数理解析研究所講究録
第436巻 1981年 261-280

Natural Language as a Specefication Language for Programs

by

Kazuhiro Fuchi

Electrotechnical Laboratory

Abstract

Using an example it is discussed how to derive formally a formal specification of a program from a natural language description. This is done in a Montaque-like framework. As a next step a (logic) program is derived from this formal specification through logical transformations.

1. Introduction

Natural language would be one of the best condidates as a specification language for programs. However, many problems are still remained to be solved for practical use of natural language as a specification language. Here is picked up one problem of converting natural language descriptions into more formal ones.

In the following, one specific problem is treated as an example. The problem is to find k such that

k is the maximum length of an upsequence contained in A[0]...A[n-1] (1)

A detailed (and informal) description is found in Dijkstra (1980).

This problem can be expressed more formally in a first order logic form like

$\exists u.up(u,n,k)\ \&\ \forall j\ (\exists u.up(u,n,j) \rightarrow j \leq k)$ (2)

assuming up(u,n,ℓ) means that u is an upsequence of length ℓ contained in A[0]...A[n-1].

The problem here is how these two expressions (1) and (2) are related, or how (2) can be derived from (1).

Another problem will be how up(u,n,ℓ) is defined in a more formal way.

2. Framework

The method used here is a simplified version of Montague's theory (PTQ). PTQ is based on intensional logic, but here an ordinary lambda calculus is used. (Intensional logic is actually an extention of lambda calculus.)

Look at the following sentence.

every woman talks (3)

The corresponding logical expression should be

$\forall x\ (W(x) \rightarrow T(x))$ (4)

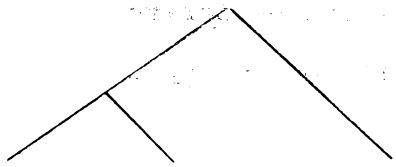
Here W and T are to stand for 'woman' and 'talk'. To see a correpondence between (3) and (4), let us try to apply lambda abstraction to (4).

$\forall x(W(x) \rightarrow T(x))$

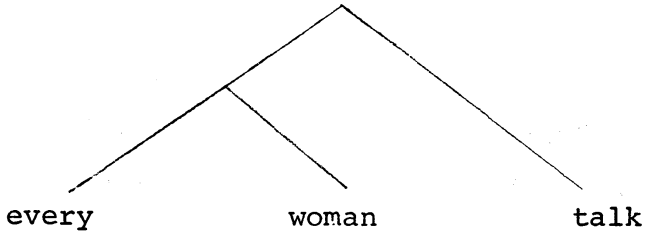
$\Leftarrow [\lambda Q.\forall x(W(x) \rightarrow Q(x))]\ (T)$

$\Leftarrow [[\lambda P \lambda Q.\forall x(P(x) \rightarrow Q(x)]\ (W)]\ (T)$

The structure of functional application of this expression can be illustrated in a tree form.



On the other hand the syntactic structure of (3) would be



These two structures can be overlapped. Then the three words in the sentence can be defined as follows.

woman ⇒ W

talk ⇒ T

every ⇒ $\lambda P \lambda Q . \forall x (P(x) \rightarrow Q(x))$

Conversely, representing the structure of (3) as

(every (woman)) (talk) (5)

then substituting a definition for each word and applying lambda conversion, the logical expression (4) can be obtained.

This approach is one of the main features of Montague's PTQ theory, which realizes Fregean functionality priciple.

Applying the method to

a woman talks ⟶ $\exists x (W(x) \& T(x))$

Mary talks ⟶ $T(m)$

we obtain the definitions of 'a' and 'Mary'.

Summarising,

every	Df	$\lambda P \lambda Q . \forall x (P(x) \rightarrow Q(x))$	(6)
a	Df	$\lambda p \lambda Q . \exists x (P(x) \& Q(x))$	(7)
Mary	Df	$\lambda P . P(m)$	(8)

The definitions of 'is' and 'the' are obtained from other sample sentences.

is	Df	$\lambda P \lambda x.P(\lambda y.x=y)$	(9)
the	Df	$\lambda P \lambda Q.Q(\iota x.P(x))$	(10)

Here $\iota x.P(x)$ (iota description) denotes x such that x exists uniquely and P(x) holds.

3. Application to the Example

Suppose $\underline{M}$ stands for 'maximum length of an upsequence contained in A[0]...A[n-1]'.

Then (1) can be written as

$\underline{k}$ ($\underline{is}$ ($\underline{the}$ ($\underline{M}$)))

assuming a proper syntactic analysis. $\underline{k}$ should be $\lambda P.P(k)$ like (8). Then applying a series of lambda conversions using (9) and (10),

$\underline{the}$ ($\underline{M}$) $\Rightarrow [\lambda P \lambda Q.Q(\iota x.P(x))](\underline{M})$

$\Rightarrow \lambda Q.Q(\iota x.\underline{M}(x))$

$\underline{is}$ ($\underline{the\ M}$) $\Rightarrow [\lambda P \lambda x.P(\lambda y.x=y)](\underline{the\ M})$

$\Rightarrow \lambda x.$ $\underline{the\ M}$ $(\lambda y.x=y)$

$\Rightarrow \lambda x.([\lambda Q.Q(\iota x'.\underline{M}(x'))](\lambda y.x=y))$

$\Rightarrow \lambda x.([\lambda y.x=y](\iota x'.\underline{M}(x')))$

$\Rightarrow \lambda x.\ x=\iota x'.\underline{M}(x')$

$\underline{k}(\underline{is\ the\ M})$ $\Rightarrow [\lambda P.P(k)](\underline{is\ the\ M})$

$\Rightarrow \underline{is\ the\ M}(k)$

$\Rightarrow [\lambda x.x=\iota x'.\underline{M}(x')](k)$

$\Rightarrow k = \iota x'.\underline{M}(x')$

From the meaning of iota

$\Rightarrow \underline{M}(k)$

On the other hand, lambda-abstracting (2),

$(2) \Leftarrow [\lambda \ell(\exists u.up(u,n,\ell)\ \&\ \forall j(\exists u.up(u,n,j) \rightarrow j \leq \ell))]\ (k)$

So, we can put

$\underline{M} \equiv \lambda \ell(\ldots)$

Applying lambda abstraction,

$$\underline{M} \Leftarrow [\lambda P \lambda \ell (P(\ell) \& \forall j (P(j) \rightarrow j \leq \ell))] \ (\lambda \ell . \exists u . up(u,n,\ell))$$

So, we obtain the definition of 'maximum'.

$$\underline{max} \quad \underline{\underline{Df}} \quad \lambda P \lambda \ell (P(\ell) \ \& \ \forall j \ (P(j) \rightarrow j \leq \ell)) \qquad (12)$$

This definition seems reasonable.

In this abstraction, we can think $\lambda \ell \ \exists u.up(u,n,\ell)$ as representing 'length of an...'. $up(u,n,\ell)$ can be rephrased as

$$up(u,n,\ell) \equiv up(u) \ \& \ in\text{-}n(u) \ \& \ len(u) = \ell$$

supposing 'up(u)' means u is an upsequence, 'in-n(u)' means u is contained in A[0]...A[n-1], and 'len(u)=ℓ' means the length of u is ℓ.

$$\lambda \ell \exists u (up(u) \ \& \ in\text{-}n(u) \ \& \ len(u) = \ell)$$
$$\Leftarrow \ \lambda \ell \exists u (up(u) \& \ in\text{-}n(u) \& \ [\lambda u . len(u) = \ell](u))$$
$$\Leftarrow \ \lambda \ell ([\lambda P \exists u (up(u) \& \ in\text{-}n(u) \& \ P(u))](\lambda u . len(u) = \ell))$$
$$\Leftarrow \ [\lambda Q \lambda \ell ([\lambda P \exists u (up(u) \& \ in\text{-}n(u) \& \ P(u))](Q(\ell)))]$$
$$(\lambda \ell \lambda u . len(u) = \ell)$$

The definition of 'length' can be thought reasonabley as

$$\underline{len} \quad \underline{\underline{Df}} \quad \lambda \ell \lambda u . len(u) = \ell \qquad (13)$$

