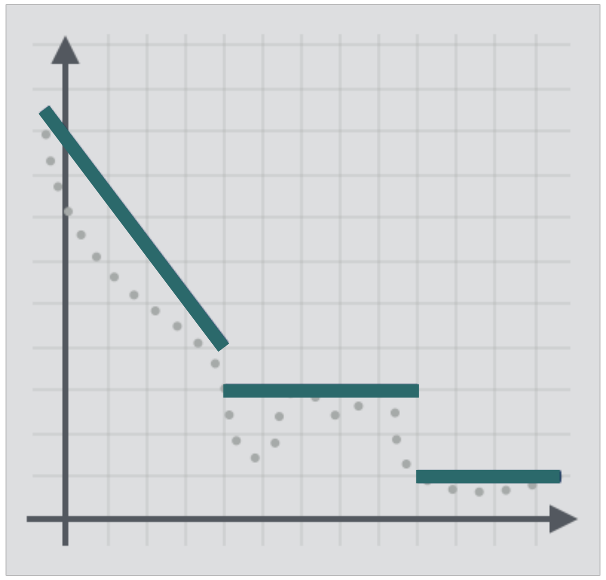
Liveness Analysis

MPRI 2-6: Abstract Interpretation, Application to Verification and Static Analysis



Caterina Urban

October 31st, 2022

Year 2022-2023

Liveness Properties

- **Guarantee Properties**
  "something good eventually happens at least once"
  - Example: Program Termination
- **Recurrence Properties**
  "something good eventually happens infinitely often"
  - Example: Starvation Freedom

Guarantee Properties

Guarantee Properties

"something good eventually happens at least once"

AF φ

$\varphi ::= e \bowtie 0 \mid \ell : e \bowtie 0 \mid \varphi \wedge \varphi \mid \varphi \vee \varphi$ $\qquad \ell \in \mathscr{L}$

Example:

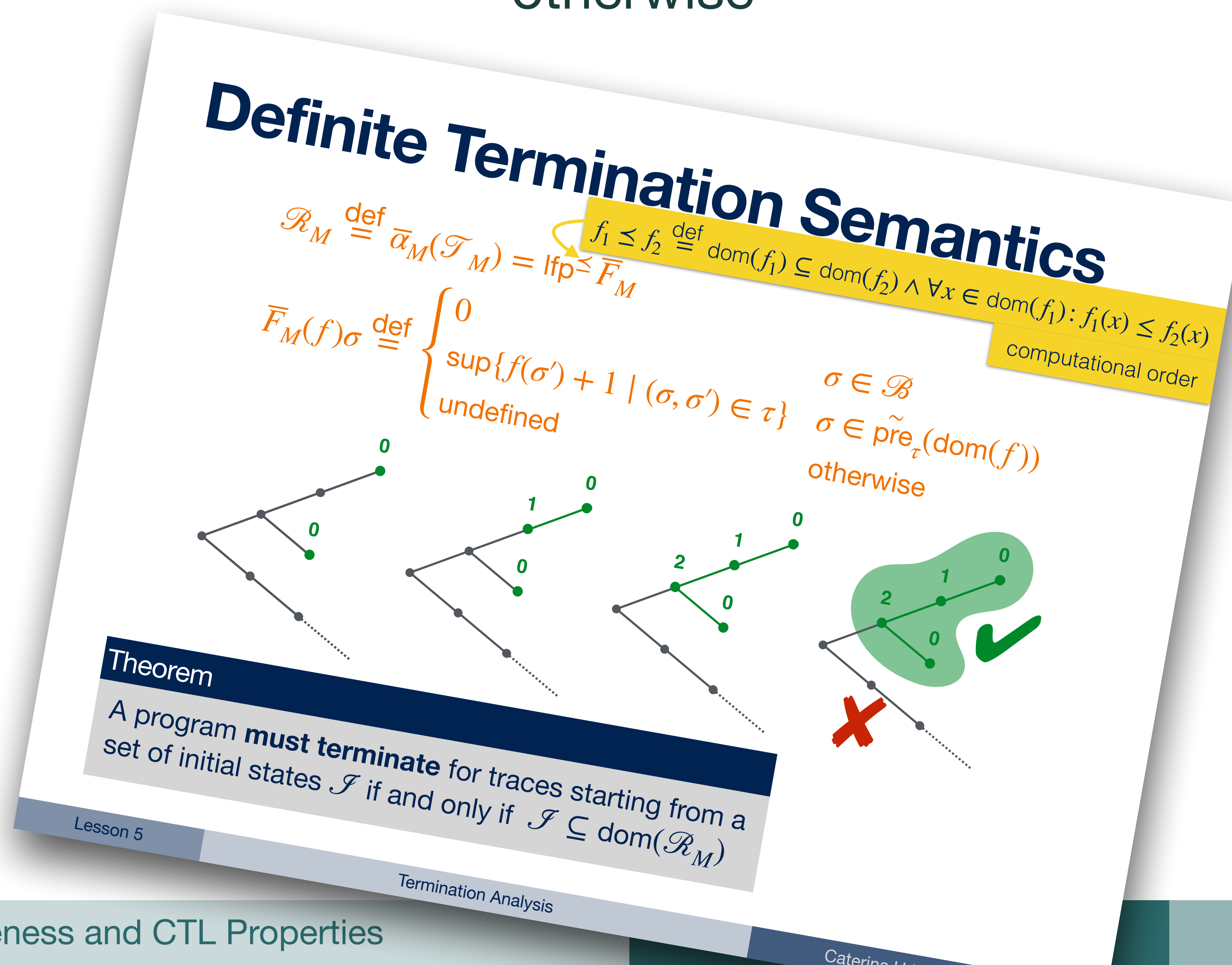
```
1x ← [-∞, +∞]
 while 2(x ≥ 0) do
     3x ← x + 1
od4
 while 5(0 ≥ 0) do
     if 6(x ≤ 10) do
          7x ← x + 1
     else
          8x ← -x
od9
```

AF $(x = 3)$ is satisfied for $\mathscr{I} \stackrel{\text{def}}{=} \{(1,\rho) \in \Sigma \mid \rho(x) \leq 3\}$

Guarantee Semantics

$$\mathcal{R}_G^{\varphi} \overset{\text{def}}{=} \text{lfp}^{\preceq}\, \overline{F}_G[\{\sigma \in \Sigma \mid \sigma \vDash \varphi\}]$$

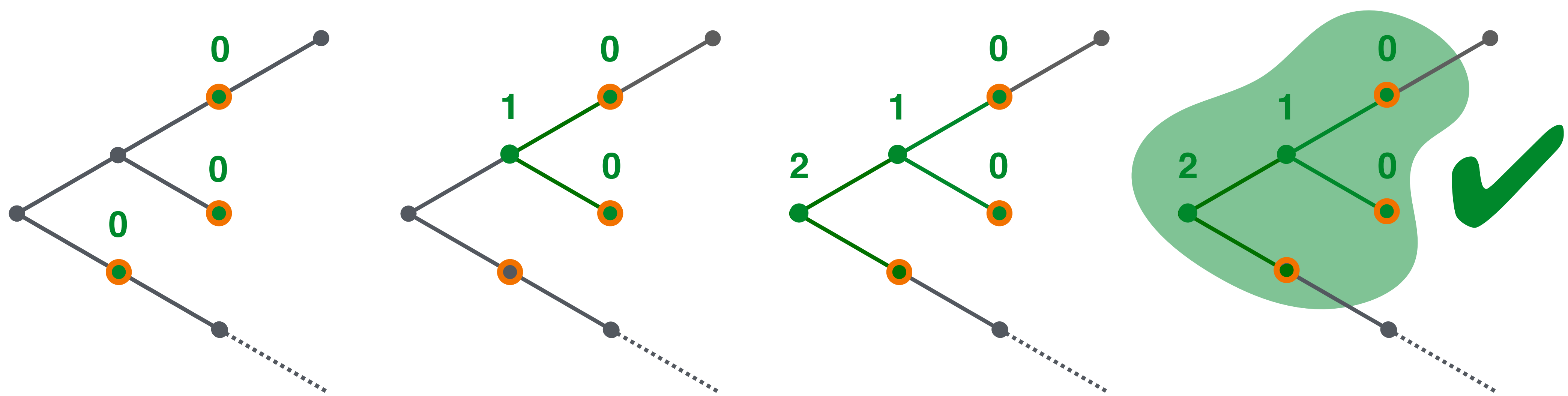
$$\overline{F}_G[S]f \overset{\text{def}}{=} \lambda\sigma.\begin{cases} 0 & \sigma \in S \\ \sup\{f(\sigma') + 1 \mid (\sigma, \sigma') \in \tau\} & \sigma \notin S \wedge \sigma \in \widetilde{\text{pre}}_{\tau}(\text{dom}(f)) \\ \text{undefined} & \text{otherwise} \end{cases}$$



Guarantee Semantics

$$\mathcal{R}_G^{\varphi} \overset{\text{def}}{=} \mathsf{lfp}^{\preceq}\, \overline{F}_G[\{\sigma \in \Sigma \mid \sigma \models \varphi\}]$$

$$\overline{F}_G[S]f \overset{\text{def}}{=} \lambda\sigma\,.\begin{cases} 0 & \sigma \in S \\ \sup\{f(\sigma') + 1 \mid (\sigma, \sigma') \in \tau\} & \sigma \notin S \wedge \sigma \in \widetilde{\mathsf{pre}}_\tau(\mathsf{dom}(f)) \\ \text{undefined} & \text{otherwise} \end{cases}$$



Theorem

A program satisfies a **guarantee property** $\mathsf{AF}\,\varphi$ for traces starting from a set of initial states $\mathcal{I}$ if and only if $\mathcal{I} \subseteq \mathsf{dom}(\mathcal{R}_G^{\varphi})$

Abstract Guarantee Semantics

For each program instruction stat, we define $\mathcal{R}_G^{\varphi\#}[\![\text{stat}]\!] : \mathcal{A} \to \mathcal{A}$:

- $\mathcal{R}_G^{\varphi\#}[\![^{\ell} X \leftarrow e]\!]t \stackrel{\text{def}}{=} \text{RESET}_A^G[\![\varphi]\!](\overleftarrow{\text{ASSIGN}}_A[\![X \leftarrow e]\!]t)$

- $\mathcal{R}_G^{\varphi\#}[\![\textbf{if } ^{\ell} e \bowtie 0 \textbf{ then } s]\!]t \stackrel{\text{def}}{=} \text{RESET}_A^G[\![\varphi]\!](X)$
  where $X \stackrel{\text{def}}{=} \text{FILTER}_A[\![e \bowtie 0]\!](\mathcal{R}_G^{\varphi\#}[\![s]\!]t) \curlyvee_T \text{FILTER}_A[\![e \not\bowtie 0]\!]t)$

- $\mathcal{R}_G^{\varphi\#}[\![\textbf{while } ^{\ell} e \bowtie 0 \textbf{ do } s \textbf{ done}]\!]t \stackrel{\text{def}}{=} \text{lfp}^\# \overline{F}_G^{\varphi\#}$
  where $\overline{F}_G^{\varphi\#}(x) \stackrel{\text{def}}{=} \text{RESET}_A^G[\![\varphi]\!](X)$
  $X \stackrel{\text{def}}{=} \text{FILTER}_A[\![e \bowtie 0]\!](\mathcal{R}_G^{\varphi\#}[\![s]\!]x) \curlyvee_T \text{FILTER}_A[\![e \not\bowtie 0]\!](t))$

- $\mathcal{R}_G^{\varphi\#}[\![s_1; s_2]\!]t \stackrel{\text{def}}{=} \mathcal{R}_G^{\varphi\#}[\![s_1]\!](\mathcal{R}_G^{\varphi\#}[\![s_2]\!]t)$

