains.

  A generic idea which works for any lattice is W/N.

- The ⊟-operator can be plugged into any reasonable solver — at the price of potential non-termination.

Summary

- A decent fixpoint engine should support lattices with infinite ascending chains.

  A generic idea which works for any lattice is W/N.

- The ⊟-operator can be plugged into any reasonable solver — at the price of potential non-termination.

- Termination can be enforced for monotonic systems when finitely many unknowns are encountered—by modifying the fixpoint algorithms appropiately.

- These seem to terminate practically — also for our nonmonotonic systems !

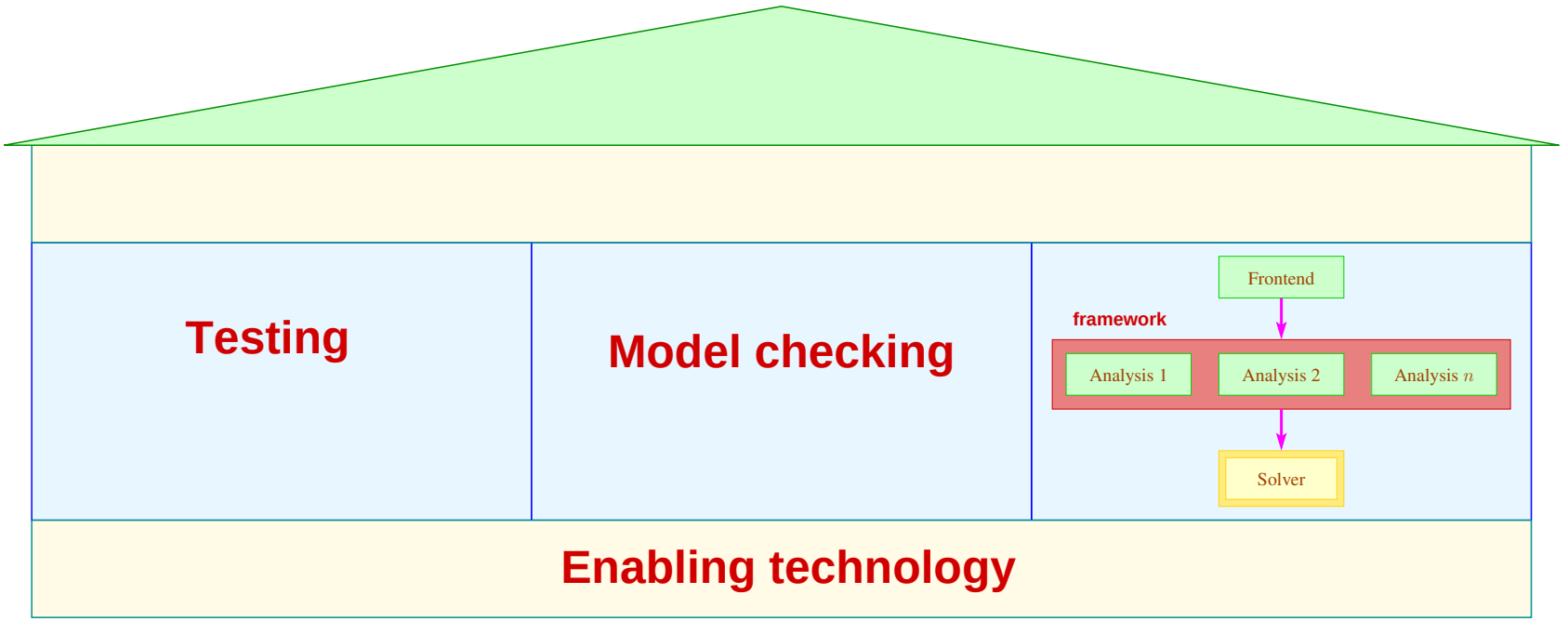
Summary (cont.)

