Denotatsioonsemantika

Kristo Heero

ATI

2004

Denotatsioonsemantika (1)

- Meid huvitab ainult programmi täitmise efekt, st. assotsiatsioon alg- ja lõppoleku vahel.
- Defineeritakse semantiline funktsioon iga süntaktilise kategooria jaoks, st. iga süntaktiline konstruktsioon seotakse matemaatilise objektiga (funktsioon, mis kirjeldab konstruktsiooni täitmise efekti).

Denotatsioonsemantika (1)

- Semantilised funktsioonid on denotatsioon-semantikas kompositsioonilised, st
  - iga süntaktilise kategooria baaselemendi jaoks on semantiline klausel
  - iga komponeeritud süntaktilise kategooria elemendi jaoks vastab kompositsiooni-elementide vahetute klauslite kompositsioon.

Otsestiili semantika

- Avaldise S täitmise efekt tähendab oleku muutmist.

- Defineerime semantilise funktsiooni

  $S_{ds} : Stm \rightarrow (State \hookrightarrow State)$

While keele denotatsioonsemantika

$$\mathcal{S}_{\mathrm{ds}}[\![x := a]\!]s = s[x \mapsto \mathcal{A}[\![a]\!]s]$$

$$\mathcal{S}_{\mathrm{ds}}[\![\mathtt{skip}]\!] = \mathrm{id}$$

$$\mathcal{S}_{\mathrm{ds}}[\![S_1 \,;\, S_2]\!] = \mathcal{S}_{\mathrm{ds}}[\![S_2]\!] \circ \mathcal{S}_{\mathrm{ds}}[\![S_1]\!]$$

$$\mathcal{S}_{\mathrm{ds}}[\![\mathtt{if}\ b\ \mathtt{then}\ S_1\ \mathtt{else}\ S_2]\!] = \mathrm{cond}(\mathcal{B}[\![b]\!], \mathcal{S}_{\mathrm{ds}}[\![S_1]\!], \mathcal{S}_{\mathrm{ds}}[\![S_2]\!])$$

$$\mathcal{S}_{\mathrm{ds}}[\![\mathtt{while}\ b\ \mathtt{do}\ S]\!] = \mathrm{FIX}\ F$$

$$\text{where } F\ g = \mathrm{cond}(\mathcal{B}[\![b]\!], g \circ \mathcal{S}_{\mathrm{ds}}[\![S]\!], \mathrm{id})$$

$S_1;S_2$

$$
\begin{aligned}
&\mathcal{S}_{\mathrm{ds}}[\![S_1 \; ; \; S_2]\!]s \\
&\quad = \quad (\mathcal{S}_{\mathrm{ds}}[\![S_2]\!] \circ \mathcal{S}_{\mathrm{ds}}[\![S_1]\!])s \\
&\quad = \quad \begin{cases} s'' & \text{if there exists } s' \text{ such that } \mathcal{S}_{\mathrm{ds}}[\![S_1]\!]s = s' \\ & \text{and } \mathcal{S}_{\mathrm{ds}}[\![S_2]\!]s' = s'' \\ \underline{\text{undef}} & \text{if } \mathcal{S}_{\mathrm{ds}}[\![S_1]\!]s = \underline{\text{undef}} \\ & \text{or if there exists } s' \text{ such that } \mathcal{S}_{\mathrm{ds}}[\![S_1]\!]s = s' \\ & \text{but } \mathcal{S}_{\mathrm{ds}}[\![S_2]\!]s' = \underline{\text{undef}} \end{cases}
\end{aligned}
$$

if then else (1)

$$\mathcal{S}_{\mathrm{ds}}[\![\texttt{if } b \texttt{ then } S_1 \texttt{ else } S_2]\!] = \mathrm{cond}(\mathcal{B}[\![b]\!], \mathcal{S}_{\mathrm{ds}}[\![S_1]\!], \mathcal{S}_{\mathrm{ds}}[\![S_2]\!])$$

$$\begin{aligned}\mathrm{cond}:\ & (\mathbf{State} \to \mathbf{T}) \times (\mathbf{State} \hookrightarrow \mathbf{State}) \times (\mathbf{State} \hookrightarrow \mathbf{State}) \\ & \to (\mathbf{State} \hookrightarrow \mathbf{State})\end{aligned}$$

$$\mathrm{cond}(p,\ g_1,\ g_2)\ s = \begin{cases} g_1\ s & \text{if } p\ s = \mathbf{tt} \\ g_2\ s & \text{if } p\ s = \mathbf{ff} \end{cases}$$

if then else (2)

$$\mathcal{S}_{\mathrm{ds}}[\![\texttt{if } b \texttt{ then } S_1 \texttt{ else } S_2]\!]\ s$$

$$= \quad \mathrm{cond}(\mathcal{B}[\![b]\!], \mathcal{S}_{\mathrm{ds}}[\![S_1]\!], \mathcal{S}_{\mathrm{ds}}[\![S_2]\!])\ s$$

$$= \begin{cases} s' & \text{if } \mathcal{B}[\![b]\!]s = \mathbf{tt} \text{ and } \mathcal{S}_{\mathrm{ds}}[\![S_1]\!]s = s' \\ & \text{or if } \mathcal{B}[\![b]\!]s = \mathbf{ff} \text{ and } \mathcal{S}_{\mathrm{ds}}[\![S_2]\!]s = s' \\ \underline{\mathrm{undef}} & \text{if } \mathcal{B}[\![b]\!]s = \mathbf{tt} \text{ and } \mathcal{S}_{\mathrm{ds}}[\![S_1]\!]s = \underline{\mathrm{undef}} \\ & \text{or if } \mathcal{B}[\![b]\!]s = \mathbf{ff} \text{ and } \mathcal{S}_{\mathrm{ds}}[\![S_2]\!]s = \underline{\mathrm{undef}} \end{cases}$$

while do

- while b do S võib esitada avaldisega
  if b then (S; while b do S) else skip
- Saame

$$\mathcal{S}_{\mathrm{ds}}[\![\texttt{while } b \texttt{ do } S]\!] = \mathrm{cond}(\mathcal{B}[\![b]\!], \mathcal{S}_{\mathrm{ds}}[\![\texttt{while } b \texttt{ do } S]\!] \circ \mathcal{S}_{\mathrm{ds}}[\![S]\!], \mathrm{id})$$

