/SU/FSI/MASTER/INFO/MU4IN503
APS
Formulaire

P. Manoury*

Janvier 2022

5 APS3

5.1 Syntaxe

Lexique

Symboles réservés

`[ ] ( ) ; : , * ->`

Mots clef

`CONST FUN REC VAR PROC  ECHO SET IF WHILE CALL`
`RETURN`
`if`
`bool int vec`
`var adr`

Constantes numériques

num défini par *('-' ?)['0'-'9']+*

Identificateurs

ident défini par *(['a'-'z''A'-'Z'])(['a'-'z''A'-'Z''0'-'9'])**
dont on exclut les mots clef.

Remarque : les symboles d'opérateurs primitifs

`not and or eq lt  add sub mul div  alloc len nth`

sont des identificateurs.

Grammaire

Programme

Prog ::= Block

Bloc

Block ::= `[` Cmds `]`

Suite de commandes

Cmds ::= Stat
| Ret
| Def `;` Cmds
| Stat `;` Cmds

*Avec la précieuse relecture de W.S. et V.M. Qu'ils en soient remerciés.

Return

RET ::= `RETURN` EXPR

Définition

DEF ::= `CONST` ident TYPE EXPR
| `FUN` ident TYPE [ARGS] EXPR
| `FUN REC` ident TYPE [ARGS] EXPR
| `VAR` ident STYPE
| `PROC` ident [ARGSP] BLOCK
| `PROC REC` ident [ARGSP] BLOCK
| `FUN` ident TYPE [ARGSP] BLOCK
| `FUN REC` ident TYPE [ARGSP] BLOCK

Type

TYPE ::= STYPE
| (TYPES -> TYPE)
TYPES ::= TYPE
| TYPE * TYPES

SType

STYPE ::= `bool` | `int`
| (`vec` STYPE)

Paramètre formel (fonctions pures)

ARGS ::= ARG
| ARG , ARGS
ARG ::= ident : TYPE

Paramètre formel (procédure et fonctions procédurales)

ARGSP ::= ::= ARGP
| ARGP , ARGSP
ARGP ::= ident : TYPE
| `var ident` : TYPE

Instruction

STAT ::= `ECHO` EXPR
| `SET` LVALUE EXPR
| `IF` EXPR BLOCK BLOCK
| `WHILE` EXPR BLOCK
| `CALL` ident EXPRSP

lvalue

LVALUE ::= ident
| (`nth` LVALUE EXPR)

Paramètres d'appel

EXPRSP ::= EXPRP
| EXPRP EXPRSP
EXPRP ::= EXPR
| `(adr LVALUE)`

Expression

EXPR ::= num
| ident
| (`if` EXPR EXPR EXPR)
| (EXPR EXPRSP)
| [ARGS] EXPR

Suite d'expressions

$$\begin{array}{lcl} \textsc{Exprs} & ::= & \textsc{Expr} \\ & | & \textsc{Expr}\ \textsc{Exprs} \end{array}$$

5.2 Typage

$$\begin{array}{lcl} t & ::= & \texttt{bool}|\texttt{int}|\texttt{void}|(\texttt{vec}\ t)|t+\texttt{void}|ts|(ts)\texttt{->}t \\ ts & ::= & t|t\texttt{*}ts \end{array}$$

Soit $p1, \ldots, p_n \in \textsc{Exprp}$.
Posons $A([p_1 : t_1, \ldots, x_n : t_n]) = [x_1 : t'_1, \ldots, x_n : t'_n]$ avec
$t'_i = \begin{cases} t_i & \text{si } p_i = x_i \\ (\texttt{ref}\ t_i) & \text{si } p_i = \texttt{var}\ x_i \end{cases}$

Programmes $\vdash p : \texttt{void}$

(PROG) si $\Gamma_0 \vdash_{\textsc{Block}} bk : \texttt{void}$
alors $\vdash bk : \texttt{void}$

Blocs $\Gamma \vdash_{\textsc{Block}} bk : t$

(BLOC) si $\Gamma \vdash_{\textsc{Cmds}} cs : t$
alors $\Gamma \vdash_{\textsc{Block}} \texttt{[}cs\texttt{]} : t$

Suite de commandes $\Gamma \vdash_{\textsc{Cmds}} cs : t$

(DEF) si $d \in \textsc{Dec}$, si $\Gamma \vdash_{\textsc{Def}} d : \Gamma'$, si $\Gamma' \vdash_{\textsc{Cmds}} cs : t$
alors $\Gamma \vdash_{\textsc{Cmds}} (d\texttt{;}cs) : t$.

(STAT0) pour tout type t, si $\Gamma \vdash_{\textsc{Stat}} s : \texttt{void}$ et $\Gamma \vdash_{\textsc{Cmds}} cs : t$ alors $\Gamma \vdash_{\textsc{Cmds}} (s\texttt{;}cs) : t$

(STAT1) si $t \neq \texttt{void}$, si $\Gamma \vdash_{\textsc{Stat}} s : t + \texttt{void}$ et $\Gamma \vdash_{\textsc{Cmds}} cs : t$ alors $\Gamma \vdash_{\textsc{Cmds}} (s\texttt{;}cs) : t$

(RET) Si $\Gamma \vdash_{\textsc{Expr}} e : t$ alors $\Gamma \vdash_{\textsc{Cmds}} (\texttt{RETURN}\ e) : t$

(END) si $\Gamma \vdash_{\textsc{Stat}} s : t$ alors $\Gamma \vdash_{\textsc{Cmds}} (s) : t$

Définitions $\Gamma \vdash_{\textsc{Def}} d : \Gamma'$

(CONST) si $\Gamma \vdash_{\textsc{Expr}} e : t$
alors $\Gamma \vdash_{\textsc{Def}} (\texttt{CONST}\ x\ t\ e) : \Gamma[x : t]$

(FUN) si $\Gamma[x_1 : t_1; \ldots; x_n : t_n] \vdash_{\textsc{Expr}} e : t$
alors $\Gamma \vdash_{\textsc{Def}} (\texttt{FUN}\ x\ t\ \texttt{[}x_1\texttt{:}t_1\texttt{,}\ldots\texttt{,}x_n\texttt{:}t_n\texttt{]}\ e) : \Gamma[x : (t_1\ \texttt{*}\ \ldots\ \texttt{*}\ t_n\ \texttt{->}\ t)]$

(FUNREC) si $\Gamma[x_1 : t_1; \ldots; x_n : t_n; x : t_1\ \texttt{*}\ \ldots\ \texttt{*}\ t_n\ \texttt{->}\ t] \vdash_{\textsc{Expr}} e : t$
alors $\Gamma \vdash_{\textsc{Def}} (\texttt{FUN REC}\ x\ t\ \texttt{[}x_1\texttt{:}t_1\texttt{,}\ldots\texttt{,}x_n\texttt{:}t_n\texttt{]}\ e) : \Gamma[x : t_1\ \texttt{*}\ \ldots\ \texttt{*}\ t_n\ \texttt{->}\ t]$

(VAR) si $t \in \{\texttt{int}, \texttt{bool}\}$
alors $\Gamma \vdash_{\textsc{Def}} (\texttt{VAR}\ x\ t) : \Gamma[x : (\texttt{ref}\ t)]$

(PROC) si $A([p_1 : t_1, \ldots, p_n : t_n]) = [x_1 : t'_1, \ldots, x_n : t'_n]$
si $\Gamma[x_1 : t'_1; \ldots; x_n : t'_n] \vdash_{\textsc{Block}} bk : \texttt{void}$
alors $\Gamma \vdash_{\textsc{Def}} (\texttt{PROC}\ x\ \texttt{[}p_1\texttt{:}t_1, \ldots, p_n\texttt{:}t_n\texttt{]}bk) : \Gamma[x : t'_1\ \texttt{*}\ \ldots\ \texttt{*}\ t'_n\ \texttt{->}\ \texttt{void}]$

(PROCREC)
si $A([p_1 : t_1, \ldots, x_n : t_n]) = [x_1 : t'_1, \ldots, x_n : t'_n]$
si $\Gamma[x_1 : t'_1; \ldots; x_n : t'_n; x : t'_1\ \texttt{*}\ \ldots\ \texttt{*}\ t'_n\ \texttt{->}\ \texttt{void}] \vdash_{\textsc{Block}} bk : \texttt{void}$
alors $\Gamma \vdash_{\textsc{Def}} (\texttt{PROC REC}\ x\ \texttt{[}p_1\texttt{:}t_1, \ldots, p_n\texttt{:}t_n\texttt{]}bk) : \Gamma[x : t'_1\ \texttt{*}\ \ldots\ \texttt{*}\ t'_n\ \texttt{->}\ \texttt{void}]$

(FUNP) si $\Gamma[x_1 : t_1; \ldots; x_n : t_n] \vdash_{\text{Block}} bk : t$
alors $\Gamma \vdash_{\text{Def}} (\texttt{FUN}\ x\ t\ \texttt{[}x_1\texttt{:}t_1\texttt{, ... ,}x_n\texttt{:}t_n\texttt{]}\ bk) : \Gamma[x : \texttt{(}t_1\ \texttt{*}\ \ldots\ \texttt{*}\ t_n\ \texttt{->}\ t\texttt{)}]$

(FUNRECP) si $\Gamma[x_1 : t_1; \ldots; x_n : t_n; x : t_1\ \texttt{*}\ \ldots\ \texttt{*}\ t_n\ \texttt{->}\ t] \vdash_{\text{Block}} bk : t$
alors $\Gamma \vdash_{\text{Def}} (\texttt{FUN REC}\ x\ t\ \texttt{[}x_1\texttt{:}t_1\texttt{, ... ,}x_n\texttt{:}t_n\texttt{]}\ bk) : \Gamma[x : t_1\ \texttt{*}\ \ldots\ \texttt{*}\ t_n\ \texttt{->}\ t]$