Abstract Guarantee Semantics

Example

$\text{int} : x,\ \ y$

while $^{1}(x \geq 0)$ do

  $^{2}x := x + 1$

od

while $^{3}($ true $)$ do

  if $^{4}(\ x \leq 10\ )$

    $^{5}x := x + 1$

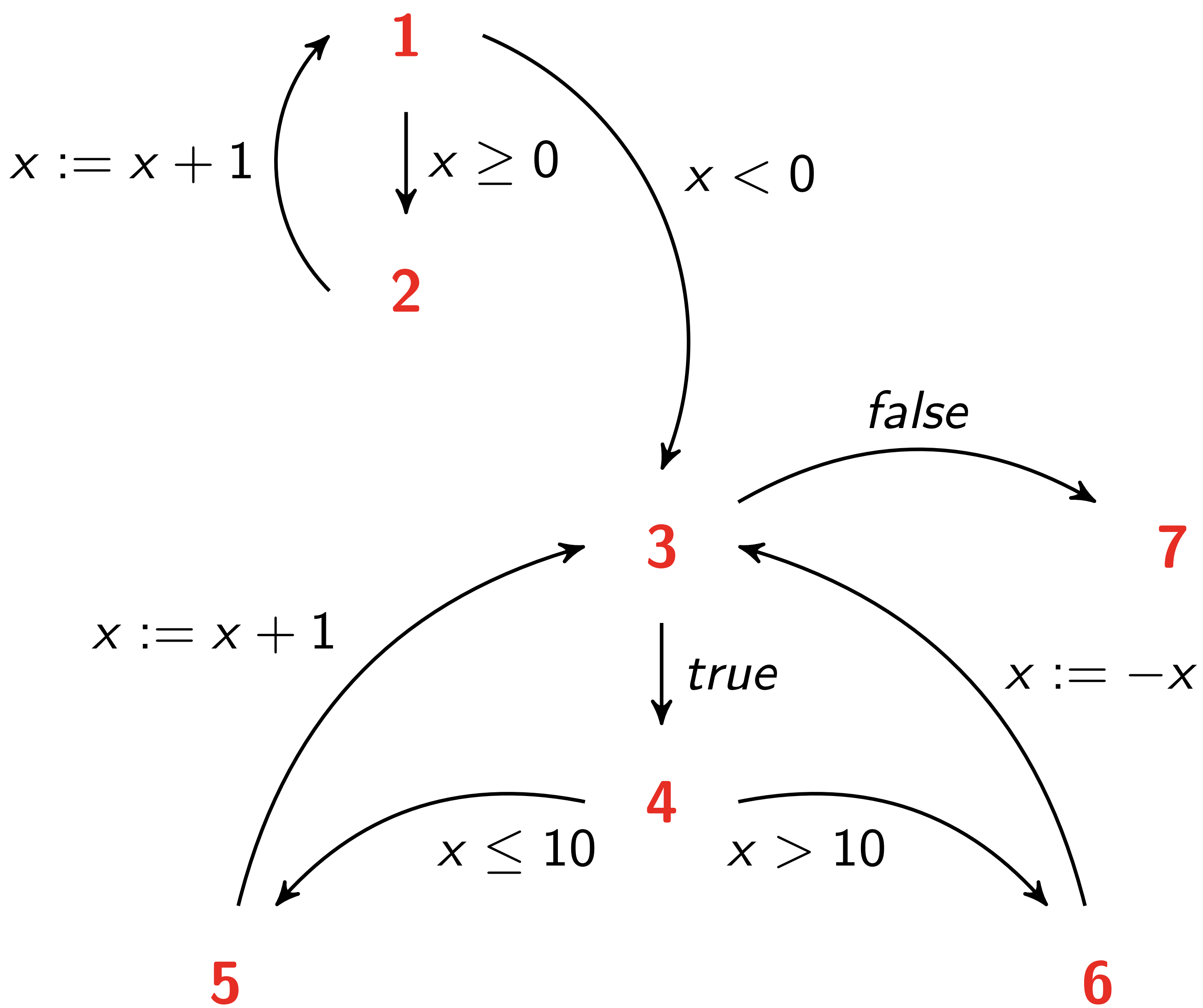
  else

    $^{6}x := -x$

od^{7}

Property

$\mathsf{AF}\,(x = 3)$



Abstract Guarantee Semantics

Example

int : x, y

while $^{1}(x \geq 0)$ do

$^{2}x := x + 1$

od

while $^{3}($ true $)$ do

if $^{4}(\ x \leq 10\ )$

$^{5}x := x + 1$

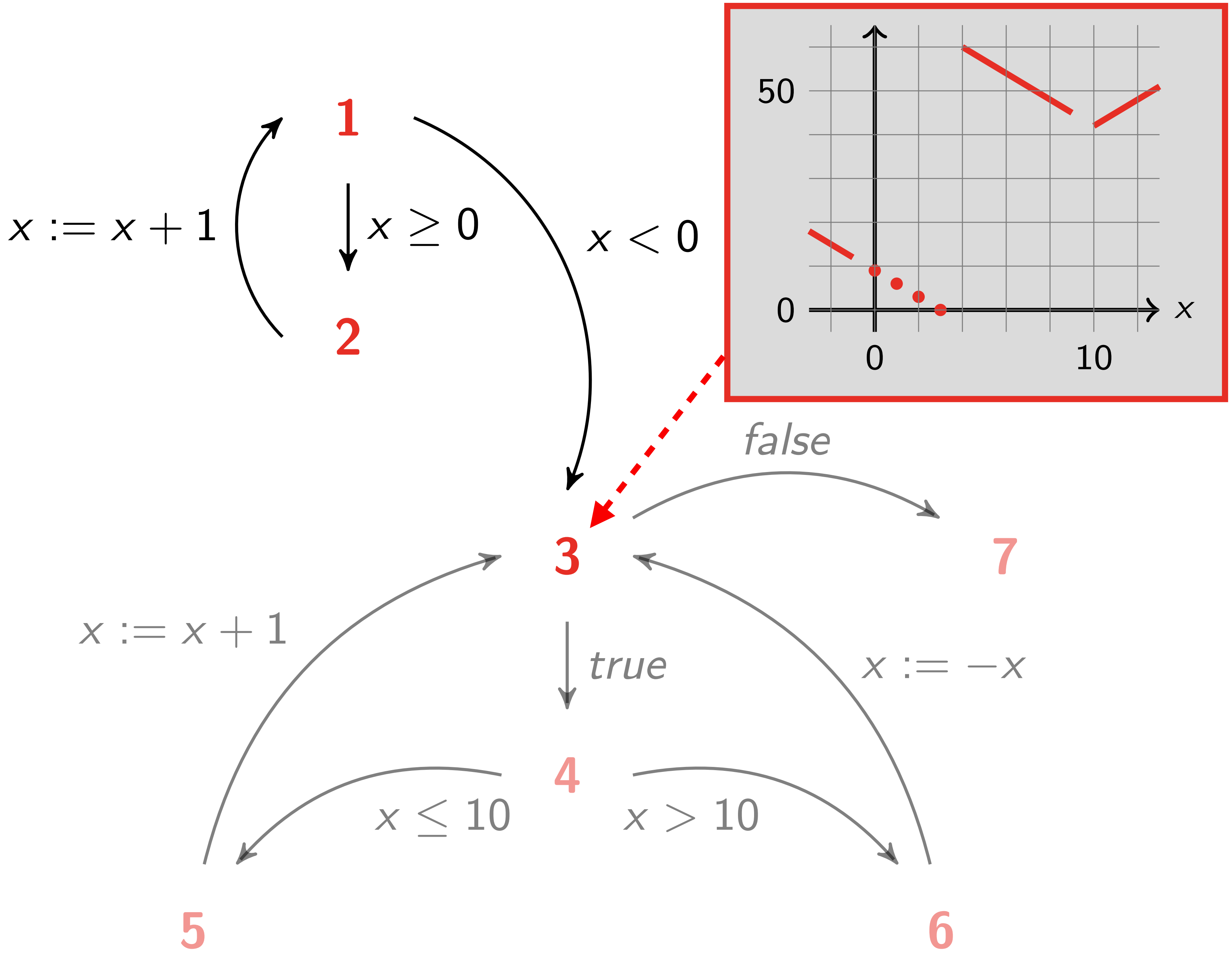
else

$^{6}x := -x$

od^{7}

Property

$$\text{AF}\,(x = 3)$$



Abstract Guarantee Semantics

Example

int : x, y

while $^{1}(x \geq 0)$

$^{2}x := x + 1$

od

while $^{3}($ true $)$ do

if $^{4}(\ x \leq 10\ )$

$^{5}x := x + 1$

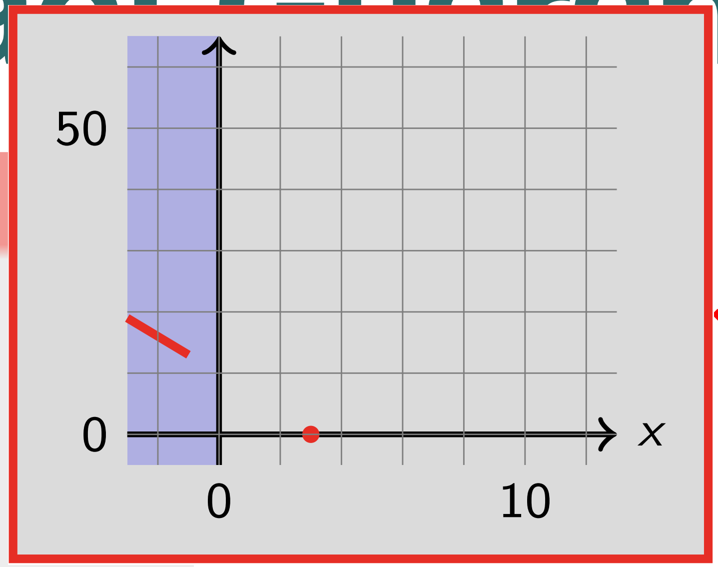
else