The $\underline{M}$ can now be put

$$\underline{M} \equiv \underline{max}([\lambda Q \ldots](\underline{len}))$$

$$\lambda Q \lambda \ell ([\lambda P \exists u (up(u) \& \ in\text{-}n(u) \& \ P(u))](Q(\ell)))$$

$\Leftarrow$ [λPλQλℓ.P(Q(ℓ))] (λPΞu(up(u)& in-n(u)& P(u)))

$\Leftarrow$ [λPλQλℓ.P(Q(ℓ))] ([λPλQΞu(P(u)& Q(u))]
(λu(up(u)& in-n(u)))

If we use the definition of 'a' (7),

$\Leftarrow$ [λPλQλℓ.P(Q(ℓ))] (a (λu(up(u)& in-n(u)))

Nouns and adjectives are defined like

up	Df	λu.up(u)	(14)
in-n	Df	λPλu(P(u)& in-n(u))	(15)

So, now we have

$\Leftarrow$ [λPλQλℓ.P(Q(ℓ))] (a (in-n(up)))

The remaining definition is for 'of'.

of	Df	λPλQλx.P(Q(x))	(16)

M ≡ max ((of(a(in-n(up))))(len))

Summarizing we can state the problem as follows.

The original sentence (1) can be expressed by a proper syntactic analysis, as

k (is (the (max((len)(of (a ((up) in-n))))))) (17)

Noticing that of and in-n are post-fix operators, the normal form of (17) should be

k(is(the(max((of(a(in-n(up))))(len))))) (18)

Substituting the obtained definitions into (18), we can obtain (2) as a result by lambda conversions.

4. Some Observations

(i) The definition of 'of' is same as the combinator of functional application. 'of' is a very complex word linguistically, and presumably have many meanings. However the definition (16) can be thought as representing one (important) function of 'of'.

(ii) Notice the role of 'a' in the sentence (1). The existential quantifiers in (2) come from this 'a' in (1). Also notice that the definition of 'a' was obtained by quite a independent motivation, but works very well.

(iii) Let us look at another example.

weight of a woman (19)

This can be expressed as

(of(a(woman))) (weight) (20)

Let us define weight like len (13).

weight Df $\lambda w \lambda x.wt(x) = w$ (21)

Then

(17) $\Rightarrow \lambda w\ x\ (woman(x)\ \&\ wt(x) = w)$ (22)

Now let us consider a case of proper nouns.

Mary's weight (23)

The structure of (23) is

(of(Mary)) (weight) (24)

Using the definition (8)

of(Mary) ⇒ [λPλQ λx.P(Q(x))] (Mary)

⇒ λQλx.Mary(Q(x))

⇒ λQλx([λP.P(m)] (Q(x)))

⇒ λQλx([Q(x)] (m))

⇒ λQλw([Q(w)](m))

of Mary (weight)

⇒ [λQλw([Q(w)](m))](λwλx.wt(x)=w)

⇒ λw([[λwλx.wt(x)=w](w)](m))

⇒ λw([λx.wt(x)=w](m))

⇒ λw.wt(m)=w (25)

This seems to represent the meaning of (19) properly, and provides an evidence as for adequateness of the definition (16).

(iv) However there are cases the definition (16) does not apply. For example in case of

upsequence of length k (26)

the definition does not work.

The appropriate logical form for (26) would be

λu(up(u) & len(u)=k) (27)

For this case a more appropriate definition of 'of' should be

of'	Df	$\lambda P \lambda Q \lambda x\ (P(x)\ \&\ Q(x))$	(28)

The proper selection among the two definitions would be possible by type-checking.

Following (14) and (15)

important Df $\lambda Q \lambda x(imp(x)\ \&\ Q(x))$ (29)

importance Df $\lambda x.imp(x)$ (30)

Then,

of'(importance)

$\Rightarrow [\lambda P \lambda Q \lambda x(P(x)\ \&\ Q(x))]\ (\lambda x.imp(x))$

$\Rightarrow \lambda Q \lambda x(imp(x)\ \&\ Q(x))$ = important

This will be another motivation for the definition (28).

5. Derivation of a Logic Program

The formal description (2) can be used as a basis for derivation of a program.

5.1. Formal Specification

Let's define some predicates:

max(n)= k: k is the maximum length of an upsequence contained in A[0]...A[n-1]

up(u,n,ℓ): u is an upsequence of length ℓ contained in A[0]...A[n-1]

Then,

$$\max(n)=k \leftrightarrow \exists u.up(u,n,k) \,\&\, \forall j(\exists u.up(u,n,j) \rightarrow j \leq k) \qquad (31)$$

(31) can be taken as a formal specification for the problem. Starting from (31), we are going to derive a program. In order to do so, we need to know the properties of 'up'.

5.2. Some Knowledge

seq(s): s is a sequence

up(u): u is an upsequence

sub(s,s'): s is a subsequence of s'

len(s)=ℓ: ℓ is the length of s

r(s): the right-most element of s

s·a: a structure composed of s and a

$\bar{A}(n)$: the sequence A[0]·A[1]·....·A[n-1]

Assuming a is an atomic element,

$$\begin{cases} seq(a). \\ seq(s\cdot a) \leftarrow seq(s) \end{cases} \qquad \begin{cases} len(a)=1 \\ len(s\cdot a)=len(s)+1 \end{cases}$$

$$\begin{cases} r(a)=a \\ r(s\cdot a)=a \end{cases} \qquad \begin{cases} up(a). \\ up(u\cdot a)\leftarrow up(u)\ \&\ r(u)\leq a \end{cases}$$

$$\begin{cases} sub(a,a). \\ sub(a,s\cdot a). \\ sub(s,s'\cdot a)\leftarrow sub(s,s') \\ sub(s\cdot a,s'\cdot a)\leftarrow sub(s,s') \end{cases} \qquad \begin{cases} \bar{A}(1)=A[0] \\ \bar{A}(n+1)=\bar{A}(n)\cdot A[n] \end{cases}$$

$$up(u,n,\ell) \equiv up(u)\ \&\ sub(u,\bar{A}(n))\ \&\ len(u)=\ell$$

So,
$$\begin{cases} up(A[0],1,1). \\ up(A[n],n+1,1). \\ up(u,n+1,\ell)\leftarrow up(u,n,\ell) \\ up(u\cdot A[n],n+1,\ell+1)\leftarrow up(u,n,\ell)\ \&\ r(u)\leq A[n] \end{cases}$$

Upsequences are generated only by the above rules, so,

$$up(u,n+1,\ell) \leftrightarrow up(u,n,\ell) \vee (u=A[n]\ \&\ \ell=1) \vee \exists u'(up(u',n,\ell-1)\ \&\ r(u')\leq A[n]\ \&\ u=u'\cdot A[n]) \quad (32)$$

5.3. Mathematical Induction

The problem is equivalent to prove (constructively) that for all natural number u, $\exists k.max(n)=k$.

The problem itself and (32) suggest to use mathematical induction on n.

[step 1] n=1

$max(1)=k \Leftarrow k=1$

$\therefore \quad max(1)=1.$ (33)

[step 2] Assuming max(n)=k, that is,

$$\exists u.up(u,n,k) \tag{34}$$

and $\forall j(\exists u.up(a,n,j) \rightarrow j \leq k)$ (35)

it should be proved that max(n+1)=k',

i.e.

$$\exists u.up(u,n+1,k') \,\&\, \forall j(\exists u.up(u,n+1,j) \rightarrow j \leq k') \tag{36}$$

$\exists u.\ up(u,n+1,k')$ (using (32))

$\Leftarrow \exists u(up(u,n,k') \vee (u=A[n] \,\&\, k'=1) \vee$

$\exists u'(up(u',n,k'-1) \,\&\, r(u') \leq A[n] \,\&\, u=u' \cdot A[n]))$

$\Leftarrow \exists u.up(u,n,k') \vee \exists u(u=A[n] \,\&\, k'=1) \vee$

$\exists u'(up(u',n,k'-1) \,\&\, r(u') \leq A[n] \,\&\, \exists u.u=u' \cdot A[n])$

$\Leftarrow k'=k \vee k'=1 \vee k'=k+1 \,\&\, \exists u(up(u,n,k) \,\&\, r(u) \leq A[n])$

(using (34))

$\forall j(\exists u.up(u,n+1,j) \rightarrow j \leq k')$ (using (32))

$\Leftarrow \forall j(up(u,n,j) \rightarrow j \leq k') \,\&\, \forall j(u=A[n] \,\&\, j=1 \rightarrow j \leq k') \,\&$

$\forall j(\exists u'(up(u',n,j-1) \,\&\, r(u') \leq A[n] \,\&\, \exists u.u=u' \cdot A[n]) \rightarrow j \leq k')$

$\Leftarrow k \leq k' \,\&\, 1 \leq k' \,\&\, (\exists u(up(u,n,k) \,\&\, r(u) \leq A[n]) \rightarrow k+1 \leq k')$

(using (35))

(Writing Q for $\exists u(up(u,n,k) \,\&\, r(u) \leq A[n])$)

max(n+1)=k'

$\Leftarrow (k'=k \vee k'=1 \vee k'=k+1 \,\&\, Q) \,\&\, k \leq k' \,\&\, 1 \leq k' \,\&\, (Q \rightarrow k+1 \leq k')$

$\Leftarrow (k'=k \vee k'=k+1 \,\&\, Q) \,\&\, (\sim Q \vee k+1 \leq k')$

$\Leftarrow k'=k \,\&\, Q \vee k'=k+1 \,\&\, \sim Q$ (37)

($\Leftarrow$ true)

So, step 2 is proved. Summarizing,

$$\max(1)=1.$$

$$\max(n)=k \;\&\; \exists u(up(u,n,k)\;\&\; r(u)\leq A[n]) \rightarrow \max(n+1)=k+1.$$

$$\max(n)=k \;\&\; \sim \exists u(up(u,n,k)\;\&\; r(u)\leq A[n]) \rightarrow \max(n+1)=k. \qquad (38)$$

(38) can be seen a specification a little bit elaborated. It seems to be approaching to a program (algorithm). Note that the quantifier $\forall j$ was eliminated.

5.4. Minimum Element

It would be desirable to eliminate $\exists u$ from (38), so let's try to find m which is independent of particular u's, such that,

$$\exists u(up(u,n,k) \;\&\; r(u)\leq x) \leftrightarrow m\leq x \qquad (39)$$

$\exists u(up(u,n,k)\;\&\; r(u)\leq x) \rightarrow m\leq x$

$\Leftarrow$ $\forall u(up(u,n,k)\;\&\; r(u)\leq x \rightarrow m\leq x)$

$\Leftarrow$ $\forall u(up(u,n,k) \rightarrow x<r(u) \vee m\leq x)$

$\Leftarrow$ $\forall u(up(u,n,k) \rightarrow m\leq r(u))$

$\because$ $r(u)\leq x \;\&\; x<m \rightarrow r(u)<m$

$\therefore$ $m\leq r(u) \rightarrow x<r(u) \vee m\leq x$

$\exists u(up(u,n,k) \;\&\; r(u)\leq x) \leftarrow m\leq x$

$\Leftarrow$ $\exists u(m\leq x \rightarrow up(u,n,k) \;\&\; r(u)\leq x)$

$\Leftarrow$ $\exists u(up(u,n,k) \;\&\; (m\leq x \rightarrow r(u)\leq x))$

$\Leftarrow$ $\exists u(up(u,n,k) \;\&\; m=r(u))$

So, (39) $\Leftarrow$

$$\exists u(up(u,n,k) \;\&\; m=r(u)) \;\&\; \forall u(up(u,n,k) \rightarrow m\leq r(u)) \qquad (40)$$

(40) is intuitively seen as representing 'minimum'.

(40) can be written more exactly

$\forall n \forall k \exists m. \ldots\ldots .$

So 'Skolemizing' m, m=min(n,k)

$$\min(n,k)=m \leftrightarrow \exists u(up(u,n,k) \& m=r(u)) \& \forall u(up(u,n,k) \rightarrow m \leqq r(u)) \quad (40)'$$

If we have a program to compute min(n,k), then (38) can be rewritten as,

$$\max(1)=1.$$
$$\max(n)=k \ \& \ \min(n,k) \leqq A[n] \rightarrow \max(n+1)=k+1.$$
$$\max(n)=k \ \& \ A[n]<\min(n,k) \rightarrow \max(n+1)=k. \quad (41)$$

and, this is almost a program.