Instructions $\Gamma \vdash_{\text{Stat}} s : t$

(ECHO) si $\Gamma \vdash_{\text{Expr}} e : \texttt{int}$
alors $\Gamma \vdash_{\text{Stat}} (\texttt{ECHO}\ e) : \texttt{void}$

(SET) si $\Gamma \vdash_{\text{Lval}} x : t$ et si $\Gamma \vdash_{\text{Expr}} e : t$
alors $\Gamma \vdash_{\text{Stat}} (\texttt{SET}\ x\ e) : \texttt{void}$

(IF0) pour tout type t, si $\Gamma \vdash_{\text{Expr}} e : \texttt{bool}$ et $\Gamma \vdash_{\text{Block}} blk_1 : t$ et $\Gamma \vdash_{\text{Block}} blk_2 : t$
alors $\Gamma \vdash_{\text{Stat}} (\texttt{IF}\ e\ blk_1\ blk_2) : t$

(IF1) pour tout $t \neq \texttt{void}$, si $\Gamma \vdash_{\text{Expr}} e : \texttt{bool}$ et $\Gamma \vdash_{\text{Block}} blk_1 : \texttt{void}$ et $\Gamma \vdash_{\text{Block}} blk_2 : t$
alors $\Gamma \vdash_{\text{Stat}} (\texttt{IF}\ e\ blk_1\ blk_2) : t + \texttt{void}$

(IF2) pour tout $t \neq \texttt{void}$, si $\Gamma \vdash_{\text{Expr}} e : \texttt{bool}$ et $\Gamma \vdash_{\text{Block}} blk_1 : t$ et $\Gamma \vdash_{\text{Block}} blk_2 : \texttt{void}$
alors $\Gamma \vdash_{\text{Stat}} (\texttt{IF}\ e\ blk_1\ blk_2) : t + \texttt{void}$

(WHILE) si $\Gamma \vdash_{\text{Expr}} e : \texttt{bool}$, si $\Gamma \vdash_{\text{Block}} bk : t$
alors $\Gamma \vdash_{\text{Stat}} (\texttt{WHILE}\ e\ bk) : t \oplus \texttt{void}$

(CALL) si $\Gamma(x) = t_1\ \texttt{*}\ \ldots\ \texttt{*}\ t_n\ \texttt{->}\ \texttt{void}$, si $\Gamma \vdash_{\text{Expar}} e_1 : t_1, \ldots$et si $\Gamma \vdash_{\text{Expar}} e_n : t_n$
alors $\Gamma \vdash_{\text{Stat}} (\texttt{CALL}\ x\ e_1 \ldots e_n) : \texttt{void}$

lvalue $\Gamma \vdash_{\text{Lval}} lv : t$

(LVAR) si $\Gamma(x) = \texttt{(ref}\ t\texttt{)}$
alors $\Gamma \vdash_{\text{Lval}} x : t$

(LNTH) si $\Gamma \vdash_{\text{Expr}} e_1 : \texttt{(vec}\ t\texttt{)}$ et $\Gamma \vdash_{\text{Expr}} e_2 : \texttt{int}$
alors $\Gamma \vdash_{\text{Lval}} \texttt{(nth}\ e_1\ e_2\texttt{)} : t$

Paramètres d'appel $\Gamma \vdash_{\text{Expar}} p : t$

(REF) si $\Gamma(x) = (\texttt{ref}\ t)$
alors $\Gamma \vdash_{\text{Expar}} \texttt{(adr}\ x\texttt{)} : (\texttt{ref}\ t)$

(VAL) si $e \in \text{Expr}$, si $\Gamma \vdash_{\text{Expr}} e : t$
alors $\Gamma \vdash_{\text{Expar}} e : t$

Expressions

(NUM) si $n \in \texttt{num}$
alors $\Gamma \vdash_{\text{Expr}} n : \texttt{int}$

(IDV) si $x \in \mathsf{ident}$, si $\Gamma(x) = t$ avec $t \neq (\texttt{ref}\ t')$
alors $\Gamma \vdash_{\text{Expr}} x : t$

(IDR) si $x \in \mathsf{ident}$,
si $\Gamma(x) = (\texttt{ref}\ t)$
alors $\Gamma \vdash_{\text{Expr}} x : t$

(IF) si $\Gamma \vdash_{\text{Expr}} e_1 : \texttt{bool}$, si $\Gamma \vdash_{\text{Expr}} e_2 : t$, si $\Gamma \vdash_{\text{Expr}} e_3 : t$
alors $\Gamma \vdash_{\text{Expr}} \texttt{(if}\ e_1\ e_2\ e_3\texttt{)} : t$

(APP) si $\Gamma \vdash_{\text{Expr}} e : \texttt{(}t_1\ \texttt{*}\ \ldots\ \texttt{*}\ t_n\ \texttt{->}\ t\texttt{)}$,
si $\Gamma \vdash_{\text{Expr}} e_1 : t_1, \ldots$, si $\Gamma \vdash_{\text{Expr}} e_n : t_n$
alors $\Gamma \vdash_{\text{Expr}} \texttt{(}e\ e_1 \ldots e_n\texttt{)} : t$

(ABS) si $\Gamma[x_1 : t_1; \ldots ; x_n : t_n] \vdash_{\text{Expr}} e : t$
alors $\Gamma \vdash_{\text{Expr}}$ `[`$x_1 : t_1, \ldots , x_n : t_n$`]`$e :$ `(`t_1 `* ... *` t_n `->` t`)`

(ALLOC) si $\Gamma \vdash_{\text{Expr}} e :$ `int`
alors $\Gamma \vdash_{\text{Expr}}$ `(alloc` e`)` : `(vec` t`)`

(LEN) si $\Gamma \vdash_{\text{Expr}} e :$ `vec` t)
alors $\Gamma \vdash_{\text{Expr}}$ `(len` e`)` : `int`

(NTH) si $\Gamma \vdash_{\text{Expr}} e_1 :$ `vec` t) et si $\Gamma \vdash_{\text{Expr}} e_2 :$ `int`
alors $\Gamma \vdash_{\text{Expr}}$ `(nth` e_1 e_2`)` : t

(VSET) si $\Gamma \vdash_{\text{Expr}} e_1 : (\text{vec } t)$, si $\Gamma \vdash_{\text{Expr}} e_1 : \text{int}$ et si $\Gamma \vdash_{\text{Expr}} e_3 : t$ alors $\Gamma \vdash_{\text{Expr}} (\text{vset } e_1\ e_2\ e_3) : (\text{vec } t)$

5.3 Sémantique

Programmes $\vdash p \rightsquigarrow (\sigma, \omega)$

(PROG) si $\varepsilon, \varepsilon, \varepsilon \vdash_{\text{Block}} bk \rightsquigarrow (\varepsilon, \sigma, \omega)$ alors $\vdash bk \rightsquigarrow (\sigma, \omega)$

Blocs $\rho, \sigma, \omega \vdash_{\text{Block}} bk \rightsquigarrow (v, \sigma', \omega')$

(BLOCK) si $\rho, \sigma, \omega \vdash_{\text{Cmds}} (cs;\varepsilon) \rightsquigarrow (v, \sigma', \omega')$ alors $\rho, \sigma, \omega \vdash_{\text{Block}}$ `[`cs`]` $\rightsquigarrow (v, \sigma', \omega')$.

Suites de commandes $\rho, \sigma, \omega \vdash_{\text{Cmds}} cs \rightsquigarrow (v, \sigma', \omega')$

(DECS) si $\rho, \sigma, \omega \vdash_{\text{Def}} d \rightsquigarrow (\rho', \sigma', \omega')$ et si $\rho', \sigma', \omega' \vdash_{\text{Cmds}} cs \rightsquigarrow (v, \sigma'', \omega'')$
alors $\rho, \omega \vdash_{\text{Cmds}} (d;\ cs) \rightsquigarrow (v, \sigma'', \omega'')$

(STATS0) si $\rho, \sigma, \omega \vdash_{\text{Stat}} s \rightsquigarrow (\varepsilon, \sigma', \omega')$ et si $\rho, \sigma', \omega' \vdash_{\text{Cmds}} cs \rightsquigarrow (v, \sigma'', \omega'')$
alors $\rho, \sigma, \omega \vdash_{\text{Cmds}} (s;\ cs) \rightsquigarrow (v, \sigma'', \omega'')$

(STATS1) si $\rho, \sigma, \omega \vdash_{\text{Stat}} s \rightsquigarrow (v, \sigma', \omega')$ avec $v \neq \varepsilon$ alors $\rho, \sigma, \omega \vdash_{\text{Cmds}} (s;\ cs) \rightsquigarrow (v, \sigma', \omega')$

(RET) si $\rho, \sigma, \omega \vdash_{\text{Expr}} e \rightsquigarrow (v, \sigma', \omega')$ alors $\rho, \sigma, \omega \vdash_{\text{Cmds}}$ (`RETURN` e) $\rightsquigarrow (v, \sigma', \omega')$

(END) si $\rho, \sigma, \omega \vdash_{\text{Stat}} s \rightsquigarrow (v, \sigma', \omega')$ alors $\rho, \sigma, \omega \vdash_{\text{Cmds}} (s) \rightsquigarrow (\varepsilon, \sigma, \omega)$

Définitions $\rho, \sigma, \omega \vdash_{\text{Def}} d \rightsquigarrow (\rho', \sigma', \omega')$

(CONST) si $\rho, \sigma, \omega \vdash_{\text{Expr}} e \rightsquigarrow (v, \sigma', \omega')$ alors $\rho, \sigma, \omega \vdash_{\text{Def}}$ (`CONST` x t e) $\rightsquigarrow (\rho[x = v], \sigma', \omega')$

(FUN) $\rho, \sigma \vdash_{\text{Def}}$ (`FUN` x t `[`x_1`:`t_1`, ... ,`x_n`:`t_n`]` e) $\rightsquigarrow (\rho[x = inF(e, (x_1; \ldots ; x_n), \rho), \sigma)$

(FUNREC) $\rho, \sigma \vdash_{\text{Def}}$ (`FUN REC` x t `[`x_1`:`t_1`, ... ,`x_n`:`t_n`]` e)
$\rightsquigarrow (\rho[x = inFR(e, x, (x_1; \ldots ; x_n)\rho), \sigma)$

(VAR) si $alloc(\sigma) = (a, \sigma')$, avec $\sigma' = \sigma[a = any]$ et $a \notin \mathsf{dom}(\sigma)$
alors $\rho, \sigma, \omega \vdash_{\text{Def}}$ (`VAR` x t) $\rightsquigarrow (\rho[x = inA(a)], \sigma', \omega)$

(PROC) $\rho, \sigma \vdash_{\text{Def}}$ (`PROC` x t `[`x_1`:`t_1`, ...`x_n`:`t_n`]` bk) $\rightsquigarrow (\rho[x = inP(bk, (x_1; \ldots ; x_n), \rho)], \sigma)$

(PROCREC) $\rho, \sigma \vdash_{\text{Def}}$ (`PROC REC` x t `[`x_1`:`t_1`, ... ,`x_n`:`t_n`]`bk) $\rightsquigarrow (\rho[x = inPR(inP(bk, x, (x_1; \ldots ; x_n), \rho), \sigma)$

(FUNP) $\rho, \sigma, \omega \vdash_{\text{Def}}$ (`FUN` x t `[`p_1`:`t_1`, ...`p_n`:`t_n`]` bk)
$\rightsquigarrow (\rho[x = inP(bk, (x_1, \ldots , x_n), \rho)], \sigma, \omega)$

(FUNRECP) $\rho, \sigma, \omega \vdash_{\text{Def}}$ (`FUN REC` x t `[`x_1`:`t_1`, ... ,`x_n`:`t_n`]`bk)
$\rightsquigarrow (\rho[x = inPR(bk, x, (x_1, \ldots , x_n), \rho)], \sigma, \omega)$

Instructions $\rho, \sigma, \omega \vdash_{\text{Stat}} s \rightsquigarrow (v, \sigma', \omega')$

(SET) si $\rho, \sigma, \omega \vdash_{\text{Expr}} e_2 \rightsquigarrow (v, \sigma', \omega')$ et si $\rho, \sigma', \omega' \vdash_{\text{Lval}} e_1 \rightsquigarrow (a, \sigma'', \omega'')$
alors $\rho, \sigma, \omega \vdash_{\text{Stat}} (\texttt{SET}\ e_1\ e_2) \rightsquigarrow (\varepsilon, \sigma''[a := v], \omega'')$

(IF1) si $\rho, \sigma, \omega \vdash_{\text{Expr}} e \rightsquigarrow (inZ(1), \sigma', \omega')$ et si $\rho, \sigma', \omega' \vdash_{\text{Block}} bk_1 \rightsquigarrow (v, \sigma'', \omega'')$
alors $\rho, \sigma, \omega \vdash_{\text{Stat}} (\texttt{IF}\ e\ bk_1\ bk_2) \rightsquigarrow (v, \sigma'', \omega'')$

(IF0) si $\rho, \sigma, \omega \vdash_{\text{Expr}} e \rightsquigarrow (inZ(0), \sigma', \omega')$ et si $\rho, \sigma', \omega' \vdash_{\text{Block}} bk_2 \rightsquigarrow (v, \sigma'', \omega'')$
alors $\rho, \sigma, \omega \vdash_{\text{Stat}} (\texttt{IF}\ e\ bk_1\ bk_2) \rightsquigarrow (v, \sigma'', \omega'')$

(LOOP0) si $\rho, \sigma, \omega \vdash_{\text{Expr}} e \rightsquigarrow (inZ(0), \sigma', \omega')$ alors $\rho, \sigma, \omega \vdash_{\text{Stat}} (\texttt{WHILE}\ e\ bk) \rightsquigarrow (\varepsilon, \sigma', \omega')$

(LOOP1A) si $\rho, \sigma, \omega \vdash_{\text{Expr}} e \rightsquigarrow (inZ(1), \sigma', \omega')$, si $\rho, \sigma', \omega' \vdash_{\text{Block}} bk \rightsquigarrow (\varepsilon, \sigma'', \omega'')$
et si $\rho, \sigma'', \omega'' \vdash_{\text{Stat}} (\texttt{WHILE}\ e\ bk) \rightsquigarrow (v, \sigma''', \omega''')$
alors $\rho, \sigma, \omega \vdash_{\text{Stat}} (\texttt{WHILE}\ e\ bk) \rightsquigarrow (v, \sigma''', \omega''')$