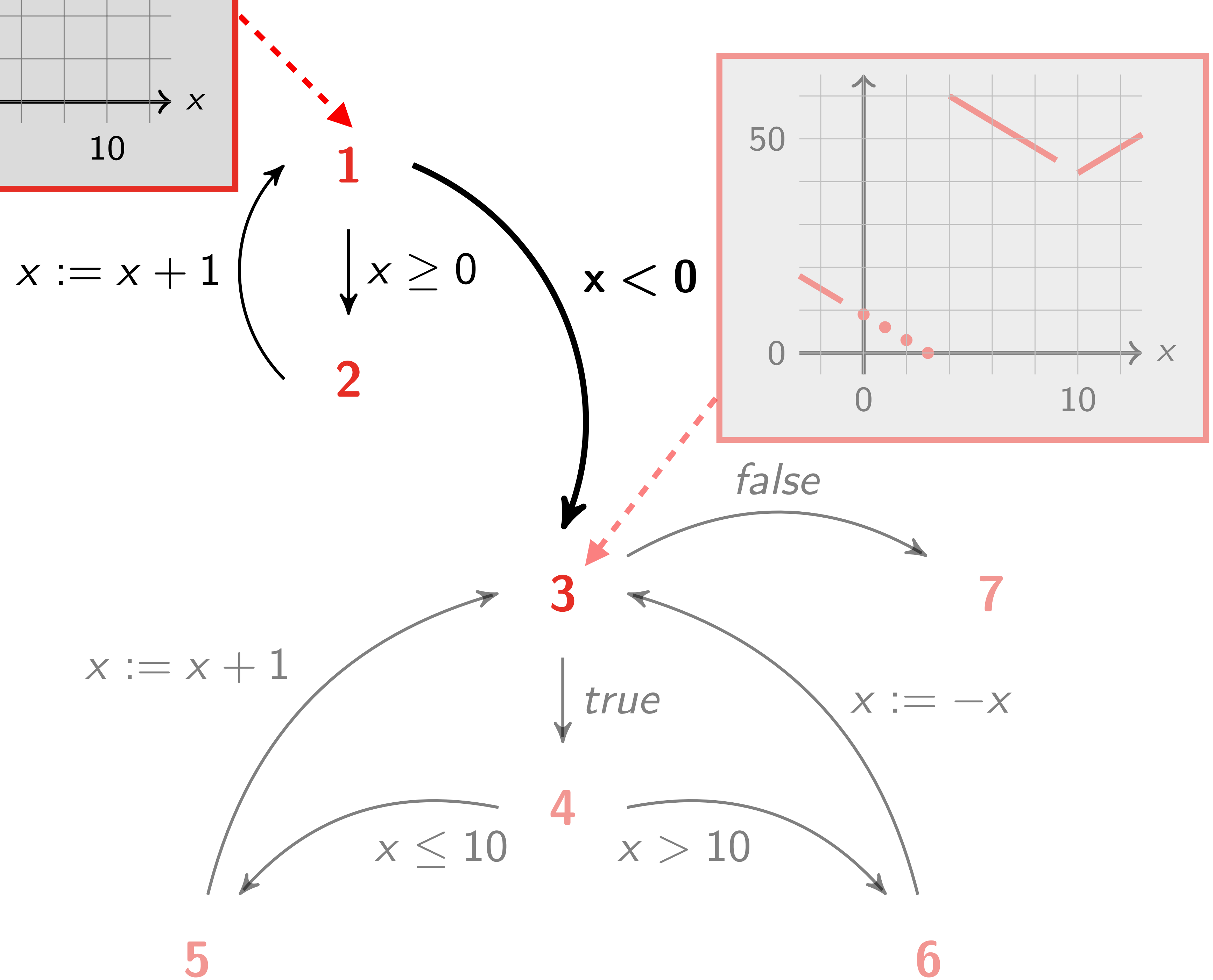
$^{6}x := -x$

od^{7}

Property

$$AF\,(x = 3)$$





Abstract Guarantee Semantics

Example

int : x, y

while $^{1}(x \geq 0)$

$\quad ^{2}x := x + 1$

od

while ^{3}(true) do

$\quad$ if $^{4}(\, x \leq 10 \,)$

$\quad\quad ^{5}x := x + 1$

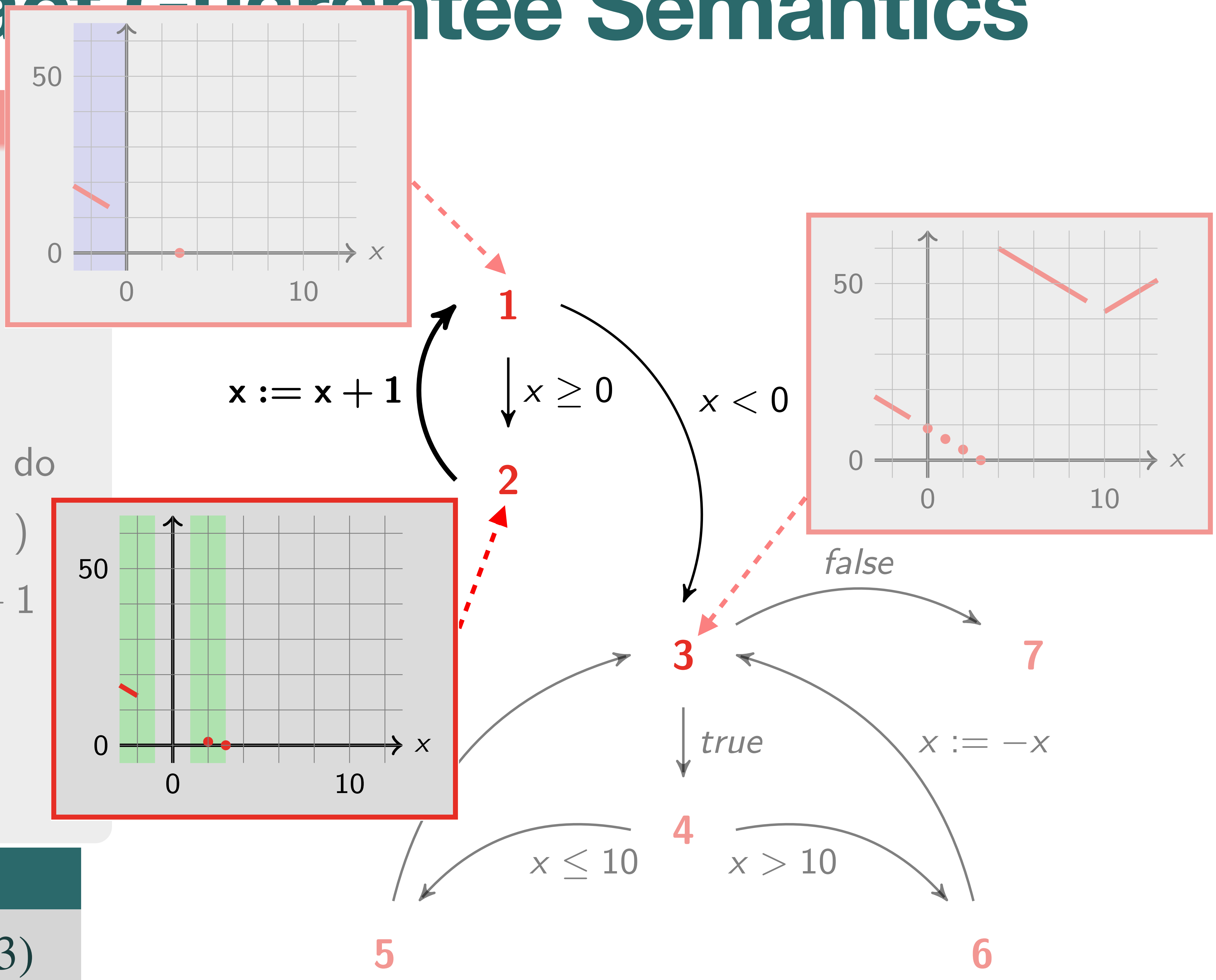
$\quad$ else

$\quad\quad ^{6}x := -x$

od^{7}

Property

$$\text{AF}\,(x = 3)$$



Abstract Guarantee Semantics

Example

int : x, y

while $^1(x \geq 0)$

  $^2x := x + 1$

od

while $^3($ true $)$ do

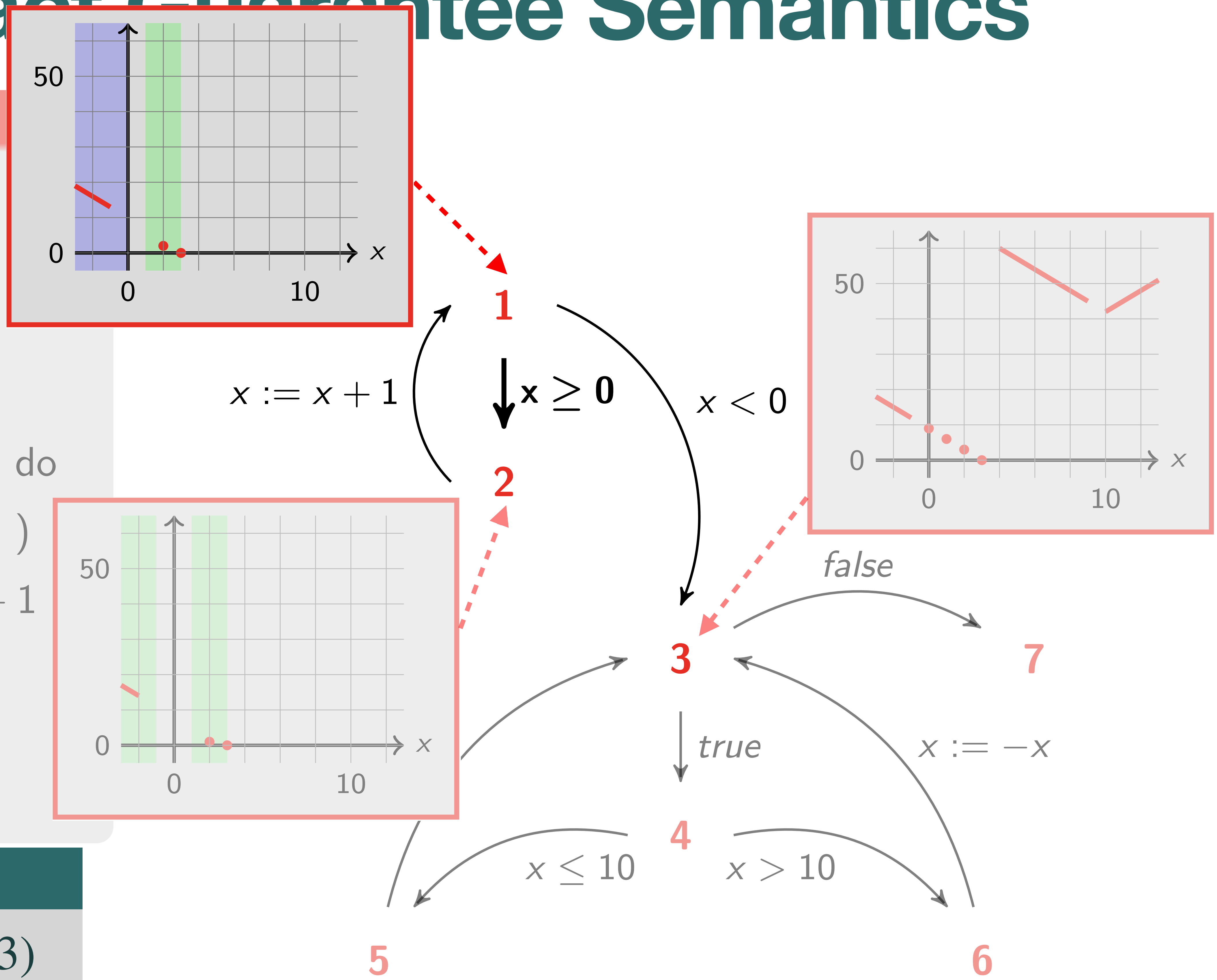
  if $^4(\ x \leq 10\ )$

    $^5x := x + 1$

  else

    $^6x := -x$

od^7



Property

$\mathsf{AF}\,(x = 3)$

Abstract Guarantee Semantics

Example

int : x, y

while ${}^{1}(x \geq 0)$

  ${}^{2}x := x + 1$

od

while ^{3}(true) do

  if ^{4}($x \leq 10$)

