A FORMALISM FOR PRIMITIVE LOGIC AND MECHANICAL PROOF-CHECKING

KATUZI ONO

Introduction

The universal character of the primitive logic **LO**[1] in the sense that popular logics such as the lower classical predicate logic **LK**, the intuitionistic predicate logic **LJ**, Johansson's minimal predicate logic **LM**[2], etc. can be faithfully interpreted in **LO** is very remarkable even from the view point of mechanical proof-checking. Since **LO** is very simple, deductions in **LO** could be mechanized in a simple form if a suitable formalism for **LO** is found out. Main purpose of this paper is to introduce a practical formalism for **LO**, practical in the sense that it is suitable at least for mechanical proof-checking business.

I will illustrate our formalism mostly for describing proofs in **LO**, although our device would possibly be applied for other purposes.

Our formalism is so designed that we need not define anything concerning well-formed formulas. Also, it is designed so as to unify IMPLICATION and UNIVERSAL QUANTIFICATION. Before giving formal exposition, I will describe the leading idea by illustrating how to express propositions and formal deductions in **LO** taking up a simple example.

Let us take up an example proof described in my practical way[3].

Proof of $(x)(y)(R(x,y) \to R(y,x)) \to ((x)(y)(z)(R(x,y) \to$
$(R(y,z) \to R(x,z))) \to (x)(y)(R(x,y) \to R(x,x)))$[4] /**A**, **b**.

A) Assume $(x)(y)(R(x,y) \to R(y,x))$
and $(x)(y)(z)(R(x,y) \to (R(y,z) \to R(x,z)))$.

b)) $(x)(y)(R(x,y) \to R(x,x))$ /**bA**, **be**.

Received August 31, 1965.

[1] See Ono [1] and [2]. In [2], **LO** is called PRIMITIVE SYSTEM OF POSITIVE LOGIC.

[2] See Johansson [1].

[3] See Ono[3]. See also Ono [2], in which the inference rules of **LO** are given.

[4] The meaning of this formal proposition is: '*Any symmetric and transitive relation R is reflexive with respect to any object which satisfies R with some object*'.

bA) $\forall uv!$ $R(u, v)$ (u for x and v for y in $R(x, y)$, and I try to show $R(u, u)$ *i.e.* u for x and v for y in $R(x, x)$.).

bb) $R(u, v) \to R(v, u)$ /**A.** (u for x and v for y in the first proposition of **A.**)

bc) $R(v, u)$ /**bA, bb.**

bd) $R(u, v) \to (R(v, u) \to R(u, u))$ /**A.** (u for x, v for y, and u for z in the second proposition of **A.**)

be) $R(u, u)$ /**bA, bc, bd.**

I will explain now, how to express deductions like this. In our formalism, we use only one category of variables (denoted by small latin letters) together with a pair of brackets '[' and ']' called here HEAD- and TAIL-BRACKET, respectively, and the usual comma[5]. I will translate the proof into our formalism keeping the original configuration so as to make correspondence conspicuous.

$R(u, v)$ is denoted as $[ruv]$ (For variables $r, u, \ldots, w$, read $[ru \cdots w]$ 'r holds for $u, \ldots, w$.') taking r as a predicate variable corresponding to R. For example, **bd** is $[ruv] \to ([rvu] \to [ruu])$ which is denoted by $[ruv][rvu][ruu]$. (For sentences $A, \ldots, K, L$, read $A \cdots KL$ '$A, \ldots, K$ imply L.') The second proposition of A is denoted by $xyz[rxy][ryz][rxz]$. (For variables $x, \ldots, z$ and a sentence A, read $x \cdots zA$ 'A holds for all $x, \ldots, z$.') We express the special case of the preceeding proposition, u for x, v for y, and u for z, by $xyz[rxy][ryz][rxz]uvu$. (For two variable series of the same length $x, \ldots, z$ and $u, \ldots, w$ and a sentence A, read $x \cdots zAu \cdots w$ '$x \cdots zA$ holds for $u, \ldots, w$.'[6]) So, this sentence is the same as $[ruv][rvu][ruu]$. The intro-

[5] Polish formalism and our formalism seem to be on antipodes in the sense that Polish formalism seems to avoid brackets or parentheses while ours relies exclusively on brackets. Our formalism is seemingly similar to the formalism introduced in Fitch [1], but interpretations are quite different. As for Polish formalism, see Lukasiewicz and Tarski [1].

[6] I believe, this notation is worth to be explained more thoroughly. One of our policy in denoting propositions is that we disregard the head-brackets at the top and the tail-brackets at the end. Hence, $x \cdots zAu \cdots w$ can be expressed as $[x \cdots zAu \cdots w]$, which can be regarded as the expression obtained from $[ru \cdots w]$ on replacing the predicate r by the universal proposition $x \cdots yA$. Naturally, I dare not say that any universal proposition is essentially the same thing as a predicate. However, I dare say that any predicate is represented by a general rule which can be denoted by a universal proposition of the form $(x) \cdots (z)A(x, \ldots, z)$ in the ordinary notation. Instead of asking whether the predicate of the general rule $(x) \cdots (z)A(x, \ldots, z)$ holds for $u, \ldots, w$, we usually ask whether the general rule holds for $u, \ldots, w$. If we admit $x \cdots zA$ represent a predicate of the general rule $x \cdots zA$, our notation $[x \cdots zAu \cdots w]$ synchronizes with our notation $[ru \cdots w]$ for elementary propositions.

In fact, ambiguous points of the ordinary notations such as $(x) \cdots (z)A(x, \ldots, z)$ can be removed by adopting our notation. So, I believe, this notation should be imported into the ordinary formalism too.

ductory index-word standing at the top of each line is followed by double commas instead of employing a capital letter at its end, when the index-word introduces propositions of assumption character. Double parentheses after an index-word are dispensable, since it can be shown by existence of non-void frame-work[7] of it.

Thus our example proof can be expressed by the following sequence of eight lines:

$[xy[rxy][ryx]][xyz[rxy][ryz][rxz]][xy[rxy][rxx]],\quad a,\ b,$

$a,,\quad xy[rxy][ryx],\ xyz[rxy][ryz][rxz],$

$b,\quad xy[rxy][rxx],\quad ba,\ be,$

$ba,,\quad xy[rxy]uