F09

Extensions of **While**

Lennart Andersson

Reviderad 2011-03-30

2011

S_1 par S_2, structural operational semantics

¡$S_1, \sigma > \Rightarrow \sigma' < S_1$ `par` $S_2, \sigma > \Rightarrow < S_2, \sigma' >$

¡$S_1, \sigma > \Rightarrow < S_1', \sigma' > < S_1$ `par` $S_2, \sigma > \Rightarrow < S_1'$ `par` $S_2, \sigma' >$

¡$S_2, \sigma > \Rightarrow \sigma' < S_1$ `par` $S_2, \sigma > \Rightarrow < S_1, \sigma' >$

¡$S_2, \sigma > \Rightarrow < S_2', \sigma' > < S_1$ `par` $S_2, \sigma > \Rightarrow < S_1$ `par` $S_2', \sigma' >$

Blocks

S x := a | . . . | `begin` D_V S `end`
D$_V$`var` $x := a$; $D_V \mid \epsilon$

```
begin
   var y:= 1;
   x:=1;
   begin
     var x:=2;
     y:=x+1
   end
   x:=x+y
end
```

Blocks and declarations

¡D$_V, \sigma > \to_D \sigma' \qquad < S, \sigma' > \to \sigma'' <$ `begin` D_V S `end`$, \sigma > \to$
$\sigma''[DV(D_V) \mapsto \sigma][block_{ns}]$

$< \epsilon, \sigma > \to_D \sigma \quad [empty_{ns}]$
¡D$_V, \sigma[x \mapsto \mathcal{A}[a]\sigma] > \to_D \sigma' <$ `var` $x := a;\ D_V, \sigma > \to_D \sigma'[var_{ns}]$

and $[DV(D_V) \mapsto \sigma]$ restores the values of the variables in $DV(D_V)$.

Procedures

S . . . | `begin` D_V D_P S `end` | `call` p
D$_V$`var` $x := a$; D_V | ϵ
D$_P$`proc` p `is` S; D_P | ϵ

```
begin
   var x:= 0;
   proc p is x:=x*2;
   proc q is call p;
   begin
     var x:=5;
     proc p is x:=x+1;
     call q;
     y:=x;
   end
end
```

Scope rules

1. dynamic scope for variables and procedures
2. dynamic scope for variables but static for procedures
3. (static scope for variables but dynamic for procedures)
4. static scope for variables and procedures

Different scope rules

```
begin
   var x:= 0;
   proc p is x:=x+2;
   proc q is call p;
   begin
     var x:=5;
     proc p is x:=x+1;
     call q;
     y:=x;
   end
end
```

variables	procedures	y=
dynamic scope	dynamic scope	6
dynamic scope	static scope	10
static scope	static scope	5

Procedure environments

$$Env_P = Pname \rightarrow Stm$$

A procedure environment maps procedure names to procedure bodies.

Transition with procedure environment

$$env_P \vdash < S, \sigma > \rightarrow \sigma'$$

In the procedure environment env_P the statement S transforms σ into σ'.

Procedure semantics, dynamic scope rules

$$env_P \vdash < x := a, \sigma > \rightarrow \sigma[x \mapsto \mathcal{A}[a]\sigma]$$

$$env_P \vdash < \texttt{skip}, \sigma > \rightarrow \sigma$$

$$env_P \vdash < S_1, \sigma > \rightarrow \sigma' \qquad env_P \vdash < S_2, \sigma' > \rightarrow \sigma'' env_P \vdash < S_1; S_2, \sigma$$

Procedure semantics, dynamic scope rules

The rules for `if` and `while` are also unaffected by the procedure environment.

Procedure semantics, dynamic scope rules

$$< D_V, \sigma > \rightarrow_D \sigma' \qquad upd_p(D_P, env_P) \vdash < S, \sigma' > \rightarrow \sigma'' env_P \vdash < \texttt{be}$$

where $upd_p(D_P, env_P)$ is env_P updated with the definitions in D_P.

Updating the environment

$$upd_P(\epsilon, env_P) = env_p$$

$$upd_P(\texttt{proc}\ \ p\ \ \texttt{is}\ \ S;\ \ D_P, env_P) = upd_P(D_P, env_p[p \mapsto S])$$

Procedure semantics, dynamic scope rules

$$env_P \vdash < S, \sigma > \longrightarrow \sigma' env_P \vdash < \texttt{call}\ \ p, \sigma > \longrightarrow \sigma'$$

where $S = env_P\ p$

Static scope rules for procedures

Modified procedure environment

$$upd_P(\epsilon, env_P) = env_p$$

$$upd_P(\texttt{proc}\ \ p\ \ \texttt{is}\ \ S;\ \ D_P, env_P) = upd_P(D_P, env_p[p \mapsto (S, env_P)])$$

Non-recursive procedures

$$env'_P \vdash < S, \sigma > \longrightarrow \sigma' env_P \vdash < \texttt{call}\ \ p, \sigma > \longrightarrow \sigma'$$

where $(S, env'_P) = env_P\ p$

Static scope rules for variables and procedures

$$\begin{aligned} Loc &= \\ Env_V &= Var \rightarrow Loc \\ Store &= Loc \cup \mathrm{next} \rightarrow \\ new &\in Loc \rightarrow Loc \end{aligned}$$

Static scope rules for variables and procedures

$$env_V,\ env_P \vdash < x := a, sto > \longrightarrow sto[\ell \mapsto v]$$

where $\ell = env_V\ x$ and $v = \mathcal{A}[a](sto \circ env_V)$

The rules for `skip`, ;, `if` and `while` are essentially unchanged.

Static scope rules for variables and procedures

¡D$_V, env_V, sto > \longrightarrow_D (env'_V, sto') \qquad env'_V,\ env'_P \vdash < S, sto' > \longrightarrow$
$sto'' env_V,\ env_P \vdash < begin\ D_V\ D_P\ S\ end, sto > \longrightarrow sto''$

where $env'_P = upd_P(D_P, env'_V, env_P)$

Static scope rules for variables and procedures

env$'_V, env'_P[p \mapsto (S, env'_V, env'_P)] \vdash < S, sto > \longrightarrow$
$sto' env_V,\ env_P \vdash < call\ p, sto > \longrightarrow sto'$

where $(S, env'_V, env'_P) = env_P\ p$