    ${}^{5}x := x + 1$

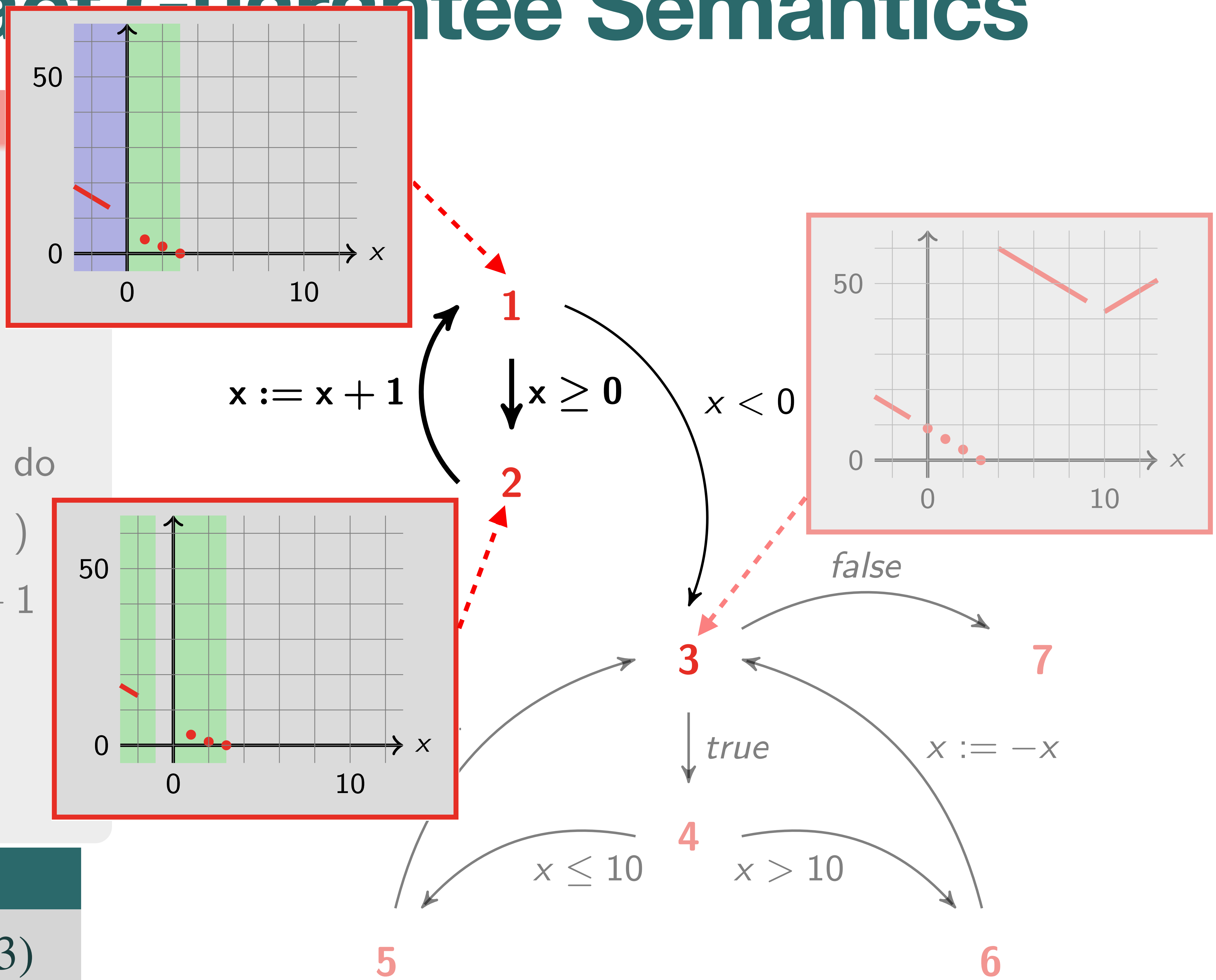
  else

    ${}^{6}x := -x$

od^{7}

Property

$\text{AF}\,(x = 3)$



Abstract Guarantee Semantics

Example

int : x, y

while $^1(x \geq 0)$

$\quad ^2x := x + 1$

od

while $^3($ true $)$ do

$\quad$ if $^4(x \leq 10)$

$\quad\quad ^5x := x + 1$

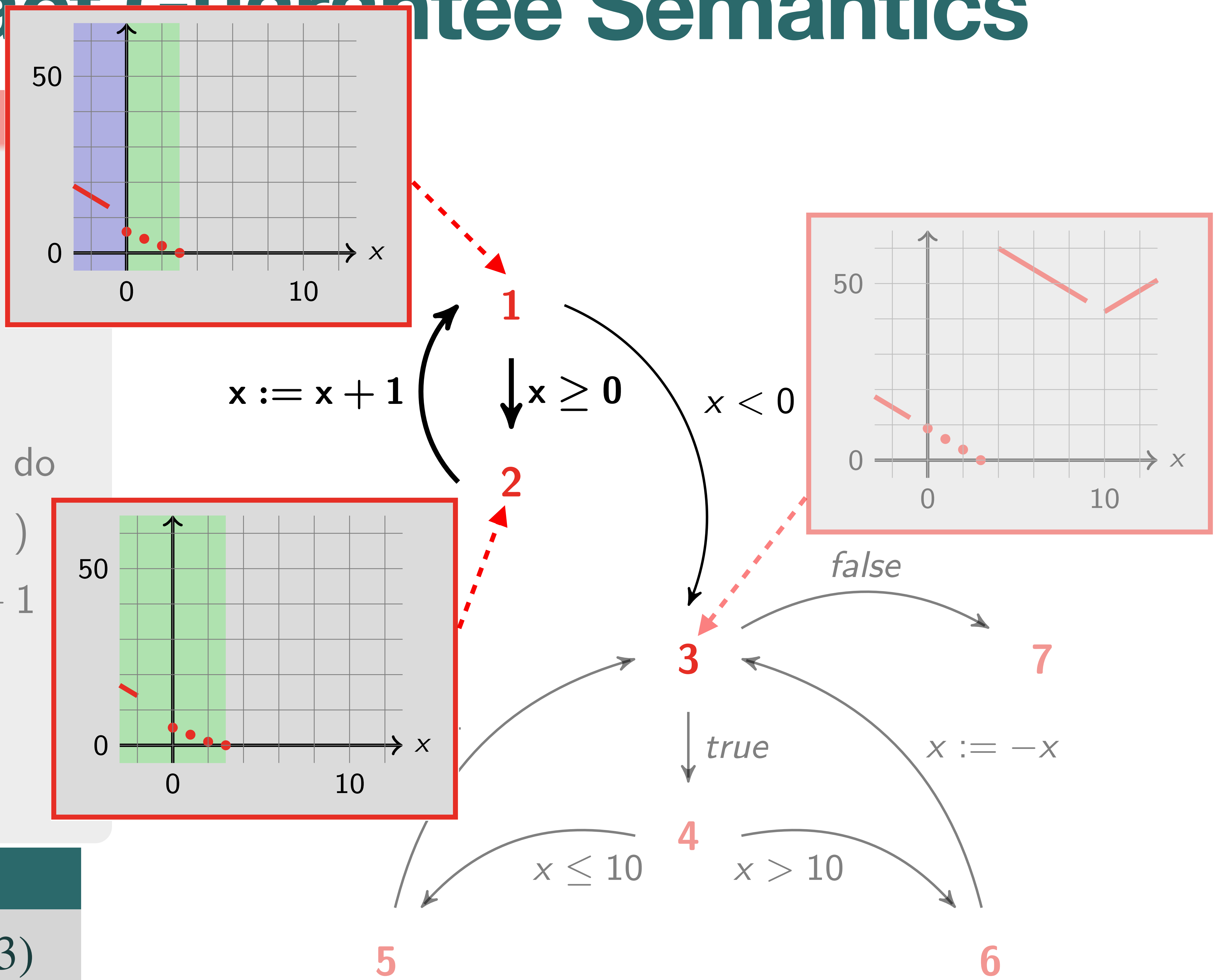
$\quad$ else

$\quad\quad ^6x := -x$

od^7

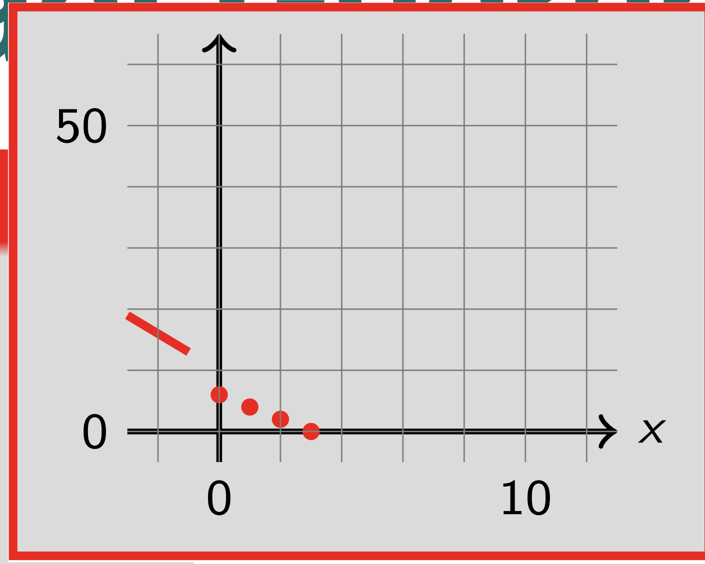
Property

$$\text{AF}\,(x = 3)$$



Abstract Guarantee Semantics

the analysis gives $x \leq 3$ as **sufficient precondition**



Example

int : x, y

while [1]$(x \geq 0)$

  [2]$x := x + 1$

od

while [3](true) do

  if [4]($x \leq 10$)

    [5]$x := x + 1$

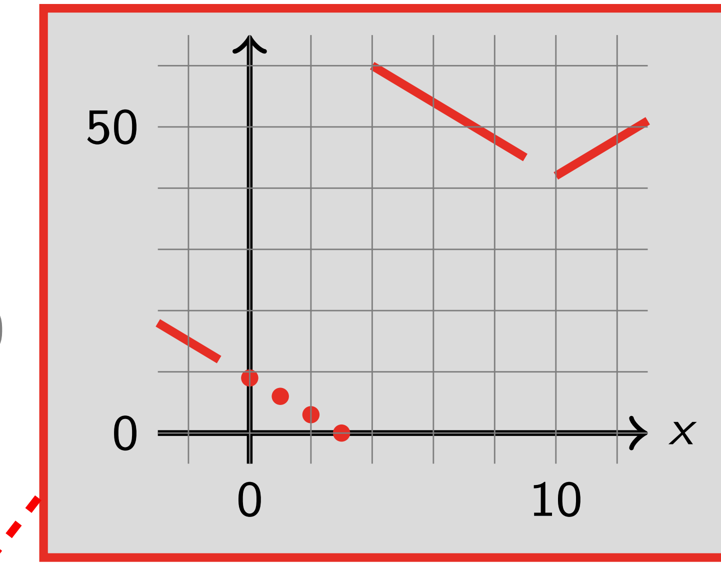
  else

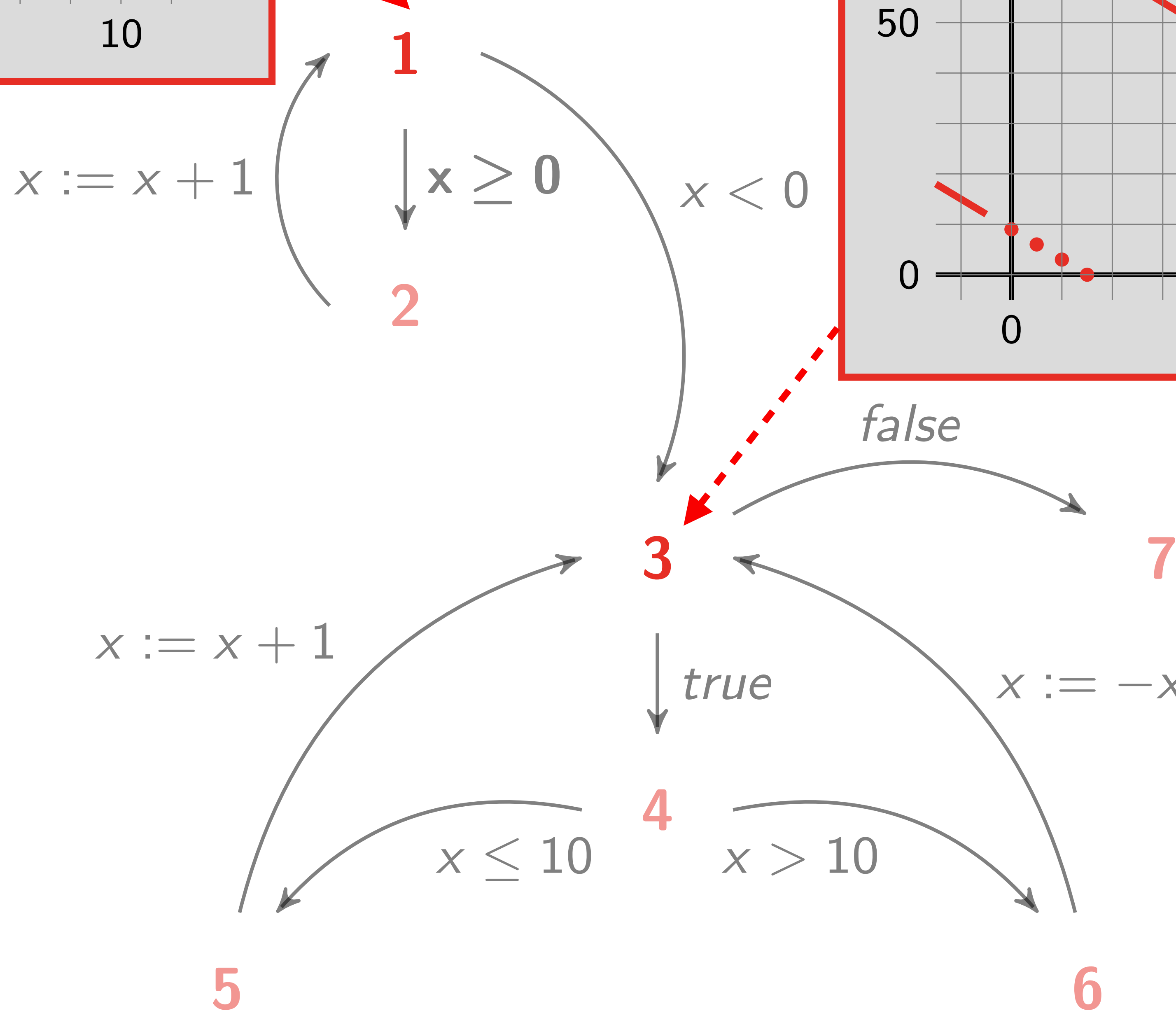
    [6]$x := -x$

od[7]

Property

AF $(x = 3)$





Abstract Guarantee Semantics

Definition

The **abstract guaranteee semantics** $\mathcal{R}_G^{\varphi\#}[\![\text{stat}^\ell]\!] \in \mathcal{A}$ of a program stat^ℓ is:

$$\mathcal{R}_G^{\varphi\#}[\![\text{stat}^\ell]\!] \stackrel{\text{def}}{=} \mathcal{R}_G^{\varphi\#}[\![\text{stat}]\!](\text{RESET}_A^G[\![\varphi]\!](\text{LEAF}: \bot_F))$$

where $\mathcal{R}_G^{\varphi\#}[\![\text{stat}]\!] : \mathcal{A} \to \mathcal{A}$ is the abstract guarantee semantics of each program instruction stat

Corollary (Soundness)

A program stat^ℓ satisfies a **guarantee property** AF φ for traces starting from a set of initial states $\mathcal{I}$ if $\mathcal{I} \subseteq \text{dom}(\gamma_A(\mathcal{R}_G^{\varphi\#}[\![\text{stat}^\ell]\!]))$

Recurrence Properties

Recurrence Properties

"something good eventually happens infinitely often"

$$\mathsf{AG\,AF}\,\varphi$$

$\varphi ::= e \bowtie 0 \mid \ell : e \bowtie 0 \mid \varphi \wedge \varphi \mid \varphi \vee \varphi$ $\qquad \ell \in \mathcal{L}$

Example:

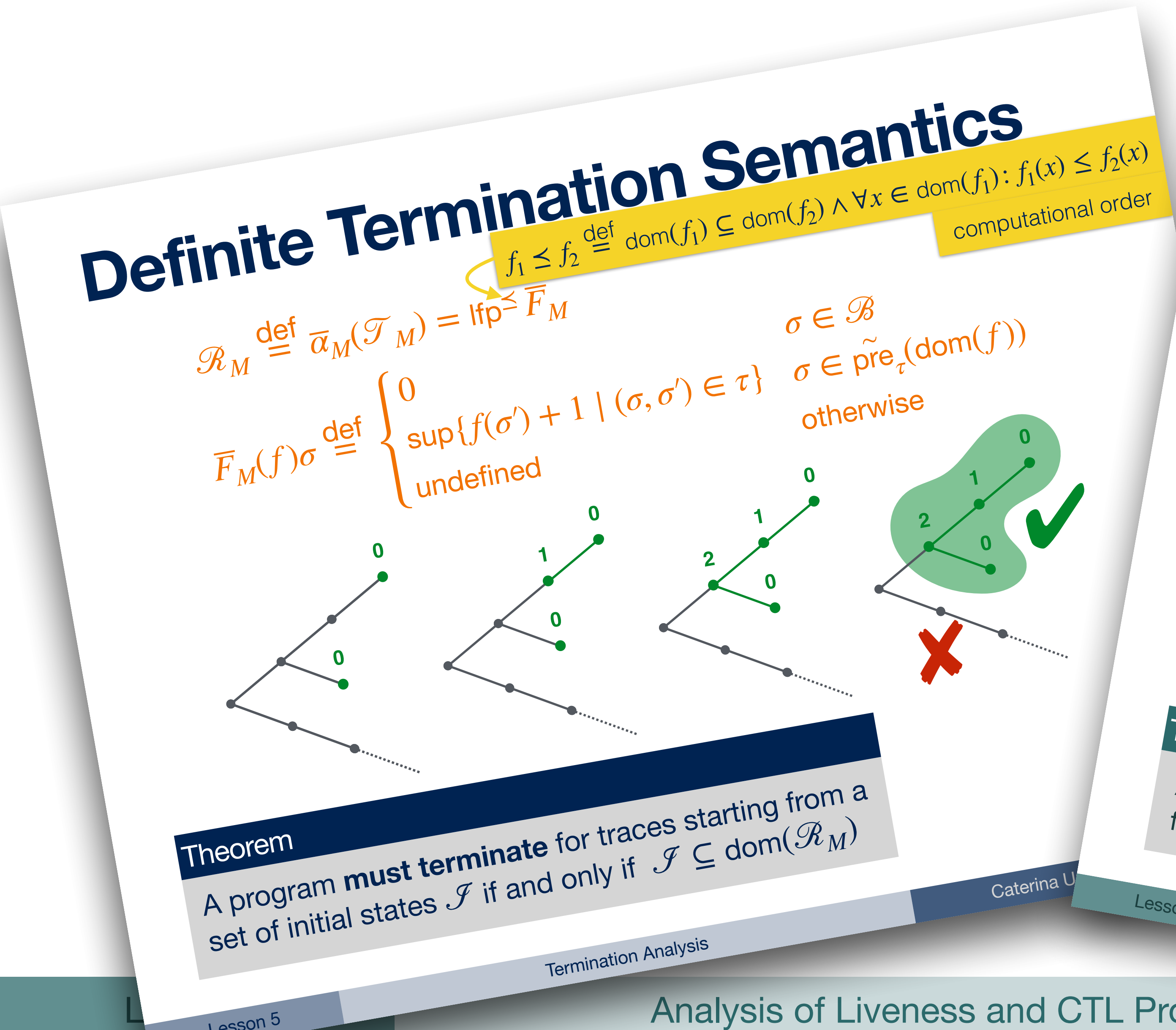
AG AF $(x = 3)$ is satisfied for $\mathcal{I} \overset{\text{def}}{=} \{(1,\rho) \in \Sigma \mid \rho(x) < 0\}$

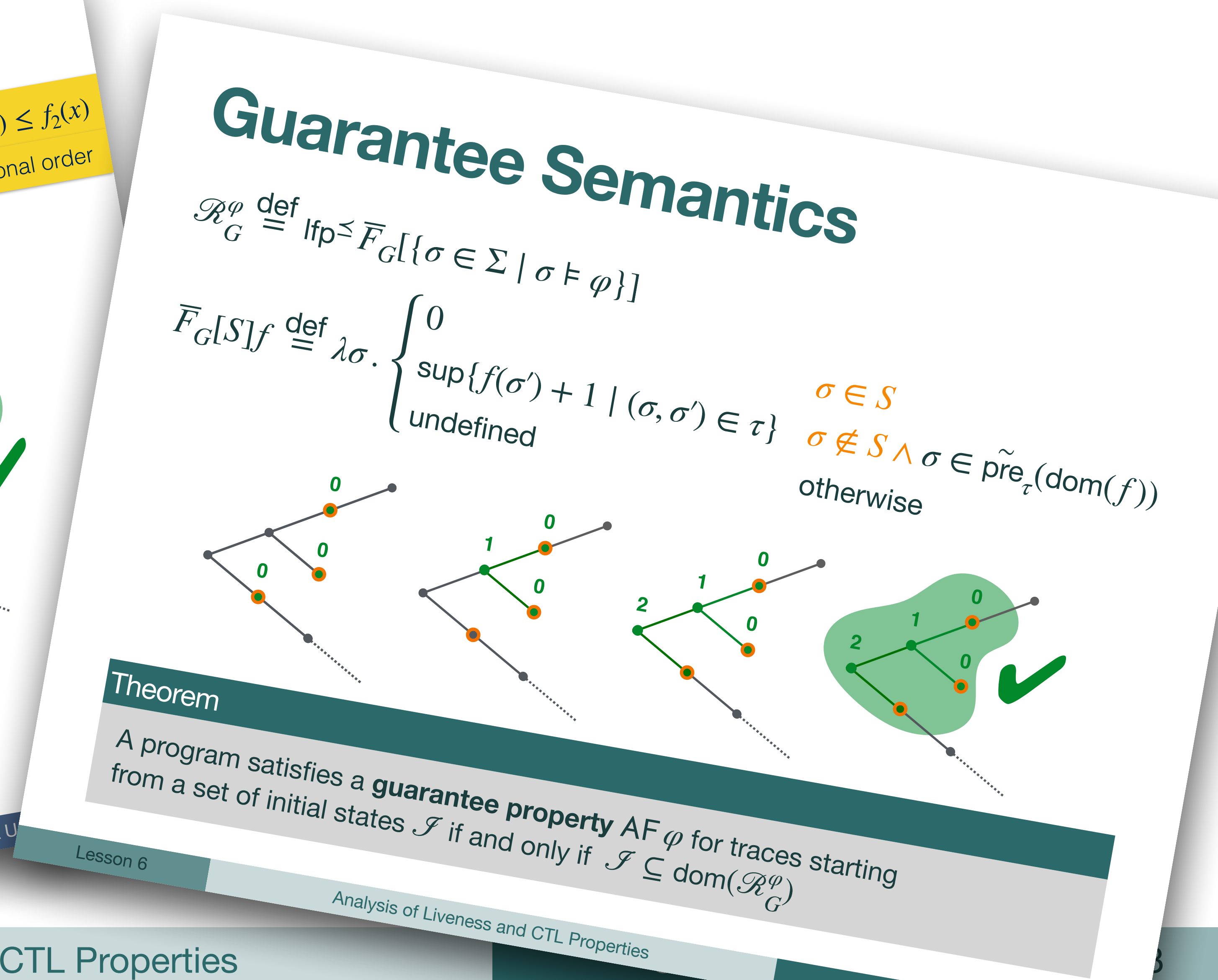
```
1x ← [-∞, +∞]
 while 2(x ≥ 0) do
      3x ← x + 1
od4
 while 5(0 ≥ 0) do
      if 6(x ≤ 10) do
            7x ← x + 1
      else
            8x ← -x
od9
```

Recurrence Semantics

$$\mathcal{R}_R^{\varphi} \stackrel{\text{def}}{=} \text{gfp}_{\mathcal{R}_G^{\varphi}}^{\preceq} \overline{F}_R$$

$$\overline{F}_R(f)\sigma \stackrel{\text{def}}{=} \begin{cases} f(\sigma) & \sigma \in \widetilde{\text{pre}}_\tau(\text{dom}(f)) \\ \text{undefined} & \text{otherwise} \end{cases}$$

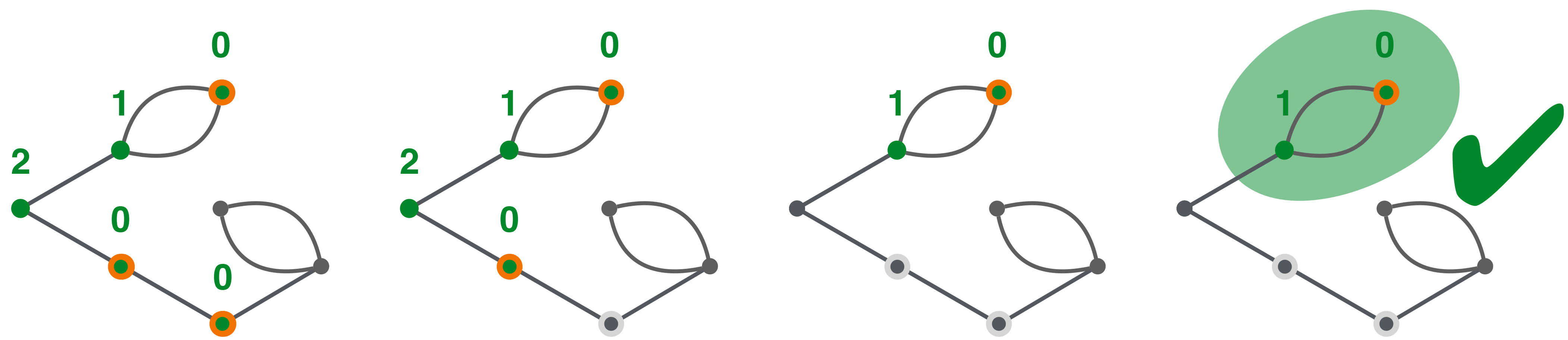




Recurrence Semantics

$$\mathcal{R}_R^{\varphi} \overset{\text{def}}{=} \text{gfp}_{\mathcal{R}_G^{\varphi}}^{\preceq} \overline{F}_R$$

$$\overline{F}_R(f)\sigma \overset{\text{def}}{=} \begin{cases} f(\sigma) & \sigma \in \widetilde{\text{pre}}_\tau(\text{dom}(f)) \\ \text{undefined} & \text{otherwise} \end{cases}$$



Theorem

A program satisfies a **recurrence property** AG AF φ for traces starting from a set of initial states $\mathcal{I}$ if and only if $\mathcal{I} \subseteq \text{dom}(\mathcal{R}_R^{\varphi})$

Abstract Recurrence Semantics

For each program instruction stat, we define $\mathcal{R}_G^{\varphi\#}[\![\text{stat}]\!] : \mathcal{A} \to \mathcal{A}$:

- $\mathcal{R}_R^{\varphi\#}[\![{}^{\ell} X \leftarrow e]\!]t \stackrel{\text{def}}{=} \mathsf{RESET}_A^R[\![\varphi]\!](\overleftarrow{\mathsf{ASSIGN}}_A[\![X \leftarrow e]\!]t)$

- $\mathcal{R}_R^{\varphi\#}[\![\textbf{if } {}^{\ell} e \bowtie 0 \textbf{ then } s]\!]t \stackrel{\text{def}}{=} \mathsf{RESET}_A^R[\![\varphi]\!](X)$
  where $X \stackrel{\text{def}}{=} \mathsf{FILTER}_A[\![e \bowtie 0]\!](\mathcal{R}_G^{\varphi\#}[\![s]\!]t) \curlyvee_T \mathsf{FILTER}_A[\![e \not\bowtie 0]\!]t)$

- $\mathcal{R}_R^{\varphi\#}[\![\textbf{while } {}^{\ell} e \bowtie 0 \textbf{ do } s \textbf{ done}]\!]t \stackrel{\text{def}}{=} \mathsf{gfp}^{\#}_{G(t)} \overline{F}_R^{\varphi\#}$
  where $G \stackrel{\text{def}}{=} \mathcal{R}_G^{\varphi\#}[\![\textbf{while } {}^{\ell} e \bowtie 0 \textbf{ do } s \textbf{ done}]\!]$
  $\overline{F}_R^{\varphi\#}(x) \stackrel{\text{def}}{=} \mathsf{RESET}_A^R[\![\varphi]\!](X)$
  $X \stackrel{\text{def}}{=} \mathsf{FILTER}_A[\![e \bowtie 0]\!](\mathcal{R}_R^{\varphi\#}[\![s]\!]x) \curlyvee_T \mathsf{FILTER}_A[\![e \not\bowtie 0]\!](t))$

- $\mathcal{R}_R^{\varphi\#}[\![s_1; s_2]\!]t \stackrel{\text{def}}{=} \mathcal{R}_R^{\varphi\#}[\![s_1]\!](\mathcal{R}_R^{\varphi\#}[\![s_2]\!]t)$

Abstract Recurrence Semantics

Example

int : x, y

while [1]$(x \geq 0)$ do

  [2]$x := x + 1$

od

while [3](true) do

  if [4]($x \leq 10$)