- In order to achieve maximal precision, the solver can
  - apply w/n reluctantly
  - e.g., only at back edges
  - restart solving when values start decreasing

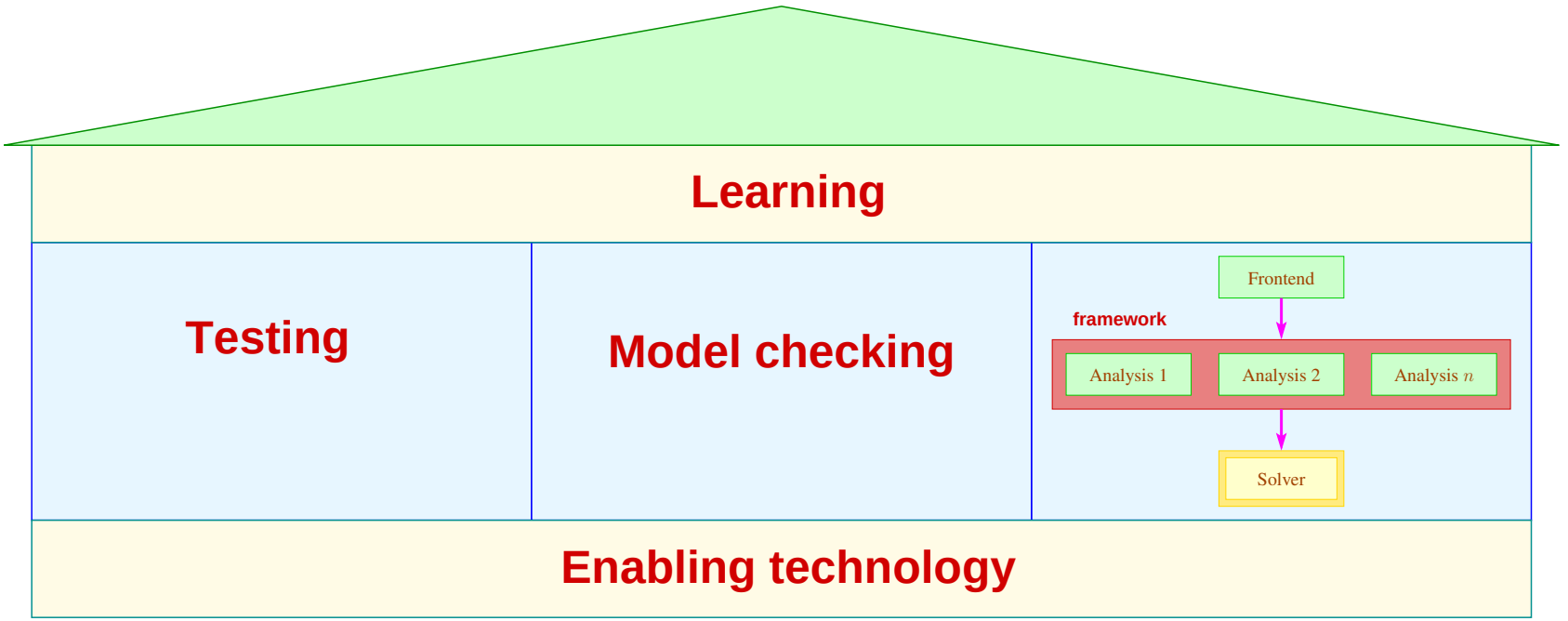
Summary (cont.)

- In order to achieve maximal precision, the solver can
  - apply w/n reluctantly
  - e.g., only at back edges
  - restart solving when values start decreasing
- The extra precision may come at an extra cost.
- ... which is surprisingly small !

Dream


Testing
Model checking
framework
Frontend
Analysis 1
Analysis 2
Analysis n
Solver
Enabling technology

Dream


Learning
Testing
Model checking
framework
Frontend
Analysis 1
Analysis 2
Analysis n
Solver
Enabling technology

Acknowledgement

This work has been supported by the EU project MBAT

Model-Based Analysis and Testing

(ARTEMIS Joint Undertaking under grant agreement 269335)