Semantikdefinitionsstile (1)

Es gibt unterschiedliche Stile, die Semantik einer Programmiersprache festzulegen. Sie richten sich an unterschiedliche Adressaten und deren spezifische Sicht auf die Semantik...

Insbesondere unterscheiden wir den...

- *denotationellen*
- *operationellen*
- *axiomatischen*

Stil.

Semantikdefinitionsstile (2)

- *Sprachentwicklersicht*
  - Denotationelle Semantik
- *Sprach- und Anwendungsimplementierersicht*
  - Operationelle Semantik
    - Strukturell operationelle Semantik (small steps semantics)
    - Natürliche Semantik (big steps semantics)
- *Programmierer- und Verifizierersicht*
  - Axiomatische Semantik

Strukturell operationelle Semantik (1)

...noch einmal wiederholt für das Beispiel von WHILE:

$[\text{skip}_{sos}]$ $\dfrac{}{\langle skip,\sigma\rangle \Rightarrow \sigma}$

$[\text{abort}_{sos}]$ $\dfrac{}{\langle abort,\sigma\rangle \Rightarrow error}$

$[\text{ass}_{sos}]$ $\dfrac{}{\langle x{:=}t,\sigma\rangle \Rightarrow \sigma[[\![\, t \,]\!]_A(\sigma)/x]}$

$[\text{comp}^1_{sos}]$ $\dfrac{\langle \pi_1,\sigma\rangle \Rightarrow \langle \pi_1',\sigma'\rangle}{\langle \pi_1;\pi_2,\sigma\rangle \Rightarrow \langle \pi_1';\pi_2,\sigma'\rangle}$

$[\text{comp}^2_{sos}]$ $\dfrac{\langle \pi_1,\sigma\rangle \Rightarrow \sigma'}{\langle \pi_1;\pi_2,\sigma\rangle \Rightarrow \langle \pi_2,\sigma'\rangle}$

Strukturell operationelle Semantik (2)

$[\text{if}^{tt}_{sos}]$ $\dfrac{}{\langle \text{if } b \text{ then } \pi_1 \text{ else } \pi_2 \text{ fi},\sigma\rangle \Rightarrow \langle \pi_1,\sigma\rangle}$ $[\![\, b \,]\!]_B(\sigma) = \texttt{tt}$

$[\text{if}^{ff}_{sos}]$ $\dfrac{}{\langle \text{if } b \text{ then } \pi_1 \text{ else } \pi_2 \text{ fi},\sigma\rangle \Rightarrow \langle \pi_2,\sigma\rangle}$ $[\![\, b \,]\!]_B(\sigma) = \texttt{ff}$

$[\text{while}_{sos}]$ $\dfrac{}{\langle \text{while } b \text{ do } \pi \text{ od},\sigma\rangle \Rightarrow \langle \text{if } b \text{ then } \pi; \text{ while } b \text{ do } \pi \text{ od else } skip \text{ fi},\sigma\rangle}$

Strukturell operationelle Semantik (3)

Der Fokus liegt auf...

- *individuellen Schritten* einer Berechnungsfolge, d.h. auf der Ausführung von Zuweisungen und Tests

  Intuitive Bedeutung der Transitionsrelation...

  $$\langle \pi, \sigma \rangle \Rightarrow \gamma$$

  ...mit γ von der Form $\langle \pi', \sigma' \rangle$ oder σ' oder $error$ beschreibt den *ersten* Schritt der Berechnungsfolge von π angesetzt auf σ. Folgende Übergänge sind möglich:

  - γ von der Form $\langle \pi', \sigma' \rangle$:
    Abarbeitung von π nicht vollständig; das Restprogramm π' ist auf σ' anzusetzen
  - γ von der Form σ':
    Abarbeitung von π vollständig; π angesetzt auf σ terminiert in einem Schritt in σ'
  - γ von der Form $error$:
    Abarbeitung von π terminiert irregulär

Natürliche Semantik (1)

...ebenfalls für das Beispiel von WHILE:

$[\text{skip}_{ns}]$ $\quad \frac{}{\langle skip, \sigma \rangle \to \sigma}$

$[\text{abort}_{ns}]$ $\quad \frac{}{\langle abort, \sigma \rangle \to error}$

$[\text{ass}_{ns}]$ $\quad \frac{}{\langle x := t, \sigma \rangle \to \sigma[[\![t]\!]_A(\sigma)/x]}$

$[\text{comp}_{ns}]$ $\quad \frac{\langle \pi_1, \sigma \rangle \to \sigma', \langle \pi_2, \sigma' \rangle \to \sigma''}{\langle \pi_1 ; \pi_2, \sigma \rangle \to \sigma''}$

Natürliche Semantik (2)

$[\text{if}^{tt}_{ns}]$ $\quad \frac{\langle \pi_1, \sigma \rangle \to \sigma'}{\langle \text{if } b \text{ then } \pi_1 \text{ else } \pi_2 \text{ fi}, \sigma \rangle \to \sigma'} \quad [\![b]\!]_B(\sigma) = \texttt{tt}$

$[\text{if}^{ff}_{ns}]$ $\quad \frac{\langle \pi_2, \sigma \rangle \to \sigma'}{\langle \text{if } b \text{ then } \pi_1 \text{ else } \pi_2 \text{ fi}, \sigma \rangle \to \sigma'} \quad [\![b]\!]_B(\sigma) = \texttt{ff}$

$[\text{while}^{tt}_{ns}]$ $\quad \frac{\langle \pi, \sigma \rangle \to \sigma', \langle \text{while } b \text{ do } \pi \text{ od}, \sigma' \rangle \to \sigma''}{\langle \text{while } b \text{ do } \pi \text{ od}, \sigma \rangle \to \sigma''} \quad [\![b]\!]_B(\sigma) = \texttt{tt}$

$[\text{while}^{ff}_{ns}]$ $\quad \frac{}{\langle \text{while } b \text{ do } \pi \text{ od}, \sigma \rangle \to \sigma} \quad [\![b]\!]_B(\sigma) = \texttt{ff}$

Natürliche Semantik (3)

Der Fokus liegt auf...

- Zusammenhang von *initialem* und *finalem* Zustand einer Berechnungsfolge

  Intuitive Bedeutung von...

  $$\langle \pi, \sigma \rangle \to \gamma$$

  ...mit γ von der Form σ' oder $error$ ist: π angesetzt auf initialen Zustand σ terminiert schließlich im finalen Zustand σ' bzw. terminiert irregulär.

Determinismus der NS-Regeln

Lemma 2.1

$\forall\, \pi \in \mathbf{Prg},\ \sigma \in \Sigma,\ \gamma, \gamma' \in \Gamma.\ \langle \pi, \sigma \rangle \to \gamma\ \wedge \langle \pi, \sigma \rangle \to \gamma' \quad \Rightarrow \quad \gamma = \gamma'$

Korollar 2.2
Die von den NS-Regeln für eine Konfiguration induzierte finale Konfiguration ist (sofern definiert) eindeutig bestimmt, d.h. *deterministisch*.

Salopper, wenn auch weniger präzise:
Die (N-) Semantik von WHILE ist deterministisch!

Das Semantikfunktional $[\![\ ]\!]_{ns}$

Korollar 2.2 erlaubt uns festzulegen:

- Die natürliche Semantik von WHILE ist gegeben durch das Funktional

  $$[\![\ .\ ]\!]_{ns} : \mathbf{Prg} \to (\Sigma \to \Sigma_\varepsilon)$$

  welches definiert wird durch:

  $$\forall\, \pi \in \mathbf{Prg},\ \sigma \in \Sigma.\ [\![\ \pi\ ]\!]_{ns}(\sigma) =_{df} \begin{cases} \sigma' & \text{falls } \langle \pi, \sigma \rangle \to \sigma' \\ error & \text{falls } \langle \pi, \sigma \rangle \to error \\ undef & \text{sonst} \end{cases}$$

Variante induktiver Beweisführung

Induktion über die Form von Ableitungsbäumen:

- *Induktionsanfang*
  - Beweise, dass A für die Axiome des Transitionssystems gilt (und somit für alle nichtzusammengesetzte Ableitungsbäume).
- *Induktionsschritt*
  - Beweise für jede echte Regel des Transitionssystems unter der Annahme, dass A für jede Prämisse dieser Regel gilt (*Induktionshypothese!*), A auch für die Konklusion dieser Regel gilt, sofern die (ggf. vorhandenen) Randbedingungen der Regel erfüllt sind.

Anwendung

- Induktive Beweisführung über die Form von Ableitungsbäumen ist typisch zum Nachweis von Aussagen über Eigenschaften natürlicher Semantik.

Ein Beispiel dafür ist der Beweis von **Lemma 2.1**!

Denotationelle Semantik (1)

...auch für das Beispiel von WHILE:

$[\![skip]\!]_{ds} = Id$

$[\![abort]\!]_{ds} = Error$

$[\![x := t]\!]_{ds}(\sigma) = \sigma[[\![t]\!]_A(\sigma)/x]$

$[\![\pi_1 ;\ \pi_2]\!]_{ds} = [\![\pi_2]\!]_{ds} \circ [\![\pi_1]\!]_{ds}$

$[\![$ if b then π_1 else π_2 fi $]\!]_{ds} = cond([\![b]\!]_{\mathcal{B}}, [\![\pi_1]\!]_{ds}, [\![\pi_2]\!]_{ds})$

$[\![$ while b do π od $]\!]_{ds} = FIX\ F$

where $F\ g = cond([\![b]\!]_{\mathcal{B}}, g \circ [\![\pi]\!]_{ds}, Id)$

Denotationelle Semantik (2)

Es bezeichnen:

- $Id : \Sigma_\varepsilon \to \Sigma_\varepsilon$ die identische Zustandstransformation:

$$\forall\, \sigma \in \Sigma_\varepsilon.\ Id(\sigma) =_{df} \sigma$$

- $Error : \Sigma_\varepsilon \to \Sigma_\varepsilon$ die konstante Zustandstransformation mit:

$$\forall\, \sigma \in \Sigma_\varepsilon.\ Error(\sigma) =_{df} error$$

Denotationelle Semantik (3)

Zur Hilfsfunktion $cond$...

Funktionalität...

$$cond : (\Sigma \to \mathbb{B}) \times (\Sigma \to \Sigma) \times (\Sigma \to \Sigma) \to (\Sigma \to \Sigma)$$

Definiert durch...

$$cond(p, g_1, g_2)\ \sigma =_{df} \begin{cases} g_1\ \sigma & \text{falls } p\ \sigma = \mathtt{tt} \\ g_2\ \sigma & \text{falls } p\ \sigma = \mathtt{ff} \end{cases}$$

Zu den Argumenten und zum Resultat von $cond$...

- 1. Argument: Prädikat (in unserem Szenario total definiert; siehe Vorlesungsteil 1)
- 2.&3. Argument: Je eine partiell definierte Zustandstransformation
- Resultat: Wieder eine partiell definierte Zustandstransformation

Denotationelle Semantik (4)

Zur Hilfsfunktion FIX...

Funktionalität...

$$FIX : ((\Sigma \to \Sigma) \to (\Sigma \to \Sigma)) \to (\Sigma \to \Sigma)$$

Definiert durch...

$$F\ g = cond([\![b]\!]_{\mathcal{B}}, g \circ [\![\pi]\!]_{ds}, Id)$$

Daraus ergibt sich...

- FIX ist ein Funktional ("Zustandstransformationsfunktional")
- Die denotationelle Semantik der while-Schleife ist ein Fixpunkt des Funktionals F (und zwar der kleinste!)

Mehr Details zu FIX und Co. später!

Denotationelle Semantik (5)

- *Operationelle* Semantik
  ...der Fokus liegt darauf, *wie* ein Programm ausgeführt wird
- *Denotationelle* Semantik
  ...der Fokus liegt auf dem *Effekt*, den die Ausführung eines Programms hat: Für jedes *syntaktische* Konstrukt gibt es eine *semantische* Funktion, die ersterem ein *mathematisches Objekt* zuweist, i.a. eine Funktion, die den Effekt der Ausführung des Konstrukts beschreibt (jedoch nicht, wie dieser Effekt erreicht wird).

Denotationelle Semantik (6)

Zentral für denotationelle Semantiken: **Kompositionalität**!

Intuitiv:

- Für jedes Element der elementaren syntaktischen Konstrukte/Kategorien gibt es eine zugehörige semantische Funktion
- Für jedes Element eines zusammengesetzten syntaktischen Konstrukts/Kategorie gibt es eine semantische Funktion, die über die semantischen Funktionen der Komponenten des zusammengesetzten Konstrukts definiert ist.

Denotationelle Semantik (7)

Lemma 2.2
Für alle $\pi \in$ **Prg** ist durch die Gleichungen von Folie "Denotationelle Semantik (1)" eine (partielle) Funktion $[\![\pi]\!]_{ds}$ definiert, die denotationelle Semantik von π.

Hauptergebnisse

Theorem

$$\forall\ \pi\ \in\ \textbf{Prg}.\ [\![\pi]\!]_{sos} = [\![\pi]\!]_{ns} = [\![\pi]\!]_{ds}$$

Die Äquivalenz der strukturell operationellen, natürlichen und denotationellen Semantik von WHILE legt es nahe, den semantikangebenden Index in der Folge fortzulassen und vereinfachend von $[\![\ ]\!]$ als der Semantik der Sprache WHILE zu sprechen:

$$[\![\ ]\!] : \textbf{Prg} \to (\Sigma \to \Sigma_\varepsilon)$$

definiert durch

$$[\![\ ]\!] =_{df} [\![\ ]\!]_{sos}$$

WHILE – Denotationelle Semantik (1)

- **Prg** ...bezeichne die Menge aller Programme der Sprache **WHILE**

Denotationelle Semantik

$$[\![\ ]\!]_{ds} : \mathbf{Prg} \to (\Sigma \to \Sigma_\varepsilon)$$

Somit...

- Die *denotationelle Semantik* eines **WHILE**-Programms ist eine (partiell definierte) *Zustandstransformation*, wobei die Menge der *Zustände* gegeben ist durch

$$\Sigma =_{df} \{\sigma \mid \sigma : V \to D\}$$

Beachte...

- Auch die operationelle (die strukturell operationelle wie auch die natürliche) Semantik eines **WHILE**-Programms ist eine (partiell definierte) *Zustandstransformation* auf Σ, nicht aber die axiomatische Semantik.

WHILE – Denotationelle Semantik (2)

Erinnerung:

$[\![skip]\!]_{ds} = Id$

$[\![abort]\!]_{ds} = Error$

$[\![x := t]\!]_{ds}(\sigma) = \sigma[[\![t]\!]_A(\sigma)/x]$

$[\![\pi_1;\ \pi_2]\!]_{ds} = [\![\pi_2]\!]_{ds} \ \circ\ [\![\pi_1]\!]_{ds}$

$[\![\textbf{if } b \textbf{ then } \pi_1 \textbf{ else } \pi_2 \textbf{ fi}]\!]_{ds} = cond([\![b]\!]_{\mathcal{B}}, [\![\pi_1]\!]_{ds}, [\![\pi_2]\!]_{ds})$

$[\![\textbf{while } b \textbf{ do } \pi \textbf{ od}]\!]_{ds} = FIX\ F$

where $F\ g = cond([\![b]\!]_{\mathcal{B}}, g \circ [\![\pi]\!]_{ds}, Id)$

WHILE – Denotationelle Semantik (3)

Noch offen...

- Die Bedeutung von...
  - $cond$ und
  - $FIX\ F$

Diese Bedeutung wollen wir in der Folge aufklären...

Zur Bedeutung von cond

Hilfsfunktion $cond$...

Funktionalität...

$$cond : (\Sigma \to \mathbb{B}) \times (\Sigma \to \Sigma) \times (\Sigma \to \Sigma) \to (\Sigma \to \Sigma)$$

Definiert durch...

$$cond(p, g_1, g_2)\ \sigma =_{df} \begin{cases} g_1\ \sigma & \text{if } p\ \sigma = \texttt{tt} \\ g_2\ \sigma & \text{if } p\ \sigma = \texttt{ff} \end{cases}$$

Zu den Argumenten und zum Resultat von $cond$...

- 1. Argument: Prädikat (in unserem Szenario total definiert; siehe Vorlesungsteil 1)
- 2.&3. Argument: Je eine partiell definierte Zustandstransformation
- Resultat: Wieder eine partiell definierte Zustandstransformation

Damit erhalten wir

...für die Bedeutung der Fallunterscheidung

$$[\![\textbf{if } b \textbf{ then } \pi_1 \textbf{ else } \pi_2 \textbf{ fi}]\!]_{ds}\ \sigma$$

$$= cond([\![b]\!]_{\mathcal{B}}, [\![\pi_1]\!]_{ds}, [\![\pi_2]\!]_{ds})\ \sigma$$

$$= \begin{cases} \sigma' & \text{falls } ([\![b]\!]_{\mathcal{B}}\ \sigma = \texttt{tt} \ \wedge\ [\![\pi_1]\!]_{\texttt{ds}}\ \sigma = \sigma') \\ & \quad \vee\ ([\![b]\!]_{\mathcal{B}}\ \sigma = \texttt{ff} \ \wedge\ [\![\pi_2]\!]_{\texttt{ds}}\ \sigma = \sigma') \\ error & \text{falls } ([\![b]\!]_{\mathcal{B}}\ \sigma = \texttt{tt} \ \wedge\ [\![\pi_1]\!]_{\texttt{ds}}\ \sigma = \texttt{error}) \\ & \quad \vee\ ([\![b]\!]_{\mathcal{B}}\ \sigma = \texttt{ff} \ \wedge\ [\![\pi_2]\!]_{\texttt{ds}}\ \sigma = \texttt{error}) \\ undef & \text{falls } ([\![b]\!]_{\mathcal{B}}\ \sigma = \texttt{tt} \ \wedge\ [\![\pi_1]\!]_{\texttt{ds}}\ \sigma = \texttt{undef}) \\ & \quad \vee\ ([\![b]\!]_{\mathcal{B}}\ \sigma = \texttt{ff} \ \wedge\ [\![\pi_2]\!]_{\texttt{ds}}\ \sigma = \texttt{undef}) \end{cases}$$

Erinnerung:

- $[\![b]\!]_{\mathcal{B}}$ ist in unserem Szenario total definiert; $[\![b]\!]_{\mathcal{B}}\ \sigma$ ist daher stets von $undef$ verschieden.

Zur Bedeutung von FIX F

Funktionalität...

$$FIX : ((\Sigma \to \Sigma) \to (\Sigma \to \Sigma)) \to (\Sigma \to \Sigma)$$

Definiert durch...

$$F\ g = cond([\![b]\!]_{\mathcal{B}}, g \circ [\![\pi]\!]_{ds}, Id)$$

Daraus ergibt sich...

- FIX ist ein Funktional ("Zustandstransformationsfunktional")
- Die denotationelle Semantik der while-Schleife ist ein Fixpunkt des Funktionals F (und zwar der kleinste!)

Schrittweise zur denotationellen Semantik der while-Schleife

Dazu folgende Beobachtung...

- $\textbf{while } b \textbf{ do } \pi \textbf{ od}$ muss dieselbe Bedeutung haben wie...
  $\textbf{if } b \textbf{ then } (\pi;\ \textbf{while } b \textbf{ do } \pi \textbf{ od}) \textbf{ else } skip \textbf{ fi}$

Daraus folgt...

- $[\![\textbf{while } b \textbf{ do } \pi \textbf{ od}]\!]_{ds} = cond([\![b]\!]_{\mathcal{B}}, [\![\textbf{while } b \textbf{ do } \pi \textbf{ od}]\!]_{ds} \circ [\![\pi]\!]_{ds}, Id)$

Und daraus schließlich...

- $[\![\textbf{while } b \textbf{ do } \pi \textbf{ od}]\!]_{ds}$ muss Fixpunkt des Funktionals F sein, dass definiert ist durch

  $$F\ g = cond([\![b]\!]_{\mathcal{B}}, g \circ [\![\pi]\!]_{ds}, Id)$$

  Oder anders ausgedrückt, es muss gelten:

  $$[\![\textbf{while } b \textbf{ do } \pi \textbf{ od}]\!]_{ds} = F([\![\textbf{while } b \textbf{ do } \pi \textbf{ od}]\!]_{ds})$$

  ...was uns wie gewünscht zu einer *kompositionellen* Definition von $[\![\textbf{while } b \textbf{ do } \pi \textbf{ od}]\!]_{ds}$ und damit von $[\![\ \ ]\!]_{ds}$ insgesamt führen wird.

Etwas formaler: Unser Arbeitsplan

Erforderlich...

- Einige Resultate aus der *Fixpunkttheorie*

Beim nächsten Mal wird nachgeholt...

- Der mathematische Hintergrund (Ordnungen, CPOs, Stetigkeit von Funktionen) und die benötigten Resultate (Fixpunktsatz)
  $\rightsquigarrow$ ...siehe Vorlesungsteil 6 vom 28.11.2006

Somit bleibt an dieser Stelle zu tun...

- Nachzuweisen, dass diese Resultate auf unsere Situation anwendbar sind.

Folgende drei Argumente...

...werden dafür entscheidend sein

1. $[\Sigma \to \Sigma]$ kann vollständig partiell geordnet werden.
2. F im Anwendungskontext ist stetig
3. Fixpunktbildung im Anwendungskontext wird ausschließlich auf stetige Funktionen angewendet.

Insgesamt ergibt sich dann daraus die Wohldefiniertheit von

$$[\![\ ]\!]_{ds} : \mathbf{Prg} \to (\Sigma \to \Sigma_\varepsilon)$$

Ordnung auf Zustandstransformationen

Bezeichne...

- $[\Sigma \to \Sigma]$ die Menge der partiell definierten Zustandstransformationen.

Wir definieren...

$$g_1 \sqsubseteq g_2 \iff \forall\, \sigma \in \Sigma.\ g_1\ \sigma\ \mathit{definiert} = \sigma' \Rightarrow g_2\ \sigma\ \mathit{definiert} = \sigma'$$
$$\text{mit } g_1, g_2 \in [\Sigma \to \Sigma_\varepsilon]$$

Lemma 1

1. $([\Sigma \to \Sigma], \sqsubseteq)$ ist eine partielle Ordnung.
2. Die *total undefinierte* (d.h. nirgends definierte) Funktion $\bot : \Sigma \to \Sigma$ mit $\bot\ \sigma = \mathit{undef}$ für alle $\sigma \in \Sigma$ ist *kleinstes* Element in $([\Sigma \to \Sigma], \sqsubseteq)$

Ordnung auf Zustandstransformationen

Sogar...

Lemma 2
Das Paar $([\Sigma \to \Sigma], \sqsubseteq)$ ist eine vollständige partielle Ordnung (CPO) mit kleinstem Element $\bot$.

Weiter gilt: Die kleinste obere Schranke $\bigsqcup Y$ einer Kette Y ist gegeben durch

$$\mathit{graph}(\bigsqcup Y) = \bigcup \{\mathit{graph}(g) \mid g \in Y\}$$

Das heißt: $(\bigsqcup Y)\,\sigma = \sigma' \iff \exists\, g \in Y.\ g\ \sigma = \sigma'$

Einschub: Graph einer Funktion

Der *Graph* einer totalen Funktion $f : M \to N$ ist definiert durch

$$\mathit{graph}(f) =_{df} \{\langle m, n\rangle \in M \times N \mid f\ m = n\}$$

Es gilt:

- $\langle m, n\rangle \in \mathit{graph}(f)\ \wedge\ \langle m, n'\rangle \in \mathit{graph}(f) \Rightarrow n = n'$ (*rechtseindeutig*)
- $\forall\, m \in M.\ \exists\, n \in N.\ \langle m, n\rangle \in \mathit{graph}(f)$ (*linkstotal*)

Der *Graph* einer partiellen Funktion $f : M \to N$ mit Definitionsbereich $M_f \subseteq M$ ist definiert durch

$$\mathit{graph}(f) =_{df} \{\langle m, n\rangle \in M \times N \mid f\ m = n \wedge m \in M_f\}$$

Vereinbarung...
Für $f : M \to N$ partiell definierte Funktion auf $M_f \subseteq M$ schreiben wir

- $f\ m = n$, falls $\langle m, n\rangle \in \mathit{graph}(f)$
- $f\ m = \mathit{undef}$, falls $m \notin M_f$

Stetigkeitkeitsresultate (1)

Lemma 3
Sei $g_0 \in [\Sigma \to \Sigma]$, sei $p \in [\Sigma \to \mathbb{B}]$ und sei F definiert durch

$$F\ g = cond(p, g, g_0)$$

Dann gilt: F ist stetig.

Zur Erinnerung: Seien $(C, \sqsubseteq_C)$ und $(D, \sqsubseteq_D)$ zwei CPOs und sei $f : C \to D$ eine Funktion von C nach D.
Dann heißt f...

- *monoton* gdw. $\forall c, c' \in C.\ c \sqsubseteq_C c' \Rightarrow f(c) \sqsubseteq_D f(c')$
  (*Erhalt der Ordnung der Elemente*)
- *stetig* gdw. $\forall C' \subseteq C.\ f(\bigsqcup_C C') =_D \bigsqcup_D f(C')$
  (*Erhalt der kleinsten oberen Schranken*)

Stetigkeitkeitsresultate (2)

Lemma 4
Sei $g_0 \in [\Sigma \to \Sigma]$ und sei F definiert durch

$$F\ g = g \circ g_0$$

Dann gilt: F ist stetig.

Zusammen mit...

Lemma 5
Die Gleichungen zur Festlegung der denotationellen Semantik von WHILE (vgl. Folie 13 von heute) definieren eine totale Funktion

$$[\![\ ]\!]_{ds} \in [\mathbf{Prg} \to (\Sigma \to \Sigma_\varepsilon)]$$

...sind wir durch! Wir können beweisen:

$$[\![\ ]\!]_{ds} : \mathbf{Prg} \to (\Sigma \to \Sigma_\varepsilon)$$

ist wohldefiniert!

Und somit wie anfangs angedeutet...

Aus...

1. Die Menge $[\Sigma \to \Sigma]$ der partiell definierten Zustandstransformationen bildet zusammen mit der Ordnung $\sqsubseteq$ eine CPO.
2. Funktional F mit "$F\ g = cond(p, g, g_0)$" und "$g \circ g_0$" ist stetig
3. In der Definition von $[\![\ ]\!]_{ds}$ wird die Fixpunktbildung ausschließlich auf stetige Funktionen angewendet.

...ergibt sich wie gewünscht:

$$[\![\ ]\!]_{ds} : \mathbf{Prg} \to (\Sigma \to \Sigma_\varepsilon)$$

...ist wohldefiniert!

Vorschau auf die nächsten Vorlesungstermine...

- Di, 28.11.2006, Vorlesung von 17:45 Uhr bis 19:15 Uhr, Bibliothek E185/1
- Di, 05.12.2006, Vorlesung von 17:45 Uhr bis 19:15 Uhr, Bibliothek E185/1
- Di, 12.12.2006, Vorlesung von 17:45 Uhr bis 19:15 Uhr, Bibliothek E185/1
- *Di, 19.12.2006: Keine Vorlesung!* (*Ferialzeit*)
- *Di, 26.12.2006: Keine Vorlesung!* (*Ferialzeit*)

Vorschau auf die weiteren Vorlesungstermine...

- *Di, 02.01.2007: Keine Vorlesung!* (*Ferialzeit*)
- Di, 09.01.2007, Vorlesung von 17:45 Uhr bis 19:15 Uhr, Bibliothek E185/1
- Di, 16.01.2007, Vorlesung von 17:45 Uhr bis 19:15 Uhr, Bibliothek E185/1
- Di, 23.01.2007, Vorlesung von 17:45 Uhr bis 19:15 Uhr, Bibliothek E185/1
- Di, 30.01.2007, Vorlesung von 17:45 Uhr bis 19:15 Uhr, Bibliothek E185/1