$\mathcal{S}_{\mathrm{ds}}[\![\texttt{while } b \texttt{ do } S]\!]$ must be a *fixed point* of the functional F defined by

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

- St. $\mathcal{S}_{\mathrm{ds}}[\![\texttt{while } b \texttt{ do } S]\!] = F\ (\mathcal{S}_{\mathrm{ds}}[\![\texttt{while } b \texttt{ do } S]\!])$

Näide: while ¬(x = 0) do skip

$$(F'\ g)\ s = \begin{cases} g\ s & \text{if } s\ \mathrm{x} \neq 0 \\ s & \text{if } s\ \mathrm{x} = 0 \end{cases}$$

$$g_1\ s = \begin{cases} \underline{\text{undef}} & \text{if } s\ \mathrm{x} \neq 0 \\ s & \text{if } s\ \mathrm{x} = 0 \end{cases}$$

$g_2\ s = \underline{\text{undef}}$ for all s

$$(F'\ g_1)\ s = \begin{cases} g_1\ s & \text{if } s\ \mathrm{x} \neq 0 \\ s & \text{if } s\ \mathrm{x} = 0 \end{cases}$$

$$= \begin{cases} \underline{\text{undef}} & \text{if } s\ \mathrm{x} \neq 0 \\ s & \text{if } s\ \mathrm{x} = 0 \end{cases}$$

$$= g_1\ s$$

while do (2)

- Kahjuks sellisest while do definitsioonist ei piisa, sest
  - Leidub funktsionaale, millel on enam kui üks püsipunkt
  - Leidub funktsionaale, millel pole üldse püsipunkti

Näited funktsionaalidest

- Eelmises näites oleval funktsionaalil on iga funktsioon g’s = s, kui s x = 0.
- Näiteks järgmisel funktsionaalil puudub püsipunkt :

$$F_1\ g = \begin{cases} g_1 & \text{if } g = g_2 \\ g_2 & \text{otherwise} \end{cases}$$

Nõudmised püsipunktile (1)

- Kehtestame nõudmised püsipunktile ja näitame, et neile tingimustele vastavaid püsipunkte on ülimalt üks.
- Ja näitame, et kõik While keele avaldistest pärinevad funktsionaalid omavad neile tingimustele vastavat püsipunkti.

Nõudmised püsipunktile (2)

- Vaatame avaldise while b do S täitmist olekus s_0.
- On kolm võimalust:

  A: termineerub

  B: lokaalses lõpmatus tsüklis (S'is on tsükkel)

  C: globaalses lõpmatus tsüklis
- Uurime, mida saab öelda funktsionaali F ja selle püsipunkti kohta igal antud võimalusel.

A

- while lause termineerumine tähendab, et eksisteerivad olekud s_1, ..., s_n, et

$$\mathcal{B}[\![b]\!]\ s_i = \begin{cases} \mathbf{tt} & \text{if } i<n \\ \mathbf{ff} & \text{if } i=n \end{cases}$$

and

$$\mathcal{S}_{ds}[\![S]\!]\ s_i = s_{i+1} \quad \text{for } i<n$$

- Näiteks avaldis:

  while $0 \leq x$ do $x := x - 1$, kus $x \geq 0$

A

- Olgu g_0 funktsionaali F suvaline püsipunkt, st $F\ g_0 = g_0$.
- Kui i < n, siis

$$\begin{aligned} g_0\ s_i &= (F\ g_0)\ s_i \\ &= \mathrm{cond}(\mathcal{B}[\![b]\!],\ g_0 \circ \mathcal{S}_{ds}[\![S]\!],\ \mathrm{id})\ s_i \\ &= g_0\ (\mathcal{S}_{ds}[\![S]\!]\ s_i) \\ &= g_0\ s_{i+1} \end{aligned}$$

- Kui i = n, siis

$$\begin{aligned} g_0\ s_n &= (F\ g_0)\ s_n \\ &= \mathrm{cond}(\mathcal{B}[\![b]\!],\ g_0 \circ \mathcal{S}_{ds}[\![S]\!],\ \mathrm{id})\ s_n \\ &= \mathrm{id}\ s_n \\ &= s_n \end{aligned}$$

- Seega $g_0\ s_0 = s_n$

B

- while b do S olekus s_0 lokaalne lõpmatu tsükkel tähendab, et eksisteerivad olekud s_1, ..., s_n, et

$$\mathcal{B}[\![b]\!]s_i = \mathbf{tt} \text{ for } i \leq n$$

and

$$\mathcal{S}_{ds}[\![S]\!]s_i = \begin{cases} s_{i+1} & \text{for } i<n \\ \underline{\text{undef}} & \text{for } i=n \end{cases}$$

- Näiteks avaldis:

```
while 0≤x do (if x=0 then (while true do skip)
                      else x := x−1)
```

B

- Analoogselt eelmise näitega saame, et
  - kui i < n, siis $g_0\ s_i = g_0\ s_{i+1}$
  - kui i = n, siis

$$\begin{aligned} g_0\ s_n &= (F\ g_0)\ s_n \\ &= \mathrm{cond}(\mathcal{B}[\![b]\!],\ g_0 \circ \mathcal{S}_{ds}[\![S]\!],\ \mathrm{id})\ s_n \\ &= (g_0 \circ \mathcal{S}_{ds}[\![S]\!])\ s_n \\ &= \underline{\mathrm{undef}} \end{aligned}$$