5.5. Mathmatical Induction Again

Looking at (40)', let's try to prove

$\forall n \forall j \exists m.\min(n,j)=m$

[in case of j=1]

$\exists u(up(u,n+1,1) \& r(u)=m')$

$\Leftarrow \quad \exists u(up(u,n,1) \& r(u)=m') \ \vee \ (u=A[n] \& r(u)=m')$

$\Leftarrow \quad m'=m_1 \ \vee \ m'=A[n]$

$\forall u(up(u,n+1,1) \rightarrow m' \leqq r(u))$

$\Leftarrow \quad \forall u(up(u,n,1) \rightarrow m' \leqq r(u)) \& (u=A[n] \rightarrow m' \leqq r(u))$

$\Leftarrow \quad m' \leqq m_1 \ \& \ m' \leqq A[n]$

.

$(m'=m_1 \ \vee \ m'=A[n]) \& m' \leqq m_1 \ \& \ m' \leqq A[n]$

$\Leftarrow \quad m'=m_1 \ \& \ m' \leqq A[n] \ \vee \ m'=A[n] \& m' \leqq m_1$

$\Leftarrow \quad m'=m_1 \ \& \ m_1 \leqq A[n] \ \vee \ m'=A[n] \ \& \ A[n] \leqq m_1$

[in case of $j=k+1$] where $\max(n)=k$

$\exists u(up(u,n+1,k+1)\ \&\ r(u)=m'$

$\Leftarrow\quad \exists u\exists u'(up(u',n,k)\ \&\ r(u')\leq A[n]$

$\&\ u=u'\cdot A[n]\ \&\ r(u)=m')$

$\Leftarrow\quad m_k\leq A[n]\ \&\ m'=r(u'\cdot A[n])$

$\Leftarrow\quad m_k\leq A[n]\ \&\ m'=A[n]$

$\forall u(up(u,n+1,k+1)\rightarrow m'\leq r(u))$

$\Leftarrow\quad \forall u\forall u'(up(u',n,k)\ \&\ r(u')\leq A[n]\ \&\ u=u'\cdot A[n]$

$\rightarrow m'\leq r(u))$

$\Leftarrow\quad m_k\leq A[n]\rightarrow m'\leq A[n]$

$m'=A[n]\ \&\ m_k\leq A[n]\ \&\ (m_k\leq A[n]\rightarrow m'\leq A[n])$

$\Leftarrow\quad m'=A[n]\ \&\ m_k\leq A[n]$

[in case of $1<j\leq k$]

$\exists u(up(u,n+1,j)\ \&\ r(u)=m')$

$\Leftarrow\quad \exists u(up(u,n,j)\ \&\ r(u)=m')\ \vee\ \exists u\exists u'(up(u',n,j-1)\ \&$

$r(u')\leq A[n]\ \&\ u=u'\cdot A[n]\ \&\ r(u)=m')$

$\Leftarrow\quad m'=m_j\ \vee\ m'=A[n]\ \&\ m_{j-1}\leq A[n]$

$\forall u(up(u,n+1,j)\rightarrow m'\leq r(u))$

$\Leftarrow\quad \forall u(up(u,n,j)\rightarrow m'\leq r(u))\ \&$

$\forall u\forall u'(up(u',n,j-1)\ \&\ r(u')\leq A[n]$

$\&\ u=u'\cdot A[n]\rightarrow m'\leq r(u))$

$\Leftarrow\quad m'\leq m_j\ \&\ (m_{j-1}\leq A[n]\rightarrow m'\leq A[n])$

$(m'=m_j\ \vee\ m'=A[n]\ \&\ m_{j-1}\leq A[n])\ \&\ m'\leq m_j$

$\&\ (m_{j-1}\leq A[n]\rightarrow m'\leq A[n])$

$\Leftarrow\quad m'=m_j\ \&\ m'\leq m_j\ \&\ (m_{j-1}\leq A[n]\rightarrow m'\leq A[n])$