    [5]$x := x + 1$

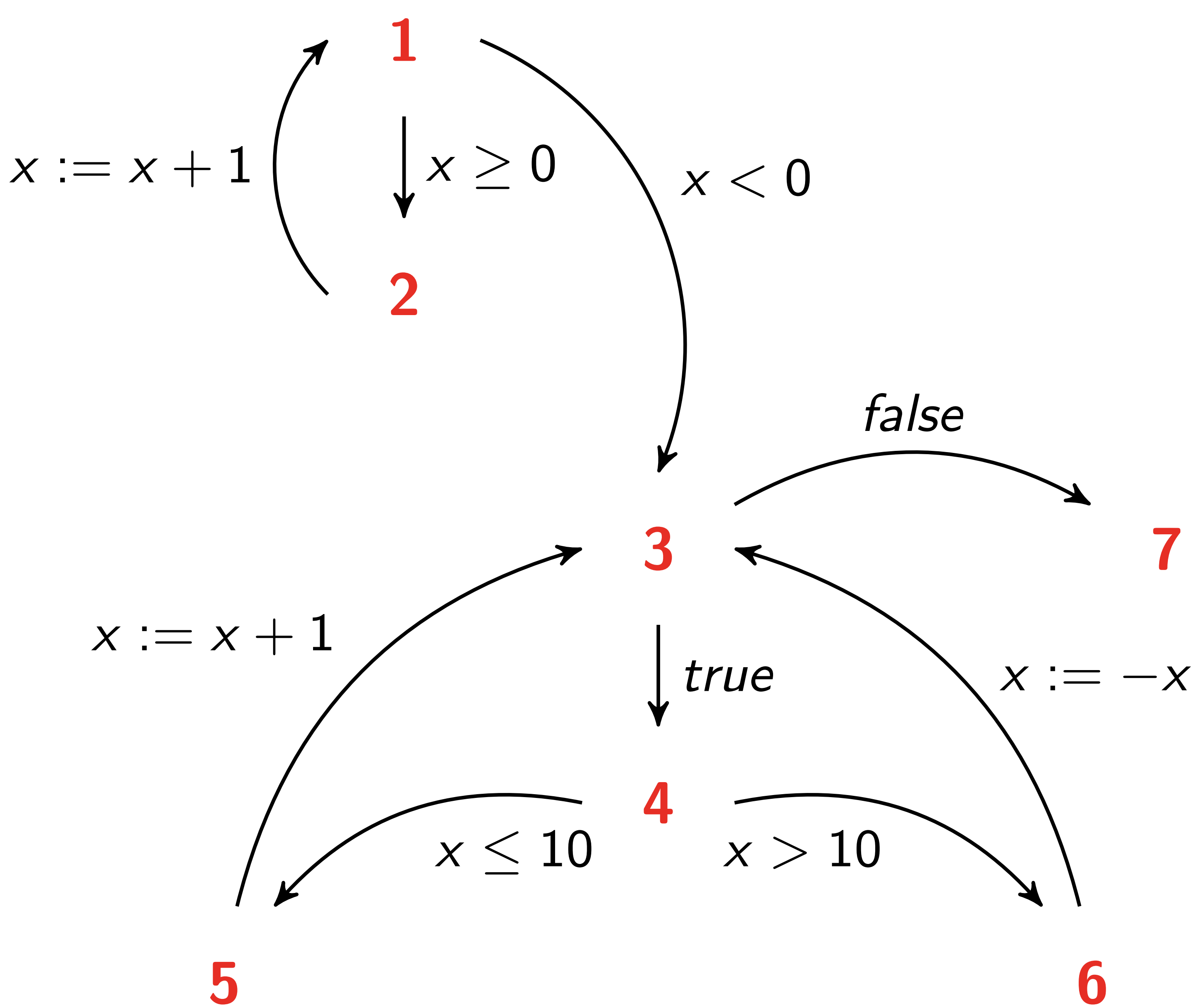
  else

    [6]$x := -x$

od[7]

Property

$\text{AGAF}\,(x = 3)$



Abstract Recurrence Semantics

Example

int : x, y

while [1]$(x \geq 0)$ do

  [2]$x := x + 1$

od

while [3](true) do

  if [4]($x \leq 10$)

    [5]$x := x + 1$

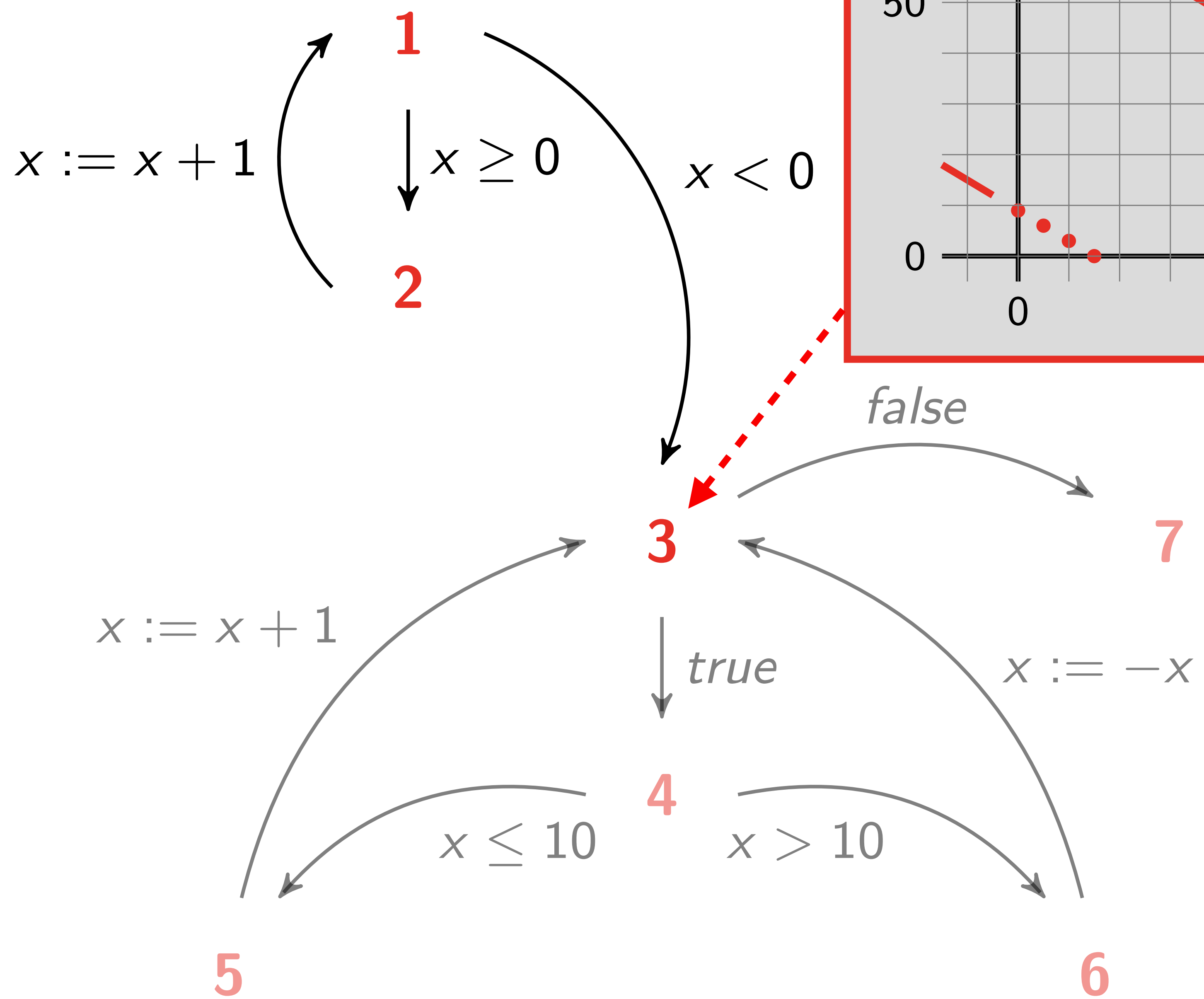
  else

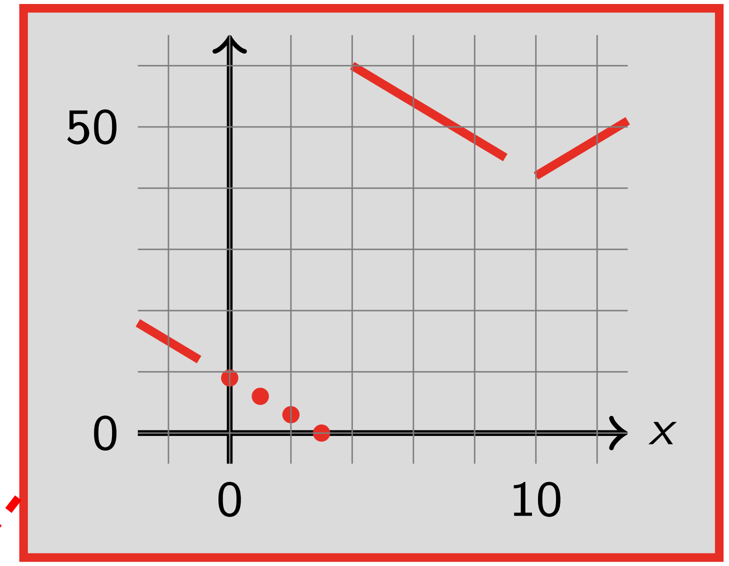
    [6]$x := -x$

od[7]