- Seega $g_0\ s_0$ = undef

C

- while b do S olekus s_0 globaalne lõpmatu tsükkel tähendab, et eksisteerivad olekud s_1, ... nii, et

$$\mathcal{B}[\![b]\!]s_i = \mathbf{tt} \text{ for all i}$$

and

$$\mathcal{S}_{ds}[\![S]\!]s_i = s_{i+1} \text{ for all i.}$$

- Näiteks avaldis

  while ¬(x = 0) do skip

C

- Analoogselt eelmise näitega saame

  $g_0\ s_i = g_0\ s_{i+1}$ iga $i \geq 0$ korral.
- Seega saame

  $g_0\ s_0 = g_0\ s_i$ iga i korral.
- Antud situatsioonis me ei saa määrata $g_0\ s_0$ väärtust. Eksisteerib palju F püsipunkte.

C

- Iga funktsioon g s = s, kui s x = 0 on antud juhul püsipunktiks, kuna avaldisel

  while ¬(x = 0) do skip

  on funktsionaal F’:

$$(F'\ g)\ s = \begin{cases} g\ s & \text{if } s\ \mathrm{x} \neq 0 \\ s & \text{if } s\ \mathrm{x} = 0 \end{cases}$$

C

- Mida me päriselt tahame:

$$\mathcal{S}_{\mathrm{ds}}[\![\texttt{while } \neg(\texttt{x=0}) \texttt{ do skip}]\!]s_0 = \begin{cases} \underline{\text{undef}} & \text{if } s_0\ \texttt{x} \neq 0 \\ s_0 & \text{if } s_0\ \texttt{x} = 0 \end{cases}$$

- Seega meie eelistatud püsipunkt on:

$$g_0\ s = \begin{cases} \underline{\text{undef}} & \text{if } s\ \texttt{x} \neq 0 \\ s & \text{if } s\ \texttt{x} = 0 \end{cases}$$

- Omadus, mis eristab g_0 mõnest teisest F' püsipunktist g' on see, et ükskõik millal $g_0\ s = s'$ siis ka $g'\ s = s'$, kuid mitte vastupidi.

Püsipunkti teooria

- Püüame anda formaalse tingimuse soovitud püsipunkti FIX F eksisteerimiseks.

Osaliste funktsioonide järjestus

- Defineerime järjestuse $\sqsubseteq$ osalistel funktsioonidel: State $\hookrightarrow$ State.

- Ütleme, et $g_1 \sqsubseteq g_2$, kui funktsioon

  g : State $\hookrightarrow$ State jagab oma tulemusi osalise funktsiooniga g_2: State $\hookrightarrow$ State, st kui

  $g_1 s = s'$, siis $g_2 s = s'$ iga valitud s ja s' korral.

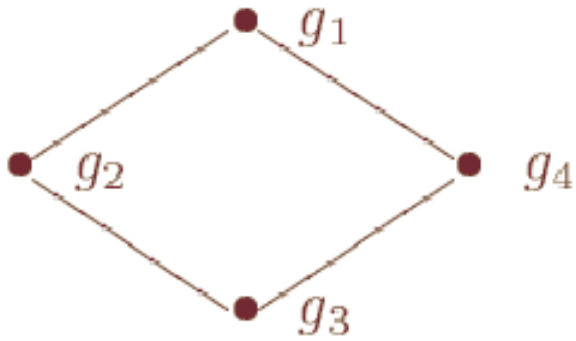
Näited

$g_1\ s = s$ for all s

$$g_2\ s = \begin{cases} s & \text{if } s\ \mathbf{x} \geq 0 \\ \underline{\text{undef}} & \text{otherwise} \end{cases}$$

$$g_3\ s = \begin{cases} s & \text{if } s\ \mathbf{x} = 0 \\ \underline{\text{undef}} & \text{otherwise} \end{cases}$$

$$g_4\ s = \begin{cases} s & \text{if } s\ \mathbf{x} \leq 0 \\ \underline{\text{undef}} & \text{otherwise} \end{cases}$$



Hasse diagramm

$g_1 \sqsubseteq g_1$,

$g_2 \sqsubseteq g_1$, $g_2 \sqsubseteq g_2$,

$g_3 \sqsubseteq g_1$, $g_3 \sqsubseteq g_2$, $g_3 \sqsubseteq g_3$, $g_3 \sqsubseteq g_4$,

$g_4 \sqsubseteq g_1$, $g_4 \sqsubseteq g_4$.

Alternatiivne esitus

- Ütleme, et $g_1 \sqsubseteq g_2$, siis ja ainult siis, kui $graph(g_1) \subseteq graph(g_2)$, kus

  $graph(g) = \{ \langle x, y \rangle \in X \times Y \mid g\ x = y \text{ ja } x \in X_g \}$

  st. iga $g\ x = y$ korral $\langle x, y \rangle \in graph(g)$ ja
  $g\ x = undef$ korral $x \notin X_g$ ehk siis
  $\neg \exists y \in Y : \langle x, y \rangle \in graph(g)$

Järjestatud hulk (1)

- **Järjestatud hulgaks** nimetame paari $(D,\sqsubseteq_D)$, kus D on hulk ja $\sqsubseteq_D$ on relatsioon hulgal D nii, et
  - $d \sqsubseteq_D d$ (refleksiivne)
  - $d_1 \sqsubseteq_D d_2$ ja $d_2 \sqsubseteq_D d_3$ järeldub $d_1 \sqsubseteq_D d_3$ (transitiivne)
  - $d_1 \sqsubseteq_D d_2$ ja $d_2 \sqsubseteq_D d_1$ järeldub $d_1 = d_2$ (antisümmeetriline)

Järjestatud hulk (2)

- Relatsiooni $\sqsubseteq_D$ nimetatakse järjestuseks hulgal D.
- $\sqsubseteq_D$ asemel võime kirjutada lihtsalt $\sqsubseteq$.