$\vee\ m'=A[n]\ \&\ m_{j-1}\leq A[n]\ \&\ m'\leq m_j\ \&\ (m_{j-1}\leq A[n]\rightarrow m'\leq A[n])$

$\Leftarrow\quad m'=m_j\ \&\ (m_{j-1}\leq A[n]\rightarrow m_j\leq A[n])$

$$\begin{aligned} & \forall\, m'=A[n] \;\&\; m_{j-1}\leqq A[n]\leqq m_j \\ \Leftarrow\quad & m'=m_j \;\&\; \sim(m_{j-1}\leqq A[n]<m_j) \\ & \forall\, m'=A[n] \;\&\; m_{j-1}\leqq A[n]\leqq m_j \end{aligned}$$

Summarizing,

$$\left[\begin{array}{l} \min(1,1)=A[0] \\ \min(n,k)\leqq A[n] \rightarrow \min(n+1,k+1)=A[n] \\ A[n]<\min(n,1) \rightarrow \min(n+1,1)=A[n] \\ \min(n,j-1)\leqq A[n]<\min(n,j) \rightarrow \min(n+1,j)=A[n] \end{array}\right]$$

in other cases $\min(n+1,j)=\min(n,j)$

where $k=\max(n)$ and $j=1,\ldots,k$ (42)

Introducing Vector

min(n): a vector of min(n,j)

j=1,...,k(=max(n))

(∵ min(n)[j]=min(n,j))

<v[i]:=x>: a vector where

<v[i]: = x>[i]= x

<v[i]: = x>[j]=v[j] if i≠j

Using a vector notation, (12) can be written as

$$\begin{aligned} & \min(1)=<m_0[1]:=A[0]>. \\ & \min(n)=m \;\&\; \max(n)=k \;\& \\ & \qquad m[k]\leqq A[n] \rightarrow \min(n+1)=<m[k+1]:=A[n]>. \\ & \min(n)=m \;\&\; \max(n)=k \;\& \\ & \qquad A[n]<m[1] \rightarrow \min(n+1)=<m[1]:=A[n]>. \\ & \min(n)=m \;\&\; \max(n)=k \;\&\; m[1]\leqq A[n]<m[k] \;\& \\ & \qquad m[j-1]\leqq A[n]<m[j] \rightarrow \min(n+1)=<m[j]:=A[n]>. \end{aligned} \tag{43}$$

5.6. Merging Two Expressions

(41) & (43) are very similar in their structures, so it seems possible to merge them. Define

$$\max(n,k,m) \leftrightarrow \max(n)=k \ \&\ \min(n)=m$$

$$\max(N)=K$$

$$\begin{array}{l}
\max(1,1,\langle m_0[1]:=A[0]\rangle). \\
\max(n,k,m) \ \&\ n\neq N \ \& \\
\quad \left\{ \begin{array}{l} m[k]\leq A[n] \rightarrow \max(n+1,k+1,\langle m[k+1]:=A[n]\rangle), \\ A[n]<m[1] \rightarrow \max(n+1,k,\langle m[1]:=A[n]\rangle), \\ m[1]\leq A[n]<A[k] \ \&\ B(j) \\ \qquad\qquad \rightarrow \max(n+1,k,\langle m[j]:=A[n]\rangle) \end{array} \right\}. \\
\max(N,K,m) \rightarrow . \qquad\qquad (44)
\end{array}$$

where $B(j) \equiv m[j-1]\leq A[n]<m[j]$

Introducing,

$$b(i,j) \leftrightarrow m[i]\leq A[n]<m[j]$$

$B(j)$

$$\begin{array}{l}
b(1,k). \\
b(i,j) \ \&\ i\neq j-1 \ \&\ h=i+j \ \underline{\mathrm{div}}\ 2 \ \& \\
\quad \left\{ \begin{array}{l} m[h]\leq A[n] \rightarrow b(h,j), \\ A[n]<m[h] \rightarrow b(i,h) \end{array} \right\}. \\
b(j-1,j) \rightarrow . \qquad\qquad (45)
\end{array}$$

(45) expresses 'binary search'.

Here we look (44) and (45) combined as a program (a logic program). So starting the specification (31) and (32) we have derived a (logic) program.