Syntaks og semantik

Lektion 12

10 april 2007

Blokke

Bip = **Bims** + blokke og parameterløse procedurer:

Kom: $S ::= x\texttt{:=}a \mid \texttt{skip} \mid S_1\texttt{;}S_2 \mid \texttt{if } b \texttt{ then } S_1 \texttt{ else } S_2 \mid \texttt{while } b \texttt{ do } S \mid \texttt{begin } D_V\ D_P\ S \texttt{ end} \mid \texttt{call } p$

ErkV: $D_V ::= \texttt{var } x\texttt{:=}a\texttt{;}D_V \mid \varepsilon$

ErkP: $D_P ::= \texttt{proc } p \texttt{ is } S\texttt{;}D_P \mid \varepsilon$

- *lokale* erklæringer af variable (**ErkV**) og procedurer (**ErkP**) i en blok
- variable *initialiseres* ved erklæring
- semantikken af *procedurekald* afhænger af scope-regler
- *bogen* beskæftiger sig både med dynamisk og statisk scope
- *vi* lægger mest vægt på *statisk* scope her

- brug for ny tilstandsmodel for at kunne erklære variable
- før: **Tilstande** = **Var** $\rightharpoonup \mathbb{Z}$
- nu: **Var** $\rightharpoonup$ **Loc** $\rightharpoonup \mathbb{Z}$
- **Loc**: lokationer; lager-adresser

$\Rightarrow$ en tilstand (env_V, sto) beskrives ved:

env_V variabel-environment

- hvilken adresse er en given variabel bundet til?
- $\mathbf{Env}_V = \mathbf{Var} \cup \{\text{next}\} \rightharpoonup \mathbf{Loc}$
- next peger til næste *frie* lokation
- *for os:* **Loc** = $\mathbb{Z}$
- opdatering: $env_V[x \mapsto \ell](x') = \begin{cases} env_V(x') & \text{hvis } x' \neq x \\ \ell & \text{hvis } x' = x \end{cases}$

sto store

- hvilken værdi indeholder en given adresse?
- **Store** = **Loc** $\rightharpoonup \mathbb{Z}$
- opdatering: $sto[\ell \mapsto v](\ell') = \begin{cases} sto(\ell') & \text{hvis } \ell' \neq \ell \\ v & \text{hvis } \ell' = \ell \end{cases}$

$$[\text{plus}_{\text{bss}}] \quad \frac{env_V, sto \vdash a_1 \rightarrow_a v_1 \quad env_V, sto \vdash a_2 \rightarrow_a v_2}{env_V, sto \vdash a_1 + a_2 \rightarrow_a v} \quad \text{hvor } v = v_1 + v_2$$

$$[\text{minus}_{\text{bss}}] \quad \frac{env_V, sto \vdash a_1 \rightarrow_a v_1 \quad env_V, sto \vdash a_2 \rightarrow_a v_2}{env_V, sto \vdash a_1 - a_2 \rightarrow_a v} \quad \text{hvor } v = v_1 - v_2$$

$$[\text{mult}_{\text{bss}}] \quad \frac{env_V, sto \vdash a_1 \rightarrow_a v_1 \quad env_V, sto \vdash a_2 \rightarrow_a v_2}{env_V, sto \vdash a_1 * a_2 \rightarrow_a v} \quad \text{hvor } v = v_1 \cdot v_2$$

$$[\text{parent}_{\text{bss}}] \quad \frac{env_V, sto \vdash a_1 \rightarrow_a v_1}{env_V, sto \vdash (a_1) \rightarrow_a v_1}$$

$$[\text{num}_{\text{bss}}] \quad env_V, sto \vdash n \rightarrow_a v \quad \text{hvis } \mathcal{N}[\![n]\!] = v$$

$$[\text{var}_{\text{bss}}] \quad env_V, sto \vdash x \rightarrow_a v \quad \text{hvis } sto(env_V(x)) = v$$

ErkV: $D_V ::= \texttt{var}\ x\texttt{:=}a\texttt{;}\ D_V \mid \varepsilon$

- erklæringer modificerer env_V (pga. nye variable) og sto (pga. nye værdier til nye variable)

⇒ transitionssystem:

  - konfigurationer $\Gamma_{DV} = \textbf{ErkV} \times \textbf{Env}_V \times \textbf{Store} \cup \textbf{Env}_V \times \textbf{Store}$
  - slutkonfigurationer $T_{DV} = \textbf{Env}_V \times \textbf{Store}$
  - dvs. konfigurationer (D_V, env_V, sto) og (env_V, sto)

$$[\text{var-erkl}_{\text{bss}}] \quad \frac{\langle D_V, env_V[x \mapsto \ell][\text{next} \mapsto \text{new}(\ell)], sto[\ell \mapsto v]\rangle \rightarrow_{DV} \langle env'_V, sto'\rangle}{\langle \texttt{var}\ x\texttt{:=}a\texttt{;}\ D_V, env_V, sto\rangle \rightarrow_{DV} \langle env'_V, sto'\rangle} \quad \text{hvor } env_V, sto \vdash a \rightarrow_a v \text{ og } \ell = env_V(\text{next})$$

$$[\text{tom-var-erkl}_{\text{bss}}] \quad \langle \varepsilon, env_V, sto\rangle \rightarrow_{DV} \langle env_V, sto\rangle$$

- big-step: variabelerklæringer sker i ét hug
- new : **Loc** → **Loc** giver næste lokation; $\text{new}(\ell) = \ell + 1$

- også procedure-environment $env_P \in \textbf{Env}_P$, til at holde styr på *procedurer*
- med tilhørende big-step-semantik for procedure-erklæringer $(\Gamma_{DP}, \rightarrow_{DP}, T_{DP})$
- men det snakker vi om senere
- dvs. procedure-environment env_P, variabel-environment env_V og store sto
- men kommandoer kan ikke ændre env_V og env_P !

⇒ transitioner på formen $env_V, env_P \vdash \langle S, sto\rangle \rightarrow sto'$

- dvs. konfigurationer **Kom** × **Store** ∪ **Store**
- og slutkonfigurationer **Store**

$$[\text{ass}_{\text{bss}}] \quad env_V, env_P \vdash \langle x\texttt{:=}a, sto\rangle \rightarrow sto[\ell \mapsto v] \quad \text{hvor } env_V, sto \vdash a \rightarrow_a v \text{ og } env_V(x) = \ell$$

$$[\text{skip}_{\text{bss}}] \quad env_V, env_P \vdash \langle \texttt{skip}, sto\rangle \rightarrow sto$$

$$[\text{comp}_{\text{bss}}] \quad \frac{env_V, env_P \vdash \langle S_1, sto\rangle \rightarrow sto'' \quad env_V, env_P \vdash \langle S_2, sto''\rangle \rightarrow sto'}{env_V, env_P \vdash \langle S_1\texttt{;}\ S_2, sto\rangle \rightarrow sto'}$$

$$[\text{if-sand}_{\text{bss}}] \quad \frac{env_V, env_P \vdash \langle S_1, sto\rangle \rightarrow sto'}{env_V, env_P \vdash \langle \texttt{if}\ b\ \texttt{then}\ S_1\ \texttt{else}\ S_2\ , sto\rangle \rightarrow sto'} \quad \text{hvis } env_V, sto \vdash b \rightarrow_b tt$$

$$[\text{if-falsk}_{\text{bss}}] \quad \frac{env_V, env_P \vdash \langle S_2, sto\rangle \rightarrow sto'}{env_V, env_P \vdash \langle \texttt{if}\ b\ \texttt{then}\ S_1\ \texttt{else}\ S_2\ , sto\rangle \rightarrow sto'} \quad \text{hvis } env_V, sto \vdash b \rightarrow_b ff$$

$$[\text{while-sand}_{\text{bss}}] \quad \frac{env_V, env_P \vdash \langle S, sto\rangle \to sto'' \quad env_V, env_P \vdash \langle \texttt{while}\ b\ \texttt{do}\ S, sto''\rangle \to sto'}{env_V, env_P \vdash \langle \texttt{while}\ b\ \texttt{do}\ S, sto\rangle \to sto'} \text{ hvis } env_V, sto \vdash b \to_b tt$$

$$[\text{while-falsk}_{\text{bss}}] \quad env_V, env_P \vdash \langle \texttt{while}\ b\ \texttt{do}\ S, sto\rangle \to sto \text{ hvis } env_V, sto \vdash b \to_b ff$$

$$[\text{blok}_{\text{bss}}] \quad \frac{\langle D_V, env_V, sto\rangle \to_{DV} \langle env'_V, sto''\rangle \quad env'_V \vdash \langle D_P, env_P\rangle \to_{DP} env'_P \quad env'_V, env'_P \vdash \langle S, sto''\rangle \to sto'}{env_V, env_P \vdash \langle \texttt{begin}\ D_V\ D_P\ S\ \texttt{end}, sto\rangle \to sto'}$$


```
begin var x:= 0;
      var y:= 42
      proc p is x:= x+3;
      proc q is call p;
      begin var x:= 9;
                proc p is x:= x+1;
                call q;
                y:=x
      end
end
```