- Kui kehtib $d \sqsubseteq d'$ iga $d' \in D$, siis nimetame d hulga D **vähimaks elemendiks**.

Järjestatud hulk (3)

Fakt: Kui järjestatud hulgal (D, $\sqsubseteq$) on vähim element d, siis d on unikaalne.

Ja seda vähimat elementi tähistame lihtsalt $\bot$ (*bottom*).

Näide (1)

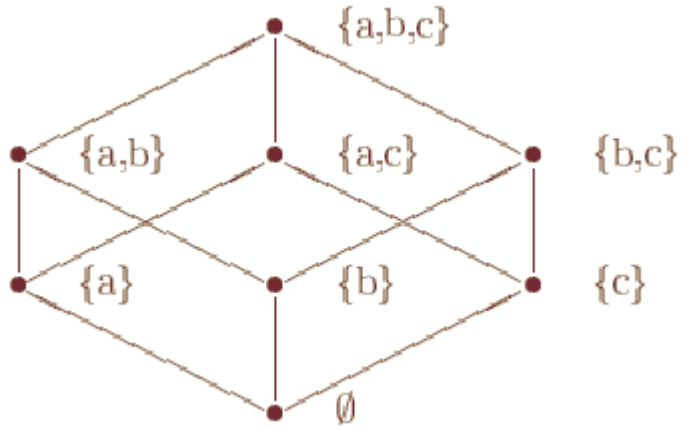
- Olgu S mittetühi hulk ja $P(S) = \{K \mid K \subseteq S\}$, siis $(P(S), \subseteq)$ on järjestatud hulk, sest
  - $\subseteq$ on refleksiivne: $K \subseteq K$
  - $\subseteq$ on transitiivne:

    kui $K_1 \subseteq K_2$ ja $K_2 \subseteq K_3$ siis $K_1 \subseteq K_3$
  - $\subseteq$ on antisümmeetriline:

    kui $K_1 \subseteq K_2$ ja $K_2 \subseteq K_1$ siis $K_1 = K_2$

Näide (2)

- Juhul kui S = {a, b, c}, siis saame järgmise diagrammi:



- Kusjuures (P (S), $\subseteq$) on ka vähim element: $\varnothing$

Lemma:

(State $\hookrightarrow$ State, $\sqsubseteq$) on järjestatud hulk, kus osaline funktsioon

$$\bot \; s = \text{undef} \quad \forall \; s \text{ korral}$$

on State $\hookrightarrow$ State vähim element.

Nõudmisi FIX F-le.

- FIX F on F püsipunkt, st. F (FIX F) = FIX F
- FIX F on vähim F püsipunkt, st. kui

  F g = g, siis FIX F $\sqsubseteq$ g.
- Analoogiliselt varem tõestatule , kui F omab vähimat püsipunki g_0, siis see on unikaalne.

- Nüüd on vaja veel veenduda, et meil esineda võivad funktsionaalid omaksid tõepoolest vähimat püsipunkti.

Täielikult järjestatud hulgad

- Olgu järjestatud hulk $(D, \sqsubseteq)$ ja $Y \subseteq D$. Siis elementi $d \in D$, mille korral

  $$\forall d' \in Y,\ d' \sqsubseteq d$$

  nimetatakse Y **ülemiseks tõkkeks**.

- Y ülemist tõket d nimetatakse **ülemiseks rajaks** siis ja ainult siis, kui d' on Y ülemine tõke ning järeldub $d \sqsubseteq d'$.

- Kui Y eksisteerib ülemine tõke, siis see on unikaalne.

- Kui Y eksisteerib ülemine raja, siis seda tähistame $\bigsqcup Y$.
- Alamhulka Y nimetatakse **ahelaks**, kui $\forall d_1, d_2 \in Y$ korral kehtib $d_1 \sqsubseteq d_2$ või $d_2 \sqsubseteq d_1$.
- Näited:
  - (P ({a, b, c}), $\subseteq$), siis alamhulk Y_0 = {∅, {a}, {a, c}} on ahel. {a,b,c} ja {a, c} on Y_0 ülemised tõkked ja {a, c} on vähim.
  - Alamhulk {∅, {a}, {c}, {a, c}} aga ei ole ahel. Kuid on olemas siiski ülemine raja {a, c}.
  - Alamhulk ∅ on ahel ja iga P ({a, b, c}) element on ülemine tõke, kusjuures ülemiseks rajaks on ∅.

Näide

- Olgu g_n : State $\hookrightarrow$ State defineeritud järgmiselt:

$$g_n\ s = \begin{cases} \underline{\text{undef}} & \text{if } s\ \mathrm{x} > \mathrm{n} \\ s[\mathrm{x} \mapsto -1] & \text{if } 0 \le s\ \mathrm{x} \text{ and } s\ \mathrm{x} \le \mathrm{n} \\ s & \text{if } s\ \mathrm{x} < 0 \end{cases}$$

- Ilmneb , et $g_n \sqsubseteq g_m$, kui $n \le m$, sest g_n on enamatel olekutel defineerimata.
- Olgu $Y_0 = \{g_n \mid n \ge 0\}$
- Siis Y_0 on ahel, kuna $g_n \sqsubseteq g_m$ iga $n \le m$.
- Y_0 vähim ülemine tõke on osaline funktsioon:

$$g\ s = \begin{cases} s[\mathrm{x} \mapsto -1] & \text{if } 0 \le s\ \mathrm{x} \\ s & \text{if } s\ \mathrm{x} < 0 \end{cases}$$

- Järjestatud hulka $(D, \sqsubseteq)$ nimetatakse **täielike ahelatega järjestatud hulgaks** (*chain complete partially ordered set* - **ccpo**), kui iga ahela Y jaoks leidub $\bigsqcup Y$

  ning nimetatakse **täielikuks võreks**, kui $\bigsqcup Y$ eksisteerib iga D alamhulga Y korral.