Property

$$AGAF\,(x = 3)$$





Abstract Recurrence Semantics

Example

int : x, y

while ${}^{1}(x \geq 0)$

${}^{2}x := x + 1$

od

while ${}^{3}($ true $)$ do

if ${}^{4}(\ x \leq 10\ )$

${}^{5}x := x + 1$

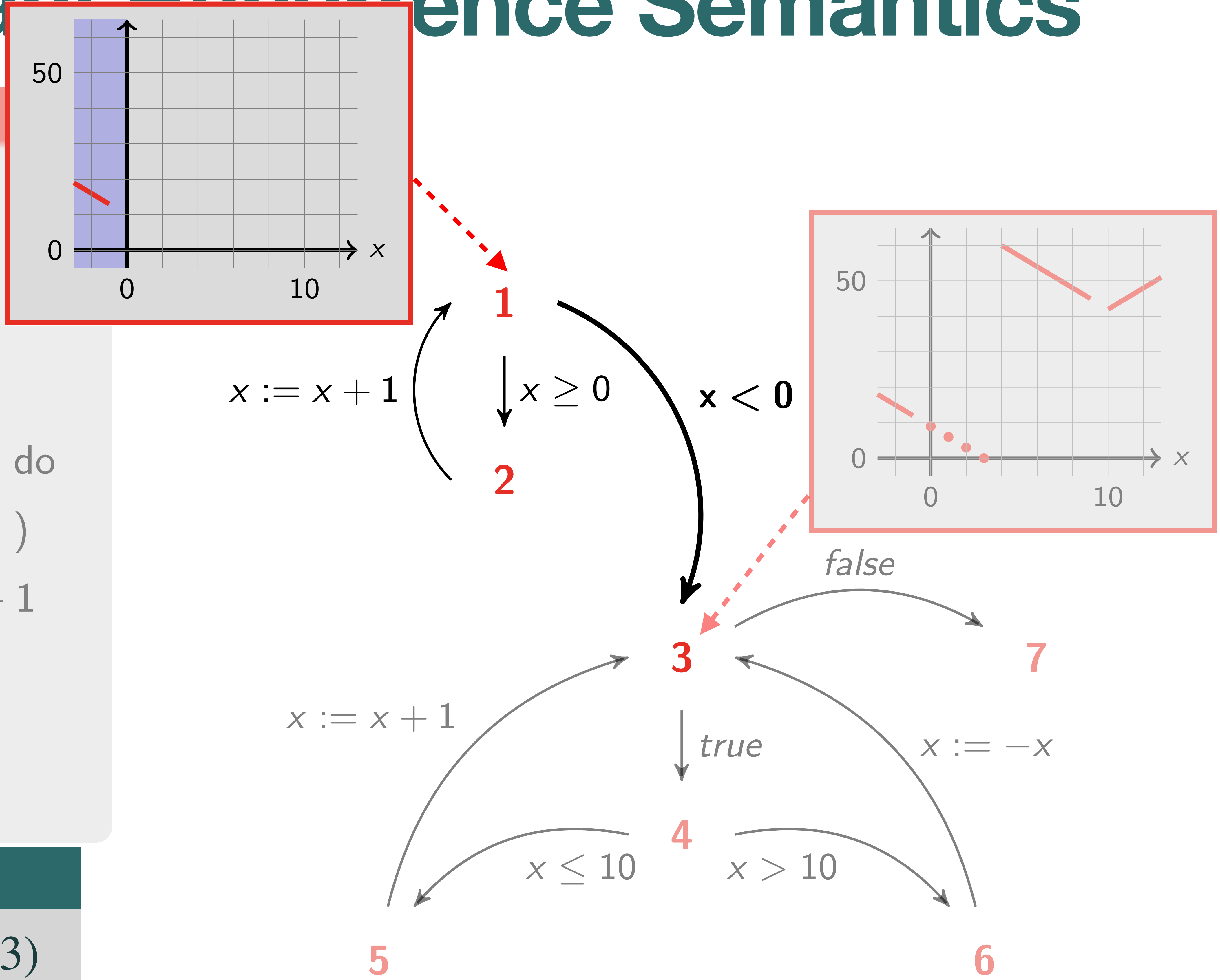
else

${}^{6}x := -x$

od^{7}

Property

$\text{AGAF}\,(x = 3)$



Abstract Recurrence Semantics

Example

int : x, y

while [1]$(x \geq 0)$

[2]$x := x + 1$

od

while [3](true) do

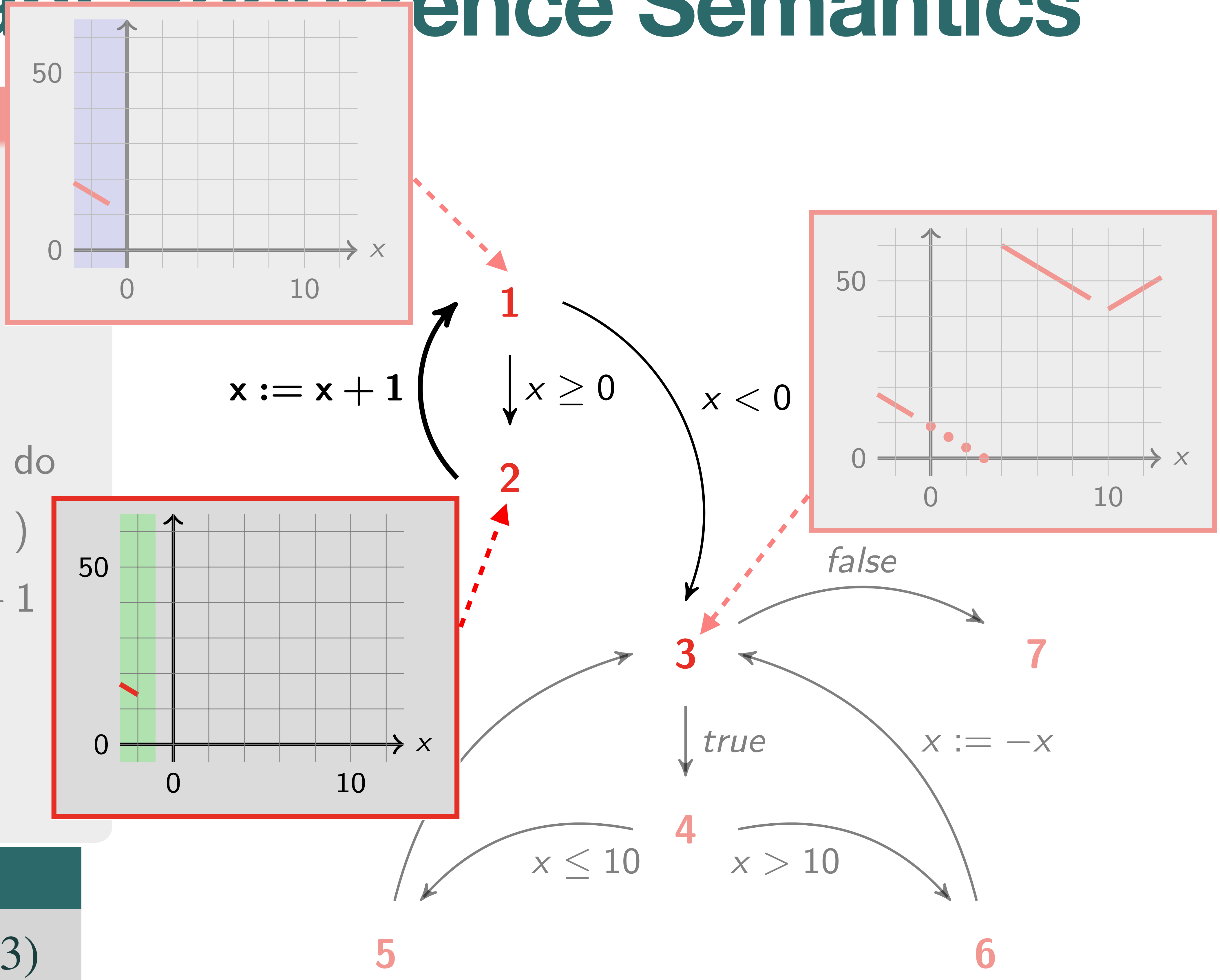
if [4]($x \leq 10$)

[5]$x := x + 1$

else

[6]$x := -x$

od[7]



Property

$\text{AGAF}\,(x = 3)$

Abstract Recurrence Semantics

Example

int : x, y

while [1]$(x \geq 0)$

  [2]$x := x + 1$

od

while [3](true) do

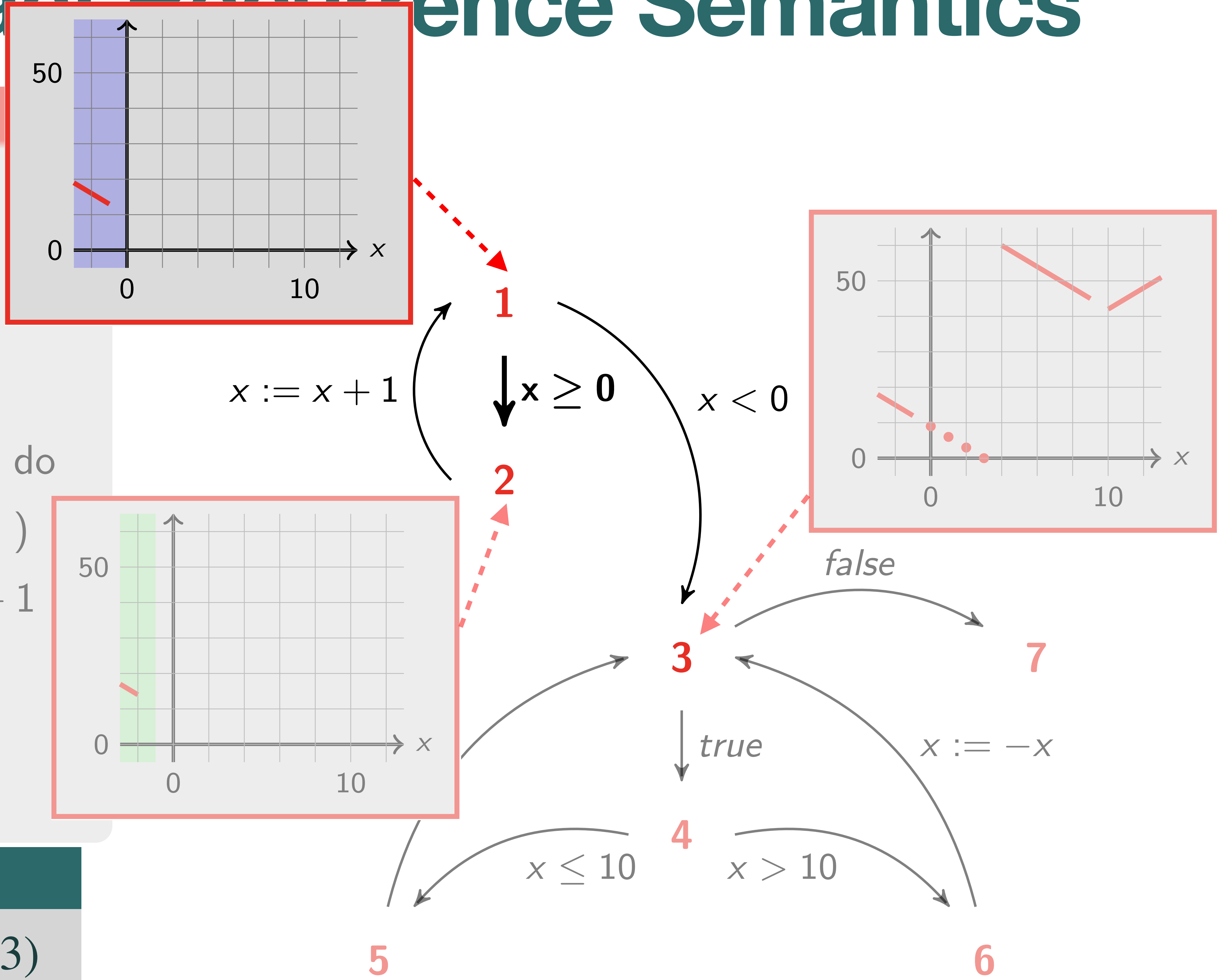
  if [4]($x \leq 10$)

    [5]$x := x + 1$

  else

    [6]$x := -x$

od[7]



Property

$\mathrm{AGAF}\,(x = 3)$

Abstract Recurrence Semantics

Example

int : x, y

while [1]$(x \geq 0)$

[2]$x := x + 1$

od

while [3](true) do

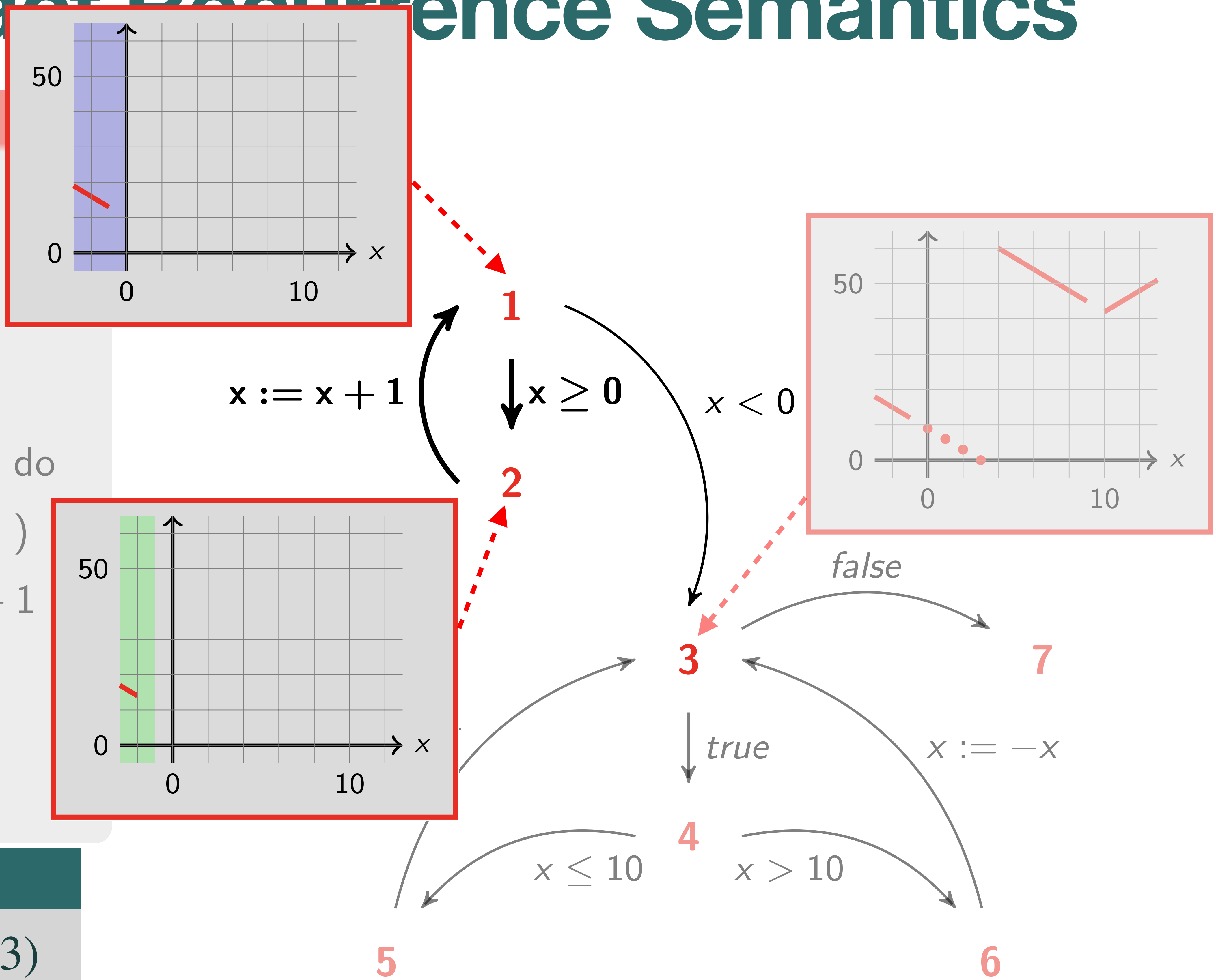
if [4]($x \leq 10$)

[5]$x := x + 1$

else

[6]$x := -x$

od[7]



Property

$\text{AGAF}\,(x = 3)$

Abstract Recurrence Semantics

Example

int : x, y

while $^{1}(x \geq 0)$

  $^{2}x := x + 1$

od

while ^{3}(true) do

  if ^{4}($x \leq 10$)