(LOOP1B) si $\rho, \sigma, \omega \vdash_{\text{Expr}} e \rightsquigarrow (inZ(1), \sigma', \omega')$, si $\rho, \sigma', \omega' \vdash_{\text{Block}} bk \rightsquigarrow (v, \sigma'', \omega'')$ avec $v \neq \varepsilon$
alors $\rho, \sigma, \omega \vdash_{\text{Stat}} (\texttt{WHILE}\ e\ bk) \rightsquigarrow (v, \sigma'', \omega'')$

(CALL) si $\rho(x) = inP(bk, (x_1, \ldots, x_n), \rho')$,
si $\rho, \sigma, \omega \vdash_{\text{Expr}} (e_1, \sigma_1) \rightsquigarrow (v_1, \sigma_1, \omega_1)$, ..., si $\rho, \sigma_{n-1}, \omega_{n-1} \vdash_{\text{Expr}} e_n \rightsquigarrow (v_n, \sigma_n, \omega_n)$
et si $\rho'[x_1 = v_1, \ldots, x_n = v_n], \sigma_n, \omega_n \vdash_{\text{Block}} bk \rightsquigarrow (v, \sigma', \omega')$
alors $\rho, \sigma, \omega \vdash_{\text{Stat}} (\texttt{CALL}\ x\ e_1 \ldots e_n) \rightsquigarrow (v, \sigma', \omega')$

(CALLR) si $\rho(x) = inPR(bk, x, (x_1, \ldots, x_n), \rho')$,
si $\rho, \sigma, \omega \vdash_{\text{Expr}} e_1 \rightsquigarrow (v_1, \sigma_1, \omega_1)$, ..., si $\rho, \sigma_{n-1}, \omega_{n-1} \vdash_{\text{Expr}} e_n \rightsquigarrow (v_n, \sigma_n, \omega_n)$
et si $\rho'[x_1 = v_1, \ldots, x_n = v_n, x = inPR(bk, x, (x_1, \ldots, x_n), \rho')], \sigma_n, \omega_n \vdash_{\text{Block}} bk \rightsquigarrow (v, \sigma', \omega'))$
alors $\rho, \omega \vdash_{\text{Stat}} (\texttt{CALL}\ x\ e_1 \ldots e_n) \rightsquigarrow (v, \sigma', \omega')$

(ECHO) si $\rho, \sigma, \omega \vdash_{\text{Expr}} e \rightsquigarrow (inZ(n), \sigma', \omega')$ alors $\rho, \sigma, \omega \vdash_{\text{Stat}} (\texttt{ECHO}\ e) \rightsquigarrow (\varepsilon, \sigma', n \cdot \omega')$

lvalue $\rho, \sigma, \omega \vdash_{\text{Lval}} e \rightsquigarrow (a, \sigma', \omega')$

(LID) si $x \in ident$, si $\rho(x) = inA(a)$ alors $\rho, \sigma, \omega \vdash_{\text{Lval}} x \rightsquigarrow (a, \sigma, \omega)$

(LNTH1) si $x \in \mathsf{ident}$, si $\rho(x) = inB(a, n)$ et si $\rho, \sigma, \omega \vdash_{\text{Expr}} e \rightsquigarrow (inZ(i), \sigma', \omega')$
alors $\rho, \sigma, \omega \vdash_{\text{Lval}} (\text{nth}\ x\ e) \rightsquigarrow (a + i, \sigma', \omega')$

(LNTH2) si $\rho, \sigma, \omega \vdash_{\text{Lval}} e_1 \rightsquigarrow (a_1, \sigma', \omega')$ avec $\sigma'(a_1) = inB(a_2, \_)$ et si $\rho, \sigma', \omega' \vdash_{\text{Expr}} e \rightsquigarrow (inZ(i), \sigma'', \omega'')$
alors $\rho, \sigma, \omega \vdash_{\text{Lval}} (\text{nth}\ lv\ e) \rightsquigarrow (a_2 + i, \sigma'', \omega'')$

Paramètres d'appel $\rho, \sigma, \omega \vdash_{\text{Expar}} p \rightsquigarrow (u, \sigma', \omega')$

(REF) si $\rho(x) = inA(a)$ alors $\rho, \sigma, \omega \vdash_{\text{Expar}} (\texttt{adr}\ x) \rightsquigarrow (inA(a), \sigma, \omega)$

(VAL) si $\rho, \sigma, \omega \vdash_{\text{Expr}} e \rightsquigarrow (v, \sigma', \omega')$ alors $\rho, \sigma, \omega \vdash_{\text{Expar}} e \rightsquigarrow (v, \sigma', \omega')$

Expressions $\rho, \sigma, \omega \vdash_{\text{Expr}} e \rightsquigarrow (v, \sigma', \omega')$

(TRUE) $\rho, \sigma, \omega \vdash_{\text{Expr}} \texttt{true} \rightsquigarrow (inZ(1), \sigma, \omega)$

(FALSE) $\rho, \sigma, \omega \vdash_{\text{Expr}} \texttt{false} \rightsquigarrow (inZ(0), \sigma, \omega)$

(NUM) si $n \in \texttt{num}$ alors $\rho, \sigma, \omega \vdash_{\text{Expr}} n \rightsquigarrow (inZ(\nu(n)), \sigma, \omega)$

(ID1) si $x \in \mathsf{ident}$ et $\rho(x) = inA(a)$ alors $\rho, \sigma, \omega \vdash_{\text{Expr}} x \rightsquigarrow (inZ(\sigma(a)), \sigma, \omega)$

(ID2) si $x \in \mathsf{ident}$ et si $\rho(x) = v$ et $v \neq inA(a)$ alors $\rho, \sigma, \omega \vdash_{\text{Expr}} e \rightsquigarrow (v, \sigma, \omega)$

(PRIM1) si $\rho, \sigma, \omega \vdash_{\text{Expr}} e \rightsquigarrow (inZ(n), \sigma', \omega')$, et si $\pi_1(not)(n) = n'$
alors $\rho, \sigma, \omega \vdash_{\text{Expr}} (\texttt{not}\ e) \rightsquigarrow (inZ(n'), \sigma', \omega')$

(PRIM2) si $x \in \{\texttt{eq lt add sub mul div}\}$,
si $\rho, \sigma, \omega \vdash_{\text{Expr}} e_1 \rightsquigarrow (inZ(n_1), \sigma', \omega')$, si $\rho, \sigma', \omega' \vdash_{\text{Expr}} e_2 \rightsquigarrow (inZ(n_2), \sigma'', \omega'')$ et si $\pi_2(x)(n_1, n_2) = n$
alors $\rho, \sigma, \omega \vdash_{\text{Expr}} (x\ e_1 e_2) \rightsquigarrow (inZ(n), \sigma'', \omega'')$

(AND0) si $\rho, \sigma, \omega \vdash_{\text{Expr}} e_1 \rightsquigarrow (inZ(0), \sigma', \omega')$
alors $\rho, \sigma, \omega \vdash_{\text{Expr}}$ `(and` e_1 e_2`)` $\rightsquigarrow (inZ(0), \sigma', \omega')$.

(AND1) si $\rho, \sigma, \omega \vdash_{\text{Expr}} e_1 \rightsquigarrow (inZ(1), \sigma', \omega')$ et si $\rho, \sigma', \omega' \vdash_{\text{Expr}} e_2 \rightsquigarrow (v, \sigma'', \omega'')$
alors $\rho, \sigma, \omega \vdash_{\text{Expr}}$ `(and` e_1 e_2`)` $\rightsquigarrow (v, \sigma'', \omega'')$.

(OR1) si $\rho, \sigma, \omega \vdash_{\text{Expr}} e_1 \rightsquigarrow (inZ(1), \sigma', \omega')$
alors $\rho, \sigma, \omega \vdash_{\text{Expr}}$ `(or` e_1 e_2`)` $\rightsquigarrow (inZ(1), \sigma', \omega')$.

(OR0) si $\rho, \sigma, \omega \vdash_{\text{Expr}} e_1 \rightsquigarrow (inZ(0)\sigma', \omega')$ et si $\rho, \sigma', \omega' \vdash_{\text{Expr}} e_2 \rightsquigarrow (v, \sigma'', \omega'')$
alors $\rho, \sigma, \omega \vdash_{\text{Expr}}$ `(or` e_1 e_2`)` $\rightsquigarrow (v, \sigma'', \omega'')$.

(IF1) si $\rho, \sigma, \omega \vdash_{\text{Expr}} e_1 \rightsquigarrow (inZ(1), \sigma', \omega')$ et si $\rho, \sigma', \omega' \vdash_{\text{Expr}} e_2 \rightsquigarrow (v, \sigma'', \omega'')$
alors $\rho, \sigma, \omega \vdash_{\text{Expr}}$ `(if` e_1 e_2 e_3`)` $\rightsquigarrow (v, \sigma'', \omega'')$

(IF0) si $\rho, \sigma, \omega \vdash_{\text{Expr}} e_1 \rightsquigarrow (inZ(0), \sigma', \omega')$ et si $\rho, \sigma', \omega' \vdash_{\text{Expr}} e_3 \rightsquigarrow (v, \sigma'', \omega'')$
alors $\rho, \sigma, \omega \vdash_{\text{Expr}}$ `(if` e_1 e_2 e_3`)` $\rightsquigarrow (v, \sigma'', \omega'')$

(ABS) $\rho, \sigma \vdash_{\text{Expr}}$ `[`x_1`:`t_1`,` $\dots$ `,`x_n`:`t_n`]`$e \rightsquigarrow (inF(e, (x_1, \dots, x_n), \rho), \sigma)$