Fakt:

Kui $(D, \sqsubseteq)$ on ccpo, siis leidub vähim element $\bot$, mis on antud järgmiselt: $\bot = \bigsqcup \emptyset$.

Lemma:

$(State \hookrightarrow State, \sqsubseteq)$ on ccpo. Ahela Y ülemine raja on antud järgmiselt:

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

st. $(\bigsqcup Y)\ s = s'$ siis ja ainult siis, kui $g\ s = s'$ mõne $g \in Y$ jaoks.

Monotoonsed funktsioonid

- Olgu $(D, \sqsubseteq)$ ja $(D', \sqsubseteq')$ ccpo'd ja täielik funktsioon $f: D \to D'$. Siis funktsiooni f nimetame **monotoonseks**, kui

  $d_1 \sqsubseteq d_2$ järeldub $f\ d_1 \sqsubseteq' f\ d_2$ iga d_1 ja d_2 valiku korra.

Näiteid

- Olgu ccpo'd $(P(\{a, b, c\}), \subseteq)$ ja $(P(\{d, e\}), \subseteq)$ siis funktsioon $f_1 : P(\{a, b, c\}) \to P(\{d, e\})$ kirjeldatud tabelina on monotoonne:

X	{a,b,c}	{a,b}	{a,c}	{b,c}	{a}	{b}	{c}	∅
$f_1\ X$	{d,e}	{d}	{d,e}	{d,e}	{d}	{d}	{e}	∅

- Funktsioon f_2 kirjeldatud tabelina ei ole monotoonne:

X	{a,b,c}	{a,b}	{a,c}	{b,c}	{a}	{b}	{c}	∅
$f_2\ X$	{d}	{d}	{d}	{e}	{d}	{e}	{e}	{e}

Näiteid harjutamiseks

- Olgu $(P\,(\mathrm{N}), \subseteq)$ ccpo ja järgmised funktsioonid $P\,(\mathrm{N}) \to P\,(\mathrm{N})$. Millised neist on monotoonsed:
  - $f_1\ X = \mathbf{N} \setminus X$
  - $f_2\ X = X \cup \{\mathbf{27}\}$
  - $f_3\ X = X \cap \{\mathbf{7}, \mathbf{9}, \mathbf{13}\}$
  - $f_4\ X = \{\ n \in X \mid n \text{ is a prime}\ \}$
  - $f_5\ X = \{\ \mathbf{2} \star n \mid n \in X\ \}$

$$(\mathbf{State} \hookrightarrow \mathbf{State}) \to (\mathbf{State} \hookrightarrow \mathbf{State})$$

- $F_0\ g = g$
- $F_1\ g = \begin{cases} g_1 & \text{if } g = g_2 \\ g_2 & \text{otherwise} \end{cases}$ where $g_1 \neq g_2$
- $(F'\ g)\ s = \begin{cases} g\ s & \text{if } s\ \mathbf{x} \neq \mathbf{0} \\ s & \text{if } s\ \mathbf{x} = \mathbf{0} \end{cases}$

Monotoonsete funktsioonide omadused

Fakt:

Olgu $(D, \sqsubseteq)$, $(D', \sqsubseteq')$ ja $(D'', \sqsubseteq'')$ ccpo'd (täielike ahelatega järjestatud hulgad) ja olgu $f : D \to D'$ ja $f' : D' \to D''$ monotoonsed funktsioonid.

Siis $f' \circ f : D \to D''$ on ka monotoonne.

Lemma:

Olgu $(D, \sqsubseteq)$, $(D', \sqsubseteq')$ ccpo'd (täielike ahelatega järjestatud hulgad) ja $f: D \to D'$ monotoonne funktsioon. Kui Y on ahel hulgas D, siis $\{f\ d \mid d \in Y\}$ on ahel hulgas D'. Ja veelgi enam, kehtib

$$\bigsqcup' \{f\ d \mid d \in Y\} \sqsubseteq' f\ (\bigsqcup Y)$$

- Üldiselt me **ei saa** loota, et monotoonne funktsioon säilitab ülemise raja

$$\sqcup'\{f\ d \mid d \in Y\} = f\ (\sqcup Y)$$

- Näiteks

$(\mathcal{P}(\mathbf{N} \cup \{a\}), \subseteq)$ $\quad f\colon \mathcal{P}(\mathbf{N} \cup \{a\}) \to \mathcal{P}(\mathbf{N} \cup \{a\})$

$$f\ X = \begin{cases} X & \text{if } X \text{ is finite} \\ X \cup \{a\} & \text{if } X \text{ is infinite} \end{cases}$$

$$Y = \{\ \{0,1,\cdots,n\} \mid n \geq 0\ \}$$

- Me ütleme, et funktsioon f: D $\rightarrow$ D´, mis on defineeritud ccpo'del (D,$\sqsubseteq$), (D´,$\sqsubseteq$ ´), on **pidev**, kui ta on monotoonne ja kehtib

  $$\bigsqcup' \{f\ d \mid d \in Y\} = f\ (\bigsqcup Y)$$

  kõigi mittetühjade ahelate Y korral.