- dynamisk binding af variable og procedurer: $\texttt{y} = 10$
- statisk binding af variable og procedurer: $\texttt{y} = 9$
- også muligt: *statisk* binding af variable og *dynamisk* binding af procedurer, og omvendt

- statisk binding af variable og procedurer: ved procedure*kald* skal anvendes det variabel- og procedure-environment der fandtes ved *erklæringen*

⇒ procedurer skal huske env_V og env_P

⇒ $\mathbf{Env}_P = \mathbf{Pnavne} \rightharpoonup \mathbf{Kom} \times \mathbf{Env}_V \times \mathbf{Env}_P$

- (**Pnavne** : procedurenavne)
- dvs. $\mathbf{Env}_P$ består af partielle afbildninger $p \mapsto \langle S, env_V, env_P\rangle$
  - S : procedure"kroppen"
  - env_V, env_P : variabel- og procedure-environment da p blev erklæret
- en rekursiv definition!
- big-step-semantik:
  - tilstande $\mathbf{ErkP} \times \mathbf{Env}_P \cup \mathbf{Env}_P$
  - sluttilstande $\mathbf{Env}_P$
  - transitioner $\to_{DP}$

$$[\text{proc}_{\text{bss}}] \quad \frac{env_V \vdash \langle D_P, env_P[p \mapsto (S, env_V, env_P)]\rangle \to_{DP} env'_P}{env_V \vdash \langle \texttt{proc}\ p\ \texttt{is}\ S\ ;\ D_P, env_P\rangle \to_{DP} env'_P}$$

$$[\text{proc-tom}_{\text{bss}}] \quad env_V \vdash \langle \varepsilon, env_P\rangle \to_{DP} env_P$$

$$[\text{call}_{\text{bss}}] \quad \frac{env'_V[\text{next} \mapsto \ell], env'_P \vdash \langle S, sto\rangle \to sto'}{env_V, env_P \vdash \langle \texttt{call}\ p, sto\rangle \to sto'} \text{ hvor } env_P(p) = (S, env'_V, env'_P) \text{ og } \ell = env_V(\text{next})$$

Procedurer med parametre

At udvide **Bip** med procedurer med én referenceparameter:

Kom: $S ::= x\texttt{:=}a \mid \texttt{skip} \mid S_1\texttt{;}S_2 \mid \texttt{if } b \texttt{ then } S_1 \texttt{ else } S_2 \mid \texttt{while } b \texttt{ do } S \mid \texttt{begin } D_V\ D_P\ S \texttt{ end} \mid \texttt{call } p\texttt{(}y\texttt{)}$

ErkV: $D_V ::= \texttt{var } x\texttt{:=}a\texttt{;}\, D_V \mid \varepsilon$

ErkP: $D_P ::= \texttt{proc } p\texttt{(var } x\texttt{) is } S\texttt{;}\, D_P \mid \varepsilon$

- referenceparametre: den formelle parameter x er en *reference* til *adressen* på den aktuelle parameter y
- Eksempel:

```
begin
   var y:=3;
   proc p(var x) is x:= x+1;
   call p(y)
end
```

$\Rightarrow$ y = 4

- procedure-environment:

$$\mathbf{Env}_P = \mathbf{Pnavne} \rightharpoonup \mathbf{Kom} \times \mathbf{Var} \times \mathbf{Env}_V \times \mathbf{Env}_P$$

- skal huske navnet på den formelle parameter
- at erklære procedurer:

$$[\text{proc}_{\text{bss}}] \quad \frac{env_V \vdash \langle D_P, env_P[p \mapsto (S, x, env_V, env_P)]\rangle \rightarrow_{DP} env'_P}{env_V \vdash \langle \texttt{proc } p\texttt{(var } x\texttt{) is } S\texttt{;}\, D_P, env_P\rangle \rightarrow_{DP} env'_P}$$

$$[\text{proc-tom}_{\text{bss}}] \quad env_V \vdash \langle \varepsilon, env_P\rangle \rightarrow_{DP} env_P$$