    $^{5}x := x + 1$

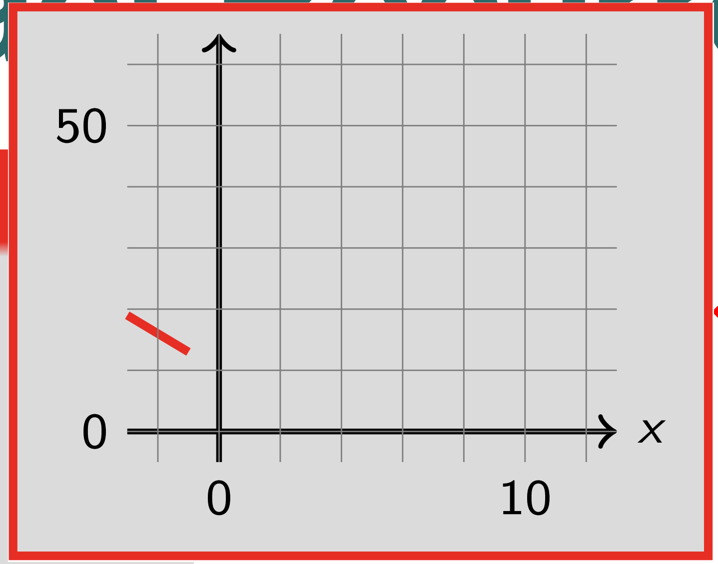
  else

    $^{6}x := -x$

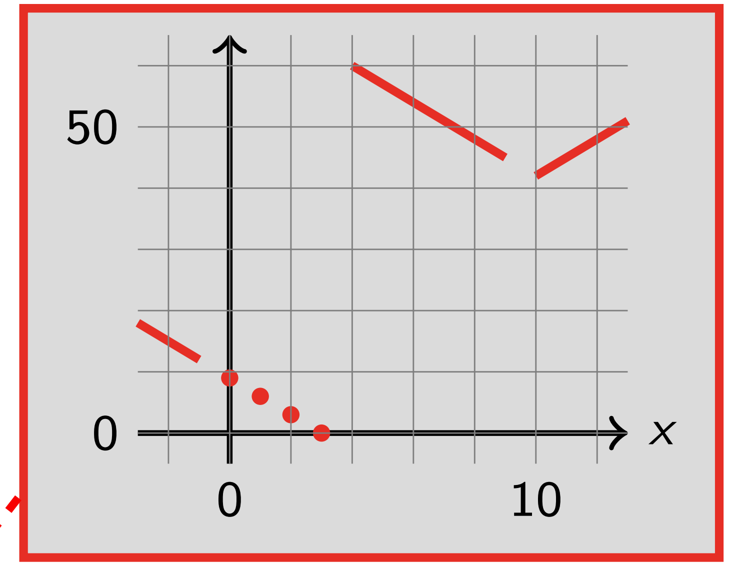
od^{7}

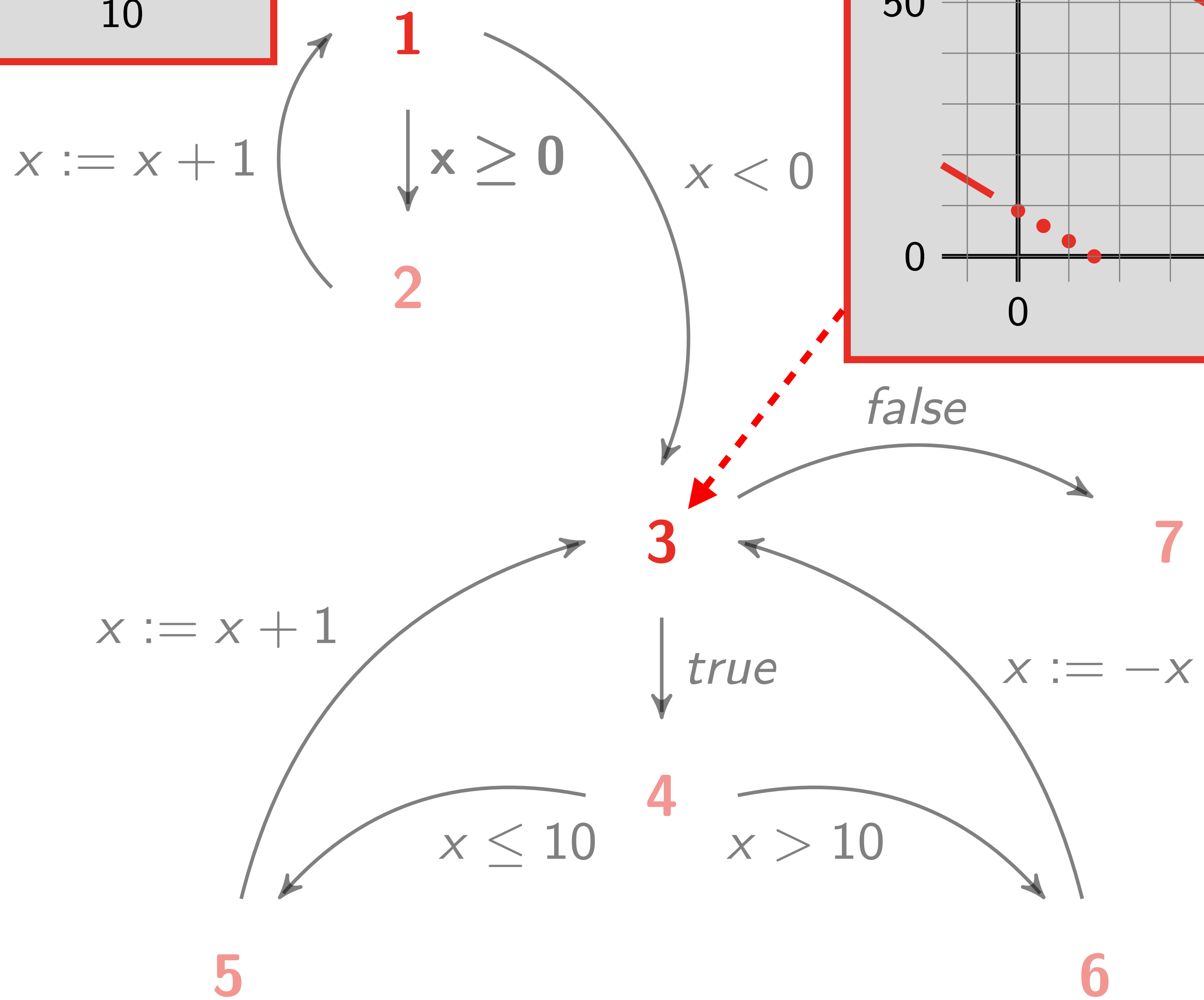
Property

$$\text{AGAF}\,(x = 3)$$



the analysis gives $x < 0$ as **sufficient precondition**





Abstract Recurrence Semantics

Definition

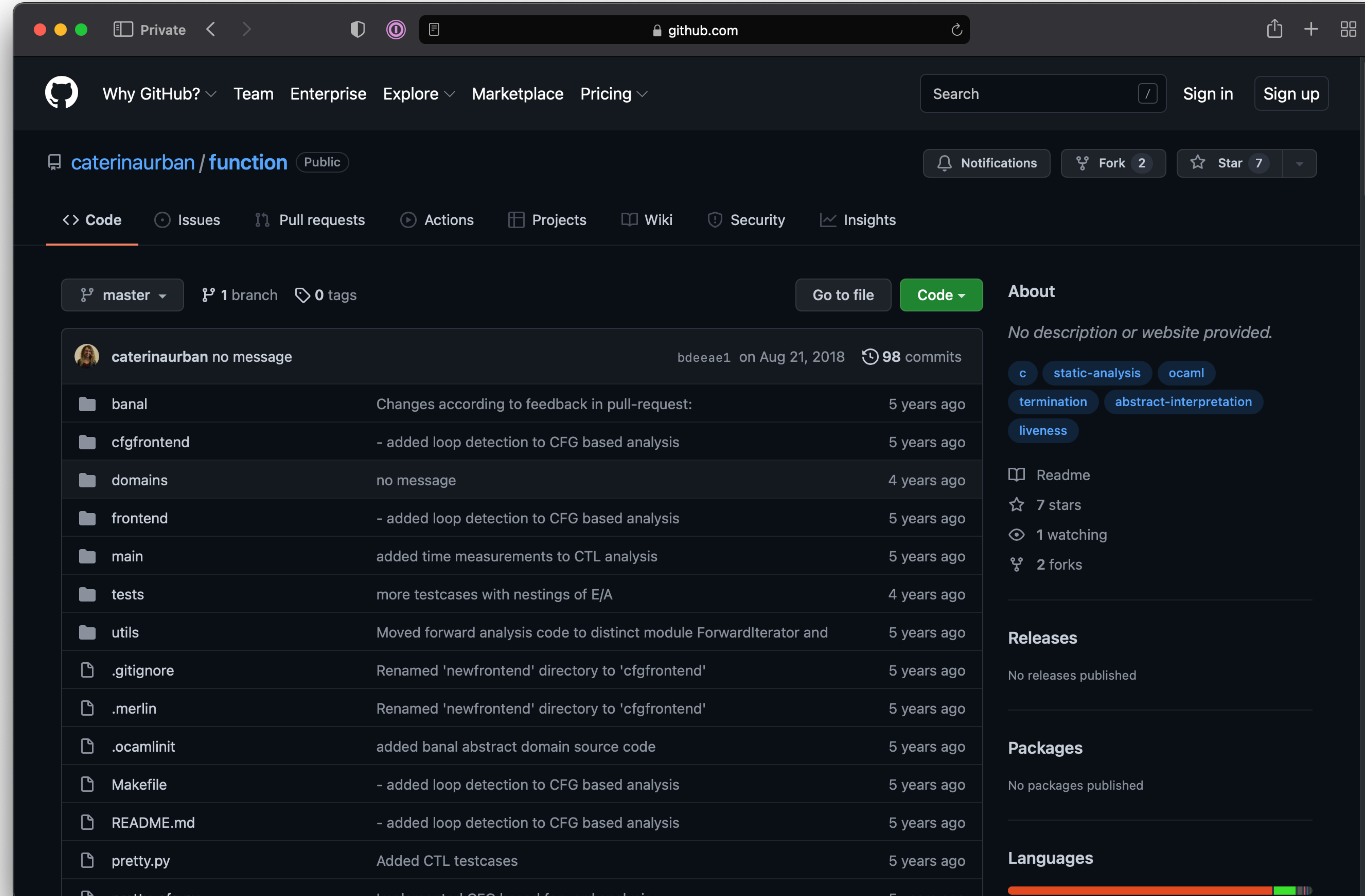
The **abstract recurrence semantics** $\mathcal{R}_R^{\varphi\#}[\![\text{stat}^\ell]\!] \in \mathcal{A}$ of a program stat^ℓ is:

$$\mathcal{R}_R^{\varphi\#}[\![\text{stat}^\ell]\!] \overset{\text{def}}{=} \mathcal{R}_R^{\varphi\#}[\![\text{stat}]\!](\text{LEAF}: \bot_F)$$

where $\mathcal{R}_R^{\varphi\#}[\![\text{stat}]\!] : \mathcal{A} \to \mathcal{A}$ is the abstract recurrence semantics of each program instruction stat

Corollary (Soundness)

A program stat^ℓ satisfies a **recurrence property** AG AF φ for traces starting from a set of initial states $\mathcal{I}$ if $\mathcal{I} \subseteq \text{dom}(\gamma_A(\mathcal{R}_R^{\varphi\#}[\![\text{stat}^\ell]\!]))$


Private
github.com
Why GitHub?
Team
Enterprise
Explore
Marketplace
Pricing
Search
Sign in
Sign up
caterinaurban / function
Public
Notifications
Fork 2
Star 7
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
master
1 branch
0 tags
Go to file
Code
caterinaurban no message
bdeeae1 on Aug 21, 2018
98 commits
banal
Changes according to feedback in pull-request:
5 years ago
cfgfrontend
- added loop detection to CFG based analysis
5 years ago
domains
no message
4 years ago
frontend
- added loop detection to CFG based analysis
5 years ago
main
added time measurements to CTL analysis
5 years ago
tests
more testcases with nestings of E/A
4 years ago
utils
Moved forward analysis code to distinct module ForwardIterator and
5 years ago
.gitignore
Renamed 'newfrontend' directory to 'cfgfrontend'
5 years ago
.merlin
Renamed 'newfrontend' directory to 'cfgfrontend'
5 years ago
.ocamlinit
added banal abstract domain source code
5 years ago
Makefile
- added loop detection to CFG based analysis
5 years ago
README.md
- added loop detection to CFG based analysis
5 years ago
pretty.py
Added CTL testcases
5 years ago
pretty_cfg.py
Implemented CFG based forward analysis
5 years ago
About
No description or website provided.
c
static-analysis
ocaml
termination
abstract-interpretation
liveness
Readme
7 stars
1 watching
2 forks
Releases
No releases published
Packages
No packages published
Languages

Bibliography

[Urban15] **Caterina Urban**. Static Analysis by Abstract Interpretation of Functional Temporal Properties of Programs. PhD Thesis, École Normale Supérieure, 2015.

[Urban17] **Caterina Urban**, Antoine Miné. Inference of Ranking Functions for Proving Temporal Properties by Abstract Interpretation. In COMLAN, 2017.