- Kui $\bigsqcup \{f\ d \mid d \in Y\} = f\ (\bigsqcup Y)$ kehtib tühja ahela korral, st. $\bot = f\ \bot$, siis funktsiooni f nimetatakse **agaraks** (*strict*).

Näide

- Vaata lk 40 esimest funktsiooni. See on rangelt pidev.

Lemma:

Olgu $(D, \sqsubseteq)$, $(D', \sqsubseteq')$ ja $(D'', \sqsubseteq'')$ ccpo'd ja olgu $f : D \to D'$ ja $f' : D' \to D''$ pidevad funktsioonid. Siis $f' \circ f : D \to D''$ on ka pidev.

Kui antud funktsioonid on ranged, siis ka nende kompositsioon on range.

Teoreem:

Olgu $f : D \to D$ pidev funktsioon ccpo'l $(D, \sqsubseteq)$ vähima elemendiga $\bot$. Siis

$$FIX\ f = \bigsqcup \{f^n \bot \mid n \geq 0\}$$

defineerib D elemendi ja see element on f vähim püsipunkt.

Otsestiili semantika eksistents

- Peame veenduma , et

$$\mathcal{S}_{\mathrm{ds}}[\![\texttt{while}\ b\ \texttt{do}\ S]\!] = \mathrm{FIX}\ F$$
$$\text{where } F\ g = \mathrm{cond}(\mathcal{B}[\![b]\!], g \circ \mathcal{S}_{\mathrm{ds}}[\![S]\!], \mathrm{id})$$

  tõepoolest defineerib funktsiooni.

- Selleks peame näitama, et F on pidev.

- Paneme tähele, et F g = F_1 (F_2 g), kus

$$F_1\ g = \mathrm{cond}(\mathcal{B}[\![b]\!], g, \mathrm{id})$$
$$F_2\ g = g \circ \mathcal{S}_{\mathrm{ds}}[\![S]\!]$$

- Kuna pidevate funktsioonide kompositsioon on pidev, siis meil on vaja näidata, et F_1 ja F_2 on pidevad.

Lemma 4.43:

Olgu $g_0 : State \hookrightarrow State$, $p : State \to T$ ja $F\ g = cond(p, g, g_0)$, siis F on pidev.

Lemma 4.45:

Olgu $g_0 : State \hookrightarrow State$ ja $F\ g = g \circ g_0$, siis F on pidev.

Väide:

While keele semantilised võrrandid (vt. slaidi 5) defineerivad täieliku funktsiooni

$$S_{ds} : Stm \rightarrow (State \rightarrow State)$$

Näide

- Vaatame faktoriaali lause denotatsioon-semantikat.

$$\mathcal{S}_{\mathrm{ds}}[\![\mathtt{y} := \mathtt{1};\ \mathtt{while}\ \neg(\mathtt{x}{=}\mathtt{1})\ \mathtt{do}\ (\mathtt{y}{:=}\mathtt{y}{\star}\mathtt{x};\ \mathtt{x}{:=}\mathtt{x}{-}\mathtt{1})]\!]$$

- Rakendame seda olekule s_0, milles x = 3.

$$\begin{aligned}&\mathcal{S}_{\mathrm{ds}}[\![\mathtt{y} := \mathtt{1};\ \mathtt{while}\ \neg(\mathtt{x}{=}\mathtt{1})\ \mathtt{do}\ (\mathtt{y}{:=}\mathtt{y}{\star}\mathtt{x};\ \mathtt{x}{:=}\mathtt{x}{-}\mathtt{1})]\!]\ s_0\\ &\quad = (\mathrm{FIX}\ F)\ s_0[\mathtt{y}{\mapsto}\mathtt{1}]\end{aligned}$$

kus

$$F\ g\ s = \begin{cases} g\ (\mathcal{S}_{\mathrm{ds}}[\![\mathtt{y}{:=}\ \mathtt{y}{\star}\mathtt{x};\ \mathtt{x}{:=}\mathtt{x}{-}\mathtt{1}]\!]\ s) & \text{if } \mathcal{B}[\![\neg(\mathtt{x}{=}\mathtt{1})]\!]\ s = \mathbf{tt} \\ s & \text{if } \mathcal{B}[\![\neg(\mathtt{x}{=}\mathtt{1})]\!]\ s = \mathbf{ff} \end{cases}$$

ehk

$$F\ g\ s = \begin{cases} g\ (s[\mathtt{y}{\mapsto}(s\ \mathtt{y}){\star}(s\ \mathtt{x})][\mathtt{x}{\mapsto}(s\ \mathtt{x}){-}\mathtt{1}]) & \text{if } s\ \mathtt{x} \neq \mathtt{1} \\ s & \text{if } s\ \mathtt{x} = \mathtt{1} \end{cases}$$

Semantiline ekvivalents

- Ütleme, et laused S1 ja S2 on semantiliselt ekvivalentsed, kui

$$\mathcal{S}_{\mathrm{ds}}[\![S_1]\!] = \mathcal{S}_{\mathrm{ds}}[\![S_2]\!]$$

Struktuurse- ja denotatsioonsemantika samaväärsus

- Teoreem: While keele iga lause S korral kehtib

$$\mathcal{S}_{\mathrm{scs}}[\![S]\!] = \mathcal{S}_{\mathrm{ds}}[\![S]\!]$$

- Mõlemad semantilised funktsioonid kohal S on State $\hookrightarrow$ State.
- Järjestatud hulga elemendid d1 ja d2 on võrdsed, kui d1 $\sqsubseteq$ d2 ja d2 $\sqsubseteq$ d1.

- Seega teoreemi tõestamiseks piisab kahe järgmise lemma tõestamisest:

Lemma 4.56:

While keele iga lause S korral kehtib

$$\mathcal{S}_{sos}[\![S]\!] \sqsubseteq \mathcal{S}_{ds}[\![S]\!]$$

Lemma 4.57:

While keele iga lause S korral kehtib

$$\mathcal{S}_{ds}[\![S]\!] \sqsubseteq \mathcal{S}_{sos}[\![S]\!]$$

Laiendame While keelt - Proc

- Toome sisse plokkstruktuuri lokaalsete muutujate ja protseduuride deklaratsioonid
- Abstraktne süntaks:

$$
\begin{array}{lcl}
S & ::= & x := a \mid \texttt{skip} \mid S_1 \,;\, S_2 \mid \texttt{if } b \texttt{ then } S_1 \texttt{ else } S_2 \\
 & \mid & \texttt{while } b \texttt{ do } S \mid \texttt{begin } D_V \; D_P \; S \texttt{ end} \mid \texttt{call } p \\
D_V & ::= & \texttt{var } x := a; \, D_V \mid \varepsilon \\
D_P & ::= & \texttt{proc } p \texttt{ is } S; \, D_P \mid \varepsilon
\end{array}
$$

Skoopimine

- Staatiline (call p täitmine sisemise blokis modifitseerib globaalset muutujat x)
- Dünaamiline (call p täitmine sisemises blokis modifitseerib lokaalset muutujat x)

```
begin var x := 7; proc p is x := 0;
      begin var x := 5; call p end
end
```

- Iga muutujaga seostame unikaalse aadressi *(location)* ja iga aadressiga seostame väärtuse.

- Seega senise **State** asemel, kus muutuja seoti vahetult tema väärtusega, kasutame **Store** (väärtuste kogum), kus muutuja aadressid on seotud väärtustega ja $\mathbf{Env_V}$ (muutuja keskkond), kus muutuja on seotud unikaalse aadressiga.

- Aadresside domeen

  $\mathrm{Loc} = \mathrm{Z}$

- Operatsioon järgmise aadressi saamiseks

  $\mathrm{new} : \mathrm{Loc} \rightarrow \mathrm{Loc}$

- Väärtuste kogum, kus *next* erimärk, milles hoitakse järgmist vaba aadressi.

  $\mathrm{Store} = \mathrm{Loc} \cup \{\mathrm{next}\} \rightarrow \mathrm{Z}$

- Muutuja keskkond

  $\mathrm{Env_V} = \mathrm{Var} \rightarrow \mathrm{Loc}$

- Seega senine olek $s = sto \circ env_V$, seega defineerime funktsiooni:

  $lookup\ \ env_V\ \ sto = sto \circ env_V$

  $lookup: Env_V \to Store \to State$

- Nüüd saame kirja panna uue semantilise funktsiooni S'_{ds}:

  $$S'_{ds} : Stm \to Env_V \to (Store \hookrightarrow Store)$$

While keele laused S'_{ds} –ga.

$$\mathcal{S}'_{\mathrm{ds}}[\![x{:=}a]\!]env_V\ sto = sto[l \mapsto \mathcal{A}[\![a]\!](\mathrm{lookup}\ env_V\ sto)]$$

$$\text{where } l = env_V\ x$$

$$\mathcal{S}'_{\mathrm{ds}}[\![\mathtt{skip}]\!]env_V = \mathrm{id}$$

$$\mathcal{S}'_{\mathrm{ds}}[\![S_1\ ;\ S_2]\!]env_V = (\mathcal{S}'_{\mathrm{ds}}[\![S_2]\!]env_V) \circ (\mathcal{S}'_{\mathrm{ds}}[\![S_1]\!]env_V)$$

$$\mathcal{S}'_{\mathrm{ds}}[\![\mathtt{if}\ b\ \mathtt{then}\ S_1\ \mathtt{else}\ S_2]\!]env_V =$$

$$\mathrm{cond}(\mathcal{B}[\![b]\!] \circ (\mathrm{lookup}\ env_V), \mathcal{S}'_{\mathrm{ds}}[\![S_1]\!]env_V, \mathcal{S}'_{\mathrm{ds}}[\![S_2]\!]env_V)$$

$$\mathcal{S}'_{\mathrm{ds}}[\![\mathtt{while}\ b\ \mathtt{do}\ S]\!]env_V = \mathrm{FIX}\ F$$

$$\text{where } F\ g = \mathrm{cond}(\mathcal{B}[\![b]\!] \circ (\mathrm{lookup}\ env_V), g \circ (\mathcal{S}'_{\mathrm{ds}}[\![S]\!]env_V), \mathrm{id})$$

Muutuja keskkonna uuendamine

- Muutuja keskkonda uuendatakse iga kord, kui sisenetakse blokki, mis sisaldab lokaalseid muutujaid.
- Muutuja deklaratsiooni süntaktilise kategooria jaoks võtame selleks otstarbeks kasutusele semantilise funktsiooni:

$$\mathcal{D}^{V}_{ds}\colon \mathbf{Dec}_V \to \mathbf{Env}_V \times \mathbf{Store} \to \mathbf{Env}_V \times \mathbf{Store}$$

- $\mathcal{D}^{\mathrm{V}}_{\mathrm{ds}}[\![D_V]\!]$ võtab argumentideks paari, mis koosneb hetke muutuja keskkonna ja väärtuste kogumi ning tagastab mõlemad uuendatuna.

$$\mathcal{D}^{\mathrm{V}}_{\mathrm{ds}}[\![\texttt{var } x := a;\ D_V]\!](env_V, sto) =$$
$$\mathcal{D}^{\mathrm{V}}_{\mathrm{ds}}[\![D_V]\!](env_V[x \mapsto l],\ sto[l \mapsto v][\mathrm{next} \mapsto \mathrm{new}\ l])$$
$$\text{where } l = sto\ \mathrm{next} \text{ and } v = \mathcal{A}[\![a]\!](\mathrm{lookup}\ env_V\ sto)$$

$$\mathcal{D}^{\mathrm{V}}_{\mathrm{ds}}[\![\varepsilon]\!] = \mathrm{id}$$

- Kuni pole protseduure veel kasutusse võetud, võime laiendada semantilist funktsiooni S'_{ds}:

$$\mathcal{S}'_{\mathrm{ds}}[\![\texttt{begin } D_V \ S \ \texttt{end}]\!]\, env_V \ sto = \mathcal{S}'_{\mathrm{ds}}[\![S]\!]\, env'_V \ sto'$$
$$\text{where } \mathcal{D}^{\mathrm{V}}_{\mathrm{ds}}[\![D_V]\!](env_V, sto) = (env'_V, sto')$$

Protseduuri keskkond

- On täielik funktsioon, mis seob iga protseduuri tema täidetava keha efektiga:

$$\mathbf{Env_P} = \mathbf{Pname} \to (\mathbf{Store} \hookrightarrow \mathbf{Store})$$

- Protseduuri keskkonna uuendamiseks kasutame semantilist funktsiooni protseduuri deklaratsioonil:

$$\mathcal{D}^{\mathbf{P}}_{\mathbf{ds}}\colon \mathbf{Dec_P} \to \mathbf{Env_V} \to \mathbf{Env_P} \to \mathbf{Env_P}$$

Mitterekursiivsete protseduride $\mathcal{D}^{\mathbf{P}}_{\mathrm{ds}}$

$$\mathcal{D}^{\mathbf{P}}_{\mathrm{ds}}[\![\texttt{proc } p \texttt{ is } S;\ D_P]\!]env_V\ env_P = \mathcal{D}^{\mathbf{P}}_{\mathrm{ds}}[\![D_P]\!]env_V\ (env_P[p \mapsto g])$$

$$\text{where } g = \mathcal{S}_{\mathrm{ds}}[\![S]\!]env_V\ env_P$$

$$\mathcal{D}^{\mathbf{P}}_{\mathrm{ds}}[\![\varepsilon]\!]env_V = \mathrm{id}$$

Semantiline funktsioon keeles Proc

$$\mathcal{S}_{\mathrm{ds}}\colon \mathbf{Stm} \to \mathbf{Env_V} \to \mathbf{Env_P} \to (\mathbf{Store} \hookrightarrow \mathbf{Store})$$

$$\mathcal{S}_{\mathrm{ds}}[\![x{:=}a]\!]\,env_V\ env_P\ sto = sto[l \mapsto \mathcal{A}[\![a]\!](\mathrm{lookup}\ env_V\ sto)]$$
$$\text{where } l = env_V\ x$$

$$\mathcal{S}_{\mathrm{ds}}[\![\texttt{skip}]\!]\,env_V\ env_P = \mathrm{id}$$

$$\mathcal{S}_{\mathrm{ds}}[\![S_1\ ;\ S_2]\!]\,env_V\ env_P = (\mathcal{S}_{\mathrm{ds}}[\![S_2]\!]\,env_V\ env_P) \circ (\mathcal{S}_{\mathrm{ds}}[\![S_1]\!]\,env_V\ env_P)$$

$$\mathcal{S}_{\mathrm{ds}}[\![\texttt{if}\ b\ \texttt{then}\ S_1\ \texttt{else}\ S_2]\!]\,env_V\ env_P =$$
$$\mathrm{cond}(\mathcal{B}[\![b]\!]\circ(\mathrm{lookup}\ env_V), \mathcal{S}_{\mathrm{ds}}[\![S_1]\!]\,env_V\ env_P,$$
$$\mathcal{S}_{\mathrm{ds}}[\![S_2]\!]\,env_V\ env_P)$$

$$\mathcal{S}_{\mathrm{ds}}[\![\texttt{while}\ b\ \texttt{do}\ S]\!]\,env_V\ env_P = \mathrm{FIX}\ F$$
$$\text{where } F\ g = \mathrm{cond}(\mathcal{B}[\![b]\!]\circ(\mathrm{lookup}\ env_V),$$
$$g \circ (\mathcal{S}_{\mathrm{ds}}[\![S]\!]\,env_V\ env_P), \mathrm{id})$$

$$\mathcal{S}_{\mathrm{ds}}[\![\texttt{begin}\ D_V\ D_P\ S\ \texttt{end}]\!]\,env_V\ env_P\ sto = \mathcal{S}_{\mathrm{ds}}[\![S]\!]\,env'_V\ env'_P\ sto'$$
$$\text{where } \mathcal{D}^{\mathrm{V}}_{\mathrm{ds}}[\![D_V]\!](env_V, sto) = (env'_V, sto')$$
$$\text{and } \mathcal{D}^{\mathrm{P}}_{\mathrm{ds}}[\![D_P]\!]\,env'_V\ env_P = env'_P$$

$$\mathcal{S}_{\mathrm{ds}}[\![\texttt{call}\ p]\!]\,env_V\ env_P = env_P\ p$$

Rekursiivsed protseduurid

$$\mathcal{D}^{\mathrm{P}}_{\mathrm{ds}}[\![\texttt{proc } p \texttt{ is } S; D_P]\!] env_V\ env_P = \mathcal{D}^{\mathrm{P}}_{\mathrm{ds}}[\![D_P]\!] env_V\ (env_P[p \mapsto \mathrm{FIX}\ F])$$

$$\text{where } F\ g = \mathcal{S}_{\mathrm{ds}}[\![S]\!] env_V\ (env_P[p \mapsto g])$$

$$\mathcal{D}^{\mathrm{P}}_{\mathrm{ds}}[\![\varepsilon]\!] env_V = \mathrm{id}$$