- at kalde procedurer:

$$[\text{call-ref}_{\text{bss}}] \quad \frac{env'_V[x \mapsto \ell][\text{next} \mapsto \ell'], env'_P \vdash \langle S, sto\rangle \rightarrow sto'}{env_V, env_P \vdash \langle \texttt{call } p(y), sto\rangle \rightarrow sto'}$$

hvor $env_P(p) = (S, x, env'_V, env'_P)$, $\ell = env_V(y)$ og $\ell' = env_V(\text{next})$

$$[\text{call-ref}_{\text{bss}}] \quad \frac{env'_V[x \mapsto \ell][\text{next} \mapsto \ell'], env'_P \vdash \langle S, sto\rangle \rightarrow sto'}{env_V, env_P \vdash \langle \texttt{call } p(y), sto\rangle \rightarrow sto'}$$

hvor $env_P(p) = (S, x, env'_V, env'_P)$, $\ell = env_V(y)$ og $\ell' = env_V(\text{next})$

Problem: dén regel tillader ikke rekursive procedurekald

- fordi env'_P er procedure-environmentet fra før p blev erklæret

Løsning: ny regel:

$$[\text{call-ref-rec}_{\text{bss}}] \quad \frac{\begin{array}{r} env'_V[x \mapsto \ell][\text{next} \mapsto \ell'], env'_P[p \mapsto (S, x, env'_V, env'_P)] \\ \vdash \langle S, sto\rangle \rightarrow sto' \end{array}}{env_V, env_P \vdash \langle \texttt{call } p(y), sto\rangle \rightarrow sto'}$$

hvor $env_P(p) = (S, x, env'_V, env'_P)$, $\ell = env_V(y)$ og $\ell' = env_V(\text{next})$

At udvide **Bip** med procedurer med én værdiparameter:

Kom: $S ::= x\texttt{:=}a \mid \texttt{skip} \mid S_1\texttt{;}\,S_2 \mid \texttt{if}\ b\ \texttt{then}\ S_1\ \texttt{else}\ S_2$
$\mid \texttt{while}\ b\ \texttt{do}\ S$
$\mid \texttt{begin}\ D_V\ D_P\ S\ \texttt{end} \mid \texttt{call}\ p\texttt{(}a\texttt{)}$

ErkV: $D_V ::= \texttt{var}\ x\texttt{:=}a\texttt{;}\,D_V \mid \varepsilon$

ErkP: $D_P ::= \texttt{proc}\ p\texttt{(var}\ x\texttt{)}\ \texttt{is}\ S\texttt{;}\,D_P \mid \varepsilon$

- værdiparametre: den formelle parameter x bliver til en lokal variabel i proceduren, med *startværdi* = værdien af den aktuelle parameter
- Eksempel:

```
begin
   var y:=3;
   proc p(var x) is x:= x+1;
   call p(y)
end
```

$\Rightarrow$ `y` $= 3$

- procedure-erklæringer (uændret):

$$[\text{proc}_{\text{bss}}]\quad \frac{env_V \vdash \langle D_P, env_P[p \mapsto (S, x, env_V, env_P)]\rangle \rightarrow_{DP} env'_P}{env_V \vdash \langle \texttt{proc}\ p\texttt{(var}\ x\texttt{)}\ \texttt{is}\ S\texttt{;}\,D_P, env_P\rangle \rightarrow_{DP} env'_P}$$

$$[\text{proc-tom}_{\text{bss}}]\quad env_V \vdash \langle \varepsilon, env_P\rangle \rightarrow_{DP} env_P$$

- procedurekald:

$$[\text{call-val}_{\text{bss}}]\quad \frac{env'_V[x \mapsto \ell][\text{next} \mapsto \text{new}(\ell)], env'_P \vdash \langle S, sto[\ell \mapsto v]\rangle \rightarrow sto'}{env_V, env_P \vdash \langle \texttt{call}\ p(a), sto\rangle \rightarrow sto'}$$
hvor $env_P(p) = (S, x, env'_V, env'_P)$, $env_V, sto \vdash a \rightarrow_a v$ og $\ell = env_V(\text{next})$

$$[\text{call-val-rec}_{\text{bss}}]\quad \frac{\begin{array}{c} env'_V[x \mapsto \ell][\text{next} \mapsto \text{new}(\ell)], \\ env'_P[p \mapsto (S, x, env'_V, env'_P)] \\ \vdash \langle S, sto[\ell \mapsto v]\rangle \rightarrow sto' \end{array}}{env_V, env_P \vdash \langle \texttt{call}\ p(a), sto\rangle \rightarrow sto'}$$
hvor $env_P(p) = (S, x, env'_V, env'_P)$, $env_V, sto \vdash a \rightarrow_a v$ og $\ell = env_V(\text{next})$