(APP) si $\rho, \sigma \vdash_{\text{Expr}} e \rightsquigarrow inF(e', (x_1; \dots; x_n), \rho')$, si $\rho, \sigma \vdash_{\text{Expr}} e_1 \rightsquigarrow v_1$, …, si $\rho, \sigma \vdash_{\text{Expr}} e_n \rightsquigarrow v_n$,
si $\rho'[x_1 = v_1; \dots; x_n = v_n], \sigma \vdash_{\text{Expr}} e' \rightsquigarrow v$
alors $\rho, \sigma \vdash$ `(`e $e_1 \dots e_n$`)` $\rightsquigarrow v$

(APPR) si $\rho, \sigma \vdash_{\text{Expr}} e \rightsquigarrow inFR(e', x, (x_1; \dots; x_n), \rho')$,
si $\rho, \sigma \vdash_{\text{Expr}} e_1 \rightsquigarrow v_1$, …, si $\rho, \sigma \vdash_{\text{Expr}} e_n \rightsquigarrow v_n$,
si $\rho'[x_1 = v_1; \dots; x_n = v_n][x = inFR(e', x, (x_1; \dots; x_n), \rho')], \sigma \vdash_{\text{Expr}} e' \rightsquigarrow v$
alors $\rho, \sigma \vdash_{\text{Expr}}$ `(`e $e_1 \dots e_n$`)` $\rightsquigarrow v$

(ALLOC) si $\rho, \sigma, \omega \vdash_{\text{Expr}} e \rightsquigarrow (inZ(n), \sigma', \omega')$, avec $n > 0$,
et si $allocb(\sigma', n) = (a, \sigma'')$, avec $\sigma'' = \sigma'[a = inZ(n)]$,
alors $\rho, \sigma, \omega \vdash_{\text{Expr}}$ `(alloc` e`)` $\rightsquigarrow (inB(a), \sigma'', \omega')$

(LEN) si $\rho, \sigma, \omega \vdash_{\text{Expr}} e \rightsquigarrow (inB(a), \sigma', \omega')$
alors $\rho, \sigma, \omega \vdash_{\text{Expr}}$ `(len` e`)` $\rightsquigarrow (\sigma'(a), \sigma', \omega')$

(NTH) si $\rho, \sigma, \omega \vdash_{\text{Expr}} e_1 \rightsquigarrow (inB(a), \sigma', \omega')$ et si $\rho, \sigma', \omega' \vdash_{\text{Expr}} e_2 \rightsquigarrow (inZ(i), \sigma'', \omega'')$
alors $\rho, \sigma, \omega \vdash_{\text{Expr}}$ `(nth` e_1 e_2`)` $\rightsquigarrow (\sigma''(a + i + 1), \sigma'', \omega'')$

(VSET) si $\rho, \sigma, \omega \vdash_{\text{Expr}} e_1 \rightsquigarrow (inB(a, n), \sigma', \omega')$, si $\rho, \sigma', \omega' \vdash_{\text{Expr}} e_2 \rightsquigarrow (inZ(i), \sigma'', \omega'')$ et si $\rho, \sigma'', \omega'' \vdash_{\text{Expr}}$ $e_3 \rightsquigarrow (v, \sigma''', \omega''')$
alors $\rho, \sigma, \omega \vdash_{\text{Expr}}$ (vset lv e_1 e_2) $\rightsquigarrow (inB(a, n), \sigma'''[a + i := v], \omega''')$

(AFP) si $x \in \mathsf{ident}$ et $\rho(x) = inP(bk, (x_1, \dots, x_n), \rho')$,
si $\rho, \sigma, \omega \vdash_{\text{Expar}} e_1 \rightsquigarrow (v_1, \sigma_1, \omega_1)$, …, si $\rho, \sigma_{n-1}, \omega_{n-1} \vdash_{\text{Expar}} e_n \rightsquigarrow (v_n, \sigma_n, \omega_n)$
si $\rho'[x_1 = v_1, \dots, x_n = v_n], \sigma_n, \omega_n \vdash_{\text{Block}} bk \rightsquigarrow (v, \sigma', \omega')$
alors $\rho, \sigma, \omega \vdash (x\ e_1 \dots e_n) \rightsquigarrow (v, \sigma', \omega')$

(AFPR) si $x \in \mathsf{ident}$ et $\rho(x) = inPR(bk, x, (x_1, \dots, x_n), \rho')$,
si $\rho, \sigma, \omega \vdash_{\text{Expar}} e_1 \rightsquigarrow (v_1, \sigma_1, \omega_1)$, …, si $\rho, \sigma_{n-1}, \omega_{n-1} \vdash_{\text{Expar}} e_n \rightsquigarrow (v_n, \sigma_n, \omega_n)$
et si $\rho'[x_1 = v_1, \dots, x_n = v_n, x = inPR(bk, x, (x_1, \dots, x_n), \rho')], \sigma_n, \omega_n \vdash_{\text{Block}} bk \rightsquigarrow (v, \sigma', \omega')$
alors $\rho, \sigma, \omega \vdash (x\ e_1 \dots e_n) \rightsquigarrow (v, \sigma', \omega')$