講　　座

ALGOL N について

(IV) Operations (つづき)

島　内　剛　一*

4.7 Elaboration

E を〈expression〉としよう. E の *elaboration* は e(E) と書かれ, これは, もう1つの operation k といっしょに, E の構成の段階に沿って, つぎのように回帰的に定義される:

e(E) は

core

if E≡〈identifier〉, then→Kidentifier, else→next;
if E≡〈go to statement〉, then→Kgotostatement, else→next;
if E≡〈block〉, then→Kblock, else→next;
if E≡〈closed expression〉, then→Kclosedexpression, else→next;
if E≡〈code〉, then→Kcode, else→next;
if E≡〈effect notaion〉, then→Keffectnotaion, else→next;
if E≡〈real notation〉, then→Krealnotation, else→next;
if E≡〈bits notation〉, then→Kbitsnotation, else→next;
if E≡〈string notation〉, then→Kstringnotation, else→next;
if E≡〈reference notation〉, then→Kreferencenotation, else→next;
if E≡〈array notation〉, then→Karraynotation, else→next;
if E≡〈structure notation〉, then→Kstructurenotation, else→next;
if E≡〈procedure notation〉, then→Kprocedurenotation, else→next;
if E≡〈array element〉, then→Karrayelement, else→next;
if E≡〈structure element〉, then→Kstructureelement, else→next;
if E≡〈procedure call〉, then→Kprocedurecall,(else⇒L0);

Kidentifier:
　let E≡"〈identifier〉 V";
　pragmatics V∈**V**. **end of pragmatics**
　if a(V)=**on** then→next, else⇒ L0;
　let Q←q(V);
　⇒Q;

Kgotostatement:
　let E≡"**go to** 〈identifier〉 L";
　pragmatics L∈**L**. **end of pragmatics**
　⇒L;

Kblock:
　let E≡"**begin let** 〈identifier〉 V_1 **be** 〈expression〉 F_1;
　　let 〈identifier〉 V_2 **be** 〈expression〉 F_2;
　　.....
　　let 〈identifier〉 V_m **be** 〈expression〉 F_m;
　　〈identifier〉 L_1^1: .... : 〈identifier〉 $L_{i_1}^1$: 〈expression〉 E_1;
　　〈identifier〉 L_1^2: .... : 〈identifier〉 $L_{i_2}^2$: 〈expression〉 E_2;
　　.....
　　〈identifier〉 L_1^n: .... : 〈identifier〉 $L_{i_n}^n$: 〈expression〉 E_n
　　end"
　with integers $m(\geq 1)$, $n(\geq 1)$, $i_1(\geq 0)$, $i_2(\geq 0)$, ..., $i_n(\geq 0)$;
　pragmatics V_1, ..., V_m∈**V**;
　　L_1^1, ..., $L_{i_1}^n$, ..., L_1^n, $L_{i_n}^n$∈**L**.

* 立教大学理学部

end of pragmatics

(a(V_j)←**off**;) for $j=1,2,\ldots,m$

(e(F_j) if Q→next, L⇒L;

a(V_j)←**on**;

q(V_j)←Q;) for $j=1,2,\ldots,m$

K111: e(E_1) if Q→next, L→K121;

K112: e(E_2) if Q→next, L→K121;

.....

K11n: e(E_n) if Q→next, L→K121;

⇒Q;

K121: if $i_1=0$ then→K122, else→next;

(if $L=L_k^1$ then→K111, else next;) for $k=1,2,\ldots,i_1$

K122: if $i_2=0$ then→K123, else→next;

(if $L=L_k^2$ then→K112, else→next;) for $k=1,2,\ldots,i_2$

.....

K12n: if $i_n=0$ then→K13, else→next;

(if $L=L_k^n$ then→K11n, else→next;) for $k=1,1,\ldots,i_n$

K13: ⇒L;

Kclosedexpression:

let E≡"(⟨expression⟩ F)";

e(F) if Q⇒Q, L⇒L;

Kcode:

let E≡"**code** ⟨structure donor⟩D ⟨primary⟩T: ⟨code body⟩X";

let F←"**structure** D";

e(F) if Q→next, L⇒L;

X(Q) if Q'⇒Q', L⇒L;

Keffectnotation:

let H←H0 [**effect**];

let W←done;

g(**Q**)→Q;

t(Q)←**effect**;

h(Q)←H;

p(H, W) if W'→next, L⇒L;

w(Q)←w';

⇒Q;

Krealnotation:

let E≡"**real** ⟨modifier⟩Y ⟨real donor⟩J";

if Y≡*empty* then→K21, else→next;

if Y≡"[ ]" then→K22, else→next;

let Y≡"[⟨expression⟩F]";

e(F) if Q→next, L⇒L;

pragmatics t(Q)=(**procedure** (**real**) **real**)

end of pragmatics

let H←k(w(Q));

→K23;

K21: let H←H0[**real**];

→K23;

K22: let H←H1[**real**];

K23: let J≡*empty* then→K24, else→next;

let W←w(J);

→K25;

K24: W←arbitrary real number;

K25: g(**Q**)⇒Q';

t(Q')←**real**;

h(Q')←H;

p(H, W) if W'→next, L⇒L;

w(Q')←W';

⇒Q';

Kbitsnotation:

let E≡"**bits** ⟨modifier⟩Y ⟨bits donor⟩J";

if Y≡*empty* then→K31, else→next;

if Y≡"[ ]" then→K32, else→next;

let Y≡"[⟨exprssion⟩F]";

e(F) if Q→next, L⇒L;

pragmatics t(Q)=(**procedure** (**bits**) **bits**)

end of pragmatics

let H←k(w(Q));

→K33;

K31: let H←H0[**bits**];

→K33;

K32: let H←H1[**bits**];

K33: if J≡*empty* then→K34, else→next;

let W←w(J);

→K35;

K34: let W←arbitrary bits-string;

K35: g(**Q**)⇒Q';

t(Q')←**bits**;

h(Q')←H;

p(H, W) if W'→next, L⇒L;

w(Q')←W';

⇒Q';

Kstringnotation:

let E≡"**string** ⟨modifiire⟩Y ⟨string donor⟩J";

if Y≡*empty* then→K41, else→next;

if $Y\equiv$"[ ]" then→K42, else→next;
let $Y\equiv$"[⟨expression⟩F]";
e(F) if Q→next, $L\Rightarrow L$;
pragmatics t(Q)=(**procedure (string) string**)
end of pragmatics
let $H\leftarrow$k(w(Q));
→K43;

K41: let $H\leftarrow$H0[**string**];
→K43;

K42: let $H\leftarrow$H1[**string**];

K43: if $J\equiv$*empty* then→K44, else→next;
let $W\leftarrow$w(J);
→K45;

K44: let $W\leftarrow$arbitrary character-string;

K45: g(**Q**)⇒Q';
t(Q')←**string**;
h(Q')←H;
p(H, W) if W'→next, $L\Rightarrow L$;
w(Q')←W';
⇒Q';

Kreferencenotation:
let $H\leftarrow$H0[**reference**];
let $W\leftarrow$arbitrary value of the *type* **reference**;
g(**Q**)⇒Q;
t(Q)←**reference**;
h(Q)←H;
p(H, W) if W'→next, $L\Rightarrow L$;
w(Q)←W';
⇒Q;

Karraynotation:
let $E=$"**array** ⟨array bound⟩Y ⟨array donor⟩J";
let $J\equiv$"(⟨expression⟩E_1,..., ⟨expression⟩E_n)"
($n\geq 1$);
if $Y\equiv$"[ ]" then→K51, else→next;
if $Y\equiv$"[⟨expression⟩F_2]" then→K52, else→next;
if $Y\equiv$"[⟨expression⟩F_1:]" then→K53, else→next;
let $Y\equiv$"[⟨expression⟩F_1: ⟨expression⟩F_2]";
e(F_1) if Q_1→next, $L\Rightarrow L$;
let $I_1\leftarrow$o(w(Q_1));
e(F_2) if Q_2→next, $L\Rightarrow L$;
let $I_2\leftarrow$o(w(Q_2));
→K55;

K51: let $I_1\leftarrow 1$;
let $I_2\leftarrow n$;
→K55;

K52: let $I_1\leftarrow 1$;
e(F_2) if Q_2→next, $L\Rightarrow L$;
let $I_2\leftarrow$o(w(Q_2));
→K55;

K53: e(F_1) if Q_1→next, $L\Rightarrow L$;
let $I_1\leftarrow$o(w(Q_1));
let $I_2\leftarrow I_1+n-1$;
→K55;

K55: let $m\leftarrow I_2-I_1+1$;
if $m\leq 0$ then→K58, else→next;
if $m>n$ then→K56, else→next;
(e(E_i) if Q_i'→next, $L\Rightarrow L$;) for $i=1, 2, \ldots, m$
→K57;

K56: (e(E_i) if Q_i'→next, $L\Rightarrow L$;) for $i=1, 2, \ldots, n$
(e(E_n) if Q_i'→next, $L\Rightarrow L$;) for $i=n+1, n+2, \ldots, m$

K57: let $W\leftarrow\{\langle I_1, Q_1'\rangle, \langle I_1+1, Q_2'\rangle, \ldots, \langle J_2, Q_m'\rangle\}$;
→K59;

K58: let $W\leftarrow\phi$;

K59: let $T\leftarrow$t(E_1);
pragmatics t(E_1)=t(E_2)=...=t(E_n)
end of pragmatics
let $T'\leftarrow$**array** [ ]T;
let $H\leftarrow$H0[T'];
g(**Q**)⇒Q;
t(Q)←T';
h(Q)←H;
p(H, W) if W'→next, $L\Rightarrow L$;
w(Q)←W';
⇒Q';

