



Semantics and Verification of Software

Winter Semester 2017/18

Lecture 3: Operational Semantics of WHILE II (Execution of Statements)

Thomas Noll

Software Modeling and Verification Group

RWTH Aachen University

<http://moves.rwth-aachen.de/teaching/ws-1718/sv-sw/>

Recap: Evaluation Relations & Structural Induction

Outline of Lecture 3

Recap: Evaluation Relations & Structural Induction

Execution of Statements

Determinism of Evaluation/Execution

Recap: Evaluation Relations & Structural Induction

Evaluation of Arithmetic Expressions

Remember: $a ::= z \mid x \mid a_1 + a_2 \mid a_1 - a_2 \mid a_1 * a_2 \in AExp$

Definition (Evaluation relation for arithmetic expressions)

If $a \in AExp$ and $\sigma \in \Sigma$, then $\langle a, \sigma \rangle$ is called a **configuration**.

Expression a **evaluates to** $z \in \mathbb{Z}$ in state σ (notation: $\langle a, \sigma \rangle \rightarrow z$) if this relationship is derivable by means of the following rules:

Axioms:

$$\frac{}{\langle z, \sigma \rangle \rightarrow z} \quad \frac{}{\langle x, \sigma \rangle \rightarrow \sigma(x)}$$

Rules:

$$\frac{\langle a_1, \sigma \rangle \rightarrow z_1 \quad \langle a_2, \sigma \rangle \rightarrow z_2}{\langle a_1 + a_2, \sigma \rangle \rightarrow z} \quad \text{where } z := z_1 + z_2$$

$$\frac{\langle a_1, \sigma \rangle \rightarrow z_1 \quad \langle a_2, \sigma \rangle \rightarrow z_2}{\langle a_1 - a_2, \sigma \rangle \rightarrow z} \quad \text{where } z := z_1 - z_2$$

$$\frac{\langle a_1, \sigma \rangle \rightarrow z_1 \quad \langle a_2, \sigma \rangle \rightarrow z_2}{\langle a_1 * a_2, \sigma \rangle \rightarrow z} \quad \text{where } z := z_1 \cdot z_2$$

Recap: Evaluation Relations & Structural Induction

Excursus: Proof by Structural Induction

Proof principle

Given: an inductive set, i.e., a set S whose elements are either

- atomic or
- obtained from atomic elements by (finite) application of certain operations

To show: property $P(s)$ applies to every $s \in S$

Proof: we verify:

Induction base: $P(s)$ holds for every atomic element s

Induction hypothesis: assume that $P(s_1)$, $P(s_2)$ etc.

Induction step: then also $P(f(s_1, \dots, s_n))$ holds for every operation f of arity n

Remark: structural induction is a special case of **well-founded induction**

Recap: Evaluation Relations & Structural Induction

Free Variables

Lemma

Let $a \in AExp$ and $\sigma, \sigma' \in \Sigma$ such that $\sigma(x) = \sigma'(x)$ for every $x \in FV(a)$. Then, for every $z \in \mathbb{Z}$,

$$\langle a, \sigma \rangle \rightarrow z \iff \langle a, \sigma' \rangle \rightarrow z.$$

Proof.

by **structural induction** on a (on the board)



Recap: Evaluation Relations & Structural Induction

Evaluation of Boolean Expressions

Definition ((Strict) evaluation relation for Boolean expressions)

For $b \in BExp$, $\sigma \in \Sigma$, and $t \in \mathbb{B}$, the **evaluation relation** $\langle b, \sigma \rangle \rightarrow t$ is defined by:

$$\begin{array}{c} \frac{\langle t, \sigma \rangle \rightarrow t}{\langle a_1, \sigma \rangle \rightarrow z \ \langle a_2, \sigma \rangle \rightarrow z} \quad \frac{\langle a_1, \sigma \rangle \rightarrow z_1 \ \langle a_2, \sigma \rangle \rightarrow z_2}{\langle a_1 = a_2, \sigma \rangle \rightarrow \text{true}} \quad \text{if } z_1 \neq z_2 \\ \frac{\langle a_1, \sigma \rangle \rightarrow z_1 \ \langle a_2, \sigma \rangle \rightarrow z_2}{\langle a_1 > a_2, \sigma \rangle \rightarrow \text{true}} \quad \text{if } z_1 > z_2 \quad \frac{\langle a_1, \sigma \rangle \rightarrow z_1 \ \langle a_2, \sigma \rangle \rightarrow z_2}{\langle a_1 > a_2, \sigma \rangle \rightarrow \text{false}} \quad \text{if } z_1 \leq z_2 \\ \frac{\langle b, \sigma \rangle \rightarrow \text{false}}{\langle \neg b, \sigma \rangle \rightarrow \text{true}} \quad \frac{\langle b, \sigma \rangle \rightarrow \text{true}}{\langle \neg b, \sigma \rangle \rightarrow \text{false}} \\ \frac{\langle b_1, \sigma \rangle \rightarrow \text{true} \ \langle b_2, \sigma \rangle \rightarrow \text{true}}{\langle b_1 \wedge b_2, \sigma \rangle \rightarrow \text{true}} \quad \frac{\langle b_1, \sigma \rangle \rightarrow \text{true} \ \langle b_2, \sigma \rangle \rightarrow \text{false}}{\langle b_1 \wedge b_2, \sigma \rangle \rightarrow \text{false}} \\ \frac{\langle b_1, \sigma \rangle \rightarrow \text{false} \ \langle b_2, \sigma \rangle \rightarrow \text{true}}{\langle b_1 \wedge b_2, \sigma \rangle \rightarrow \text{false}} \quad \frac{\langle b_1, \sigma \rangle \rightarrow \text{false} \ \langle b_2, \sigma \rangle \rightarrow \text{false}}{\langle b_1 \wedge b_2, \sigma \rangle \rightarrow \text{false}} \end{array}$$

(\vee analogously)

Execution of Statements

Outline of Lecture 3

Recap: Evaluation Relations & Structural Induction

Execution of Statements

Determinism of Evaluation/Execution

Execution of Statements

Meaning of Statements

Effect of statement = **modification of program state**

Execution of Statements

Meaning of Statements

Effect of statement = **modification of program state**

Example 3.1

Goal: define **execution relation** \rightarrow such that, e.g.,

$$\langle x := 5, \sigma \rangle \rightarrow \sigma[x \mapsto 5]$$

where for every $\sigma \in \Sigma$, $x, y \in Var$, and $z \in \mathbb{Z}$:

$$\sigma[x \mapsto z](y) := \begin{cases} z & \text{if } y = x \\ \sigma(y) & \text{otherwise} \end{cases}$$

Execution of Statements

Execution of Statements

Remember:

$c ::= \text{skip} \mid x := a \mid c_1; c_2 \mid \text{if } b \text{ then } c_1 \text{ else } c_2 \text{ end} \mid \text{while } b \text{ do } c \text{ end} \in \text{Cmd}$

Execution of Statements

Execution of Statements

Remember:

$c ::= \text{skip} \mid x := a \mid c_1; c_2 \mid \text{if } b \text{ then } c_1 \text{ else } c_2 \text{ end} \mid \text{while } b \text{ do } c \text{ end} \in \text{Cmd}$

Definition 3.2 (Execution relation for statements)

For $c \in \text{Cmd}$ and $\sigma, \sigma' \in \Sigma$, the **execution relation** $\langle c, \sigma \rangle \rightarrow \sigma'$ is defined by:

$$\begin{array}{c} \frac{}{\langle \text{skip}, \sigma \rangle \rightarrow \sigma} \text{(skip)} \\ \frac{\langle c_1, \sigma \rangle \rightarrow \sigma' \quad \langle c_2, \sigma' \rangle \rightarrow \sigma''}{\langle c_1; c_2, \sigma \rangle \rightarrow \sigma''} \text{(seq)} \\ \frac{\langle b, \sigma \rangle \rightarrow \text{false} \quad \langle c_2, \sigma \rangle \rightarrow \sigma'}{\langle \text{if } b \text{ then } c_1 \text{ else } c_2 \text{ end}, \sigma \rangle \rightarrow \sigma'} \text{(if-f)} \\ \frac{\langle b, \sigma \rangle \rightarrow \text{true} \quad \langle c, \sigma \rangle \rightarrow \sigma' \quad \langle \text{while } b \text{ do } c \text{ end}, \sigma' \rangle \rightarrow \sigma''}{\langle \text{while } b \text{ do } c \text{ end}, \sigma \rangle \rightarrow \sigma''} \text{(wh-t)} \\ \frac{}{\langle a, \sigma \rangle \rightarrow z} \text{(asgn)} \\ \frac{\langle b, \sigma \rangle \rightarrow \text{true} \quad \langle c_1, \sigma \rangle \rightarrow \sigma'}{\langle \text{if } b \text{ then } c_1 \text{ else } c_2 \text{ end}, \sigma \rangle \rightarrow \sigma'} \text{(if-t)} \\ \frac{\langle b, \sigma \rangle \rightarrow \text{false}}{\langle \text{while } b \text{ do } c \text{ end}, \sigma \rangle \rightarrow \sigma} \text{(wh-f)} \end{array}$$

Execution of Statements

An Execution Example

Example 3.3

• $c := y := 1; \text{ while } \underbrace{\neg(x=1)}_b \text{ do } \underbrace{y := y*x}_{c_1}; \underbrace{x := x-1}_{c_2} \text{ end}$

$\underbrace{\hspace{15em}}_{c_0}$

- Claim: $\langle c, \sigma \rangle \rightarrow \sigma_{1,6}$ for every $\sigma \in \Sigma$ with $\sigma(x) = 3$
- Notation: $\sigma_{i,j}$ means $\sigma(x) = i, \sigma(y) = j$
- Derivation tree: on the board

Execution of Statements

Non-Terminating Statements

Corollary 3.4

*The execution relation for statements is not **total**, i.e., there exist $c \in \text{Cmd}$ and $\sigma \in \Sigma$ such that $\langle c, \sigma \rangle \rightarrow \sigma'$ for no $\sigma' \in \Sigma$.*

Execution of Statements

Non-Terminating Statements

Corollary 3.4

The execution relation for statements is not *total*, i.e., there exist $c \in \text{Cmd}$ and $\sigma \in \Sigma$ such that $\langle c, \sigma \rangle \rightarrow \sigma'$ for no $\sigma' \in \Sigma$.

Proof.

Example: $c = \text{while true do skip end}$ (proof by contradiction; on the board) □

Determinism of Evaluation/Execution

Outline of Lecture 3

Recap: Evaluation Relations & Structural Induction

Execution of Statements

Determinism of Evaluation/Execution

Determinism of Evaluation/Execution

Determinism of Execution Relation I

This operational semantics is well defined in the following sense:

Theorem 3.5

*The execution relation for statements is **deterministic**, i.e., whenever $c \in \text{Cmd}$ and $\sigma, \sigma', \sigma'' \in \Sigma$ such that $\langle c, \sigma \rangle \rightarrow \sigma'$ and $\langle c, \sigma \rangle \rightarrow \sigma''$, then $\sigma' = \sigma''$.*

Determinism of Evaluation/Execution

Determinism of Execution Relation I

This operational semantics is well defined in the following sense:

Theorem 3.5

*The execution relation for statements is **deterministic**, i.e., whenever $c \in \text{Cmd}$ and $\sigma, \sigma', \sigma'' \in \Sigma$ such that $\langle c, \sigma \rangle \rightarrow \sigma'$ and $\langle c, \sigma \rangle \rightarrow \sigma''$, then $\sigma' = \sigma''$.*

The proof is based on the corresponding result for expressions.

Determinism of Evaluation/Execution

Determinism of Evaluation Relations

Lemma 3.6

1. For every $a \in AExp$, $\sigma \in \Sigma$, and $z, z' \in \mathbb{Z}$:
 $\langle a, \sigma \rangle \rightarrow z$ and $\langle a, \sigma \rangle \rightarrow z'$ implies $z = z'$.
2. For every $b \in BExp$, $\sigma \in \Sigma$, and $t, t' \in \mathbb{B}$:
 $\langle b, \sigma \rangle \rightarrow t$ and $\langle b, \sigma \rangle \rightarrow t'$ implies $t = t'$.

Determinism of Evaluation/Execution

Determinism of Evaluation Relations

Lemma 3.6

1. For every $a \in AExp$, $\sigma \in \Sigma$, and $z, z' \in \mathbb{Z}$:
 $\langle a, \sigma \rangle \rightarrow z$ and $\langle a, \sigma \rangle \rightarrow z'$ implies $z = z'$.
2. For every $b \in BExp$, $\sigma \in \Sigma$, and $t, t' \in \mathbb{B}$:
 $\langle b, \sigma \rangle \rightarrow t$ and $\langle b, \sigma \rangle \rightarrow t'$ implies $t = t'$.

Remarks:

- Lemma 3.6(1) is **not** implied by Lemma 2.6
 (“ $\sigma|_{FV(a)} = \sigma'|_{FV(a)} \Rightarrow (\langle a, \sigma \rangle \rightarrow z \iff \langle a, \sigma' \rangle \rightarrow z)$ ”)!)

The latter just implies

$$\{z \in \mathbb{Z} \mid \langle a, \sigma \rangle \rightarrow z\} = \{z \in \mathbb{Z} \mid \langle a, \sigma' \rangle \rightarrow z\}$$

while Lemma 3.6(1) states that

$$|\{z \in \mathbb{Z} \mid \langle a, \sigma \rangle \rightarrow z\}| \leq 1.$$

- Lemma 3.6 can easily be shown by **induction on the structure of expressions**.

Determinism of Evaluation/Execution

Excursus: Proof by Structural Induction V

Application: Boolean expressions (Def. 1.2)

Definition: $BExp$ is the least set which

- contains the truth values $t \in \mathbb{B}$ and, for every $a_1, a_2 \in AExp$, $a_1 = a_2$ and $a_1 > a_2$, and
- contains $\neg b_1$, $b_1 \wedge b_2$ and $b_1 \vee b_2$ whenever $b_1, b_2 \in BExp$

Induction base: $P(t)$, $P(a_1 = a_2)$ and $P(a_1 > a_2)$ holds

(for every $t \in \mathbb{B}$, $a_1, a_2 \in AExp$)

Induction hypothesis: $P(b_1)$ and $P(b_2)$ holds

Induction step: $P(\neg b_1)$, $P(b_1 \wedge b_2)$ and $P(b_1 \vee b_2)$ holds

Determinism of Evaluation/Execution

Excursus: Proof by Structural Induction V

Application: Boolean expressions (Def. 1.2)

Definition: $BExp$ is the least set which

- contains the truth values $t \in \mathbb{B}$ and, for every $a_1, a_2 \in AExp$, $a_1 = a_2$ and $a_1 > a_2$, and
- contains $\neg b_1$, $b_1 \wedge b_2$ and $b_1 \vee b_2$ whenever $b_1, b_2 \in BExp$

Induction base: $P(t)$, $P(a_1 = a_2)$ and $P(a_1 > a_2)$ holds
(for every $t \in \mathbb{B}$, $a_1, a_2 \in AExp$)

Induction hypothesis: $P(b_1)$ and $P(b_2)$ holds

Induction step: $P(\neg b_1)$, $P(b_1 \wedge b_2)$ and $P(b_1 \vee b_2)$ holds

Proof (Lemma 3.6).

1. by structural induction on a (omitted)
2. by structural induction on b (omitted)



Determinism of Execution Relation II

- How to prove that $\langle c, \sigma \rangle \rightarrow \sigma'$ is deterministic (Theorem 3.5)?
- Idea: use **induction on the syntactic structure** of c

Excursus: Proof by Structural Induction VI

Application: syntax of WHILE statements (Def. 1.2)

Definition: Cmd is the least set which

- contains `skip` and, for every $x \in Var$ and $a \in AExp$, $x := a$, and
- contains $c_1 ; c_2$, `if b then c_1 else c_2 end` and `while b do c_1 end` whenever $b \in BExp$ and $c_1, c_2 \in Cmd$

Induction base: $P(\text{skip})$ and $P(x := a)$ holds (for every $x \in Var$ and $a \in AExp$)

Induction hypothesis: $P(c_1)$ and $P(c_2)$ holds

Induction step: $P(c_1 ; c_2)$, $P(\text{if } b \text{ then } c_1 \text{ else } c_2 \text{ end})$ and $P(\text{while } b \text{ do } c_1 \text{ end})$ holds (for every $b \in BExp$)

Determinism of Execution Relation III

- But: proof of Theorem 3.5 fails!

Determinism of Evaluation/Execution

Determinism of Execution Relation III

- But: proof of Theorem 3.5 fails!
- Problematic case:

$c = \text{while } b \text{ do } c_0 \text{ end}$ where $\langle b, \sigma \rangle \rightarrow \text{true}$

Determinism of Evaluation/Execution

Determinism of Execution Relation III

- But: **proof of Theorem 3.5 fails!**
- Problematic case:

$c = \text{while } b \text{ do } c_0 \text{ end}$ where $\langle b, \sigma \rangle \rightarrow \text{true}$

- Here $\langle c, \sigma \rangle \rightarrow \sigma'$ and $\langle c, \sigma \rangle \rightarrow \sigma''$ require existence of $\sigma_1, \sigma_2 \in \Sigma$ such that

$$\text{(wh-t)} \frac{\langle b, \sigma \rangle \rightarrow \text{true} \quad \langle c_0, \sigma \rangle \rightarrow \sigma_1 \quad \langle c, \sigma_1 \rangle \rightarrow \sigma'}{\langle c, \sigma \rangle \rightarrow \sigma'}$$

and

$$\text{(wh-t)} \frac{\langle b, \sigma \rangle \rightarrow \text{true} \quad \langle c_0, \sigma \rangle \rightarrow \sigma_2 \quad \langle c, \sigma_2 \rangle \rightarrow \sigma''}{\langle c, \sigma \rangle \rightarrow \sigma''}$$

Determinism of Evaluation/Execution

Determinism of Execution Relation III

- But: **proof of Theorem 3.5 fails!**
- Problematic case:

$c = \text{while } b \text{ do } c_0 \text{ end}$ where $\langle b, \sigma \rangle \rightarrow \text{true}$

- Here $\langle c, \sigma \rangle \rightarrow \sigma'$ and $\langle c, \sigma \rangle \rightarrow \sigma''$ require existence of $\sigma_1, \sigma_2 \in \Sigma$ such that

$$\text{(wh-t)} \frac{\langle b, \sigma \rangle \rightarrow \text{true} \quad \langle c_0, \sigma \rangle \rightarrow \sigma_1 \quad \langle c, \sigma_1 \rangle \rightarrow \sigma'}{\langle c, \sigma \rangle \rightarrow \sigma'}$$

and

$$\text{(wh-t)} \frac{\langle b, \sigma \rangle \rightarrow \text{true} \quad \langle c_0, \sigma \rangle \rightarrow \sigma_2 \quad \langle c, \sigma_2 \rangle \rightarrow \sigma''}{\langle c, \sigma \rangle \rightarrow \sigma''}$$

- c_0 proper substatement of c
 \Rightarrow induction hypothesis yields $\sigma_1 = \sigma_2$

Determinism of Evaluation/Execution

Determinism of Execution Relation III

- But: **proof of Theorem 3.5 fails!**
- Problematic case:

$c = \text{while } b \text{ do } c_0 \text{ end}$ where $\langle b, \sigma \rangle \rightarrow \text{true}$

- Here $\langle c, \sigma \rangle \rightarrow \sigma'$ and $\langle c, \sigma \rangle \rightarrow \sigma''$ require existence of $\sigma_1, \sigma_2 \in \Sigma$ such that

$$\text{(wh-t)} \frac{\langle b, \sigma \rangle \rightarrow \text{true} \quad \langle c_0, \sigma \rangle \rightarrow \sigma_1 \quad \langle c, \sigma_1 \rangle \rightarrow \sigma'}{\langle c, \sigma \rangle \rightarrow \sigma'}$$

and

$$\text{(wh-t)} \frac{\langle b, \sigma \rangle \rightarrow \text{true} \quad \langle c_0, \sigma \rangle \rightarrow \sigma_2 \quad \langle c, \sigma_2 \rangle \rightarrow \sigma''}{\langle c, \sigma \rangle \rightarrow \sigma''}$$

- c_0 proper substatement of c
 \Rightarrow induction hypothesis yields $\sigma_1 = \sigma_2$
- c **not** proper substatement of $c \Rightarrow$ **conclusion $\sigma' = \sigma''$ invalid!**

Determinism of Evaluation/Execution

Excursus: Proof by Structural Induction VII

Application: derivation trees of execution relation (Def. 3.2)

(skip): for every $\sigma \in \Sigma$, $\frac{}{\langle \text{skip}, \sigma \rangle \rightarrow \sigma}$ is a derivation tree for $\langle \text{skip}, \sigma \rangle \rightarrow \sigma$

(asgn): if s is a derivation tree for $\langle a, \sigma \rangle \rightarrow z$ (Def. 2.2), then $\frac{s}{\langle x := a, \sigma \rangle \rightarrow \sigma[x \mapsto z]}$ is a derivation tree for $\langle x := a, \sigma \rangle \rightarrow \sigma[x \mapsto z]$

(seq): if s_1 and s_2 are derivation trees for $\langle c_1, \sigma \rangle \rightarrow \sigma'$ and, respectively, $\langle c_2, \sigma' \rangle \rightarrow \sigma''$, then $\frac{s_1 \ s_2}{\langle c_1 ; c_2, \sigma \rangle \rightarrow \sigma''}$ is a derivation tree for $\langle c_1 ; c_2, \sigma \rangle \rightarrow \sigma''$

(if-t): if s_1 and s_2 are derivation trees for $\langle b, \sigma \rangle \rightarrow \text{true}$ (Def. 2.7) and, respectively, $\langle c_1, \sigma \rangle \rightarrow \sigma'$, then $\frac{s_1 \ s_2}{\langle \text{if } b \text{ then } c_1 \text{ else } c_2 \text{ end}, \sigma \rangle \rightarrow \sigma'}$ is a derivation tree for $\langle \text{if } b \text{ then } c_1 \text{ else } c_2 \text{ end}, \sigma \rangle \rightarrow \sigma'$

(if-f): analogously

(wh-t): if s_1 , s_2 and s_3 are derivation trees for $\langle b, \sigma \rangle \rightarrow \text{true}$ (Def. 2.7), $\langle c, \sigma \rangle \rightarrow \sigma'$ and $\langle \text{while } b \text{ do } c \text{ end}, \sigma' \rangle \rightarrow \sigma''$, respectively, then $\frac{s_1 \ s_2 \ s_3}{\langle \text{while } b \text{ do } c \text{ end}, \sigma \rangle \rightarrow \sigma''}$ is a derivation tree for $\langle \text{while } b \text{ do } c \text{ end}, \sigma \rangle \rightarrow \sigma''$

(wh-f): if s is a derivation tree for $\langle b, \sigma \rangle \rightarrow \text{false}$ (Def. 2.7), then $\frac{s}{\langle \text{while } b \text{ do } c \text{ end}, \sigma \rangle \rightarrow \sigma}$ is a derivation tree for $\langle \text{while } b \text{ do } c \text{ end}, \sigma \rangle \rightarrow \sigma$

Determinism of Evaluation/Execution

Excursus: Proof by Structural Induction VIII

Application: derivation trees of execution relation (continued)

Induction base: $P \left(\frac{}{\langle \text{skip}, \sigma \rangle \rightarrow \sigma} \right)$ holds for every $\sigma \in \Sigma$, and $P(s)$ holds for every derivation tree s for an arithmetic or Boolean expression.

Induction hypothesis: $P(s_1)$, $P(s_2)$ und $P(s_3)$ hold.

Induction step: it also holds that

$$\bullet P \left(\frac{s_1}{\text{(asgn)} \langle x := a, \sigma \rangle \rightarrow \sigma[x \mapsto z]} \right)$$

$$\bullet P \left(\frac{s_1 \ s_2}{\text{(seq)} \langle c_1 ; c_2, \sigma \rangle \rightarrow \sigma''} \right)$$

$$\bullet P \left(\frac{s_1 \ s_2}{\text{(if-t)} \langle \text{if } b \text{ then } c_1 \text{ else } c_2 \text{ end}, \sigma \rangle \rightarrow \sigma'} \right)$$

$$\bullet P \left(\frac{s_1 \ s_2}{\text{(if-f)} \langle \text{if } b \text{ then } c_1 \text{ else } c_2 \text{ end}, \sigma \rangle \rightarrow \sigma'} \right)$$

$$\bullet P \left(\frac{s_1 \ s_2 \ s_3}{\text{(wh-t)} \langle \text{while } b \text{ do } c \text{ end}, \sigma \rangle \rightarrow \sigma''} \right)$$

$$\bullet P \left(\frac{s_1}{\text{(wh-f)} \langle \text{while } b \text{ do } c \text{ end}, \sigma \rangle \rightarrow \sigma} \right)$$

Determinism of Evaluation/Execution

Determinism of Execution Relation IV

Proof (Theorem 3.5).

To show:

$$\langle c, \sigma \rangle \rightarrow \sigma', \langle c, \sigma \rangle \rightarrow \sigma'' \Rightarrow \sigma' = \sigma''$$

(by structural induction on derivation trees; on the board)