Kstructurenotation:
let $E\equiv$"**structure** (⟨selector⟩S_1 ⟨expression⟩E_1, ⟨selector⟩S_2 ⟨expression⟩E_2, ... ⟨selector⟩S_n ⟨expression⟩E_n)"
where $n\geq 0$;
if $n=0$ then→K61, else→next;
(e(E_i) if Q_i→next, $L\Rightarrow L$;) for $i=1, 2, \ldots, n$
let $W\leftarrow\{\langle S_1, Q_1\rangle, \langle S_2, Q_2\rangle, \ldots, \langle S_n, Q_n\rangle\}$;
(let $T_i\leftarrow$t(E_i);) $i=1, 2, \ldots, n$

let T←**structure** ($S_1T_1, S_2T_2, \ldots, S_nT_n$);
→K62;
K61: let W←ϕ;
let T←**structure** ( );
K62: let H←H0[T];
g(**Q**)⇒Q;
t(Q)←T;
h(Q)←H;
p(H, W) if W'→next, L⇒L;
w(Q)←W';
⇒Q;
Kprocedurenotation:
let E≡"**procedure** (⟨expression⟩$T_1, \ldots,$
⟨expression⟩T_n)
⟨primary⟩T ⟨procedure donor⟩J"
where $n \geq 0$;
if $J \equiv$ *empty* then→K71, else→next;
let W←E;
→K72;
K71: let F←arbitrary *legal program of the form*
"**begin let** ⟨identifier⟩V_1 **be** T_1;
.....
let ⟨identifier⟩V_n **be** T_n;
⟨primary⟩E' **end**"
where E' contains no ⟨mark⟩'s and is of the type T;
let W←"E: ($V_1, \ldots, V_n$)E'";
K72: let T'←t(E);
pragmatics T'=(**procedure** (t(T_1), ...
..., t(T_n)t(T))
end of pragmatics
let T'←H0[T];
g(**Q**)⇒Q;
t(Q)←T';
h(Q)←H;
p(H, W) if W'→next, L⇒L;
w(Q)←w(Q');
⇒Q;
Karrayelement:
let E≡"⟨secondary⟩F [⟨expression⟩E']";
let T←t(E);
pragmatics t(F)=**array** [ ]T, t(E')=**real**
end of pragmatics
e(F) if Q→next, L⇒L;
e(E') if Q'→next, L⇒L;
if w(Q)≠ϕ then→next, else⇒L0;
let w(Q)≡{⟨v, Q_v⟩, ⟨$v+1, Q_{v+1}$⟩, ..., ⟨u, Q_u⟩}
where v, u are integers ($u \geq v$), and Q_v, Q_{v+1}, ..., Q_u are quantities;
pragmatics t(Q_v)=t(Q_{v+1})=...=t(Q_u)=T
end of pragmatics
let i←o(w(Q'));
if $i \geq v$ then→next' else⇒L0;
if $i \leq u$ then⇒Q_i, else⇒L0;
Kstructureelement:
let E≡"⟨secondary⟩F [⟨selector⟩S]";
e(F) if Q→next, L⇒L;
pragmatics t(Q)∈**T[structure]**
end of pragmatics
if w(Q)≠ϕ then→next, else⇒L0;
let w(Q)≡{⟨S_1, Q_1⟩, ⟨S_2, Q_2⟩, ..., ⟨S_n, Q_n⟩}
where an integer $n>0$, and $S_1, S_2, \ldots, S_n$ are ⟨selector⟩'s,
and $Q_1, Q_2, \ldots, Q_n$ are quantities;
(if $S=S_i$ then⇒Q_i, else→next;) for $i=1, 2, \ldots, n$
(⇒L0;)
Kprocedurecall:
let E≡"⟨secondary⟩F(⟨expression⟩$E_1, \ldots,$
⟨expression⟩E_n)"
where $n \geq 0$;
e(F) if Q→next, L⇒L;
let w(Q)≡"**procedure** (⟨expression⟩T_1,
..., ⟨expression⟩T_n)⟨primary⟩T:
(⟨identifier⟩$V_1, \ldots,$ ⟨identifier⟩V_n)
⟨primary⟩F'";
pragmatics t(Q)=(**procedure**($T_1, \ldots, T_n$)T),
t(E_1)=$T_1, \ldots,$ t(E_n)=T_n, t(E)=T
end of pragmatics
(let E_i'←"(E_i)";) for $i=1, 2, \ldots, n$
s(F'; $E_1'/V_1, \ldots, E_n'/V_n$)⇒$F''$;
r(F'')⇒F''';
e(F''') if Q'⇒Q', L⇒L;
end of core

4.8 Projection の結合

T を type とし，F を (**procedure** (T)T)-type の ⟨expression⟩ としよう．F によって惹き起こされる projection は，

k(F)

と書かれ，つぎのように定義される：

W が *T-type* の *value* を表わしているとき，

p(k(F), W)

は，

core

g(**Q**)⇒Q;
t(Q)←T;
h(Q)←H0[T];
w(Q)←H0[T](W);
g(V)⇒V;
a(V)←**on**;
q(V)←Q;
let E←"F(V)";
e(E) if Q'→next, L⇒L;
W'←w(Q');
⇒W'

end of core

(昭和 47 年 3 月 29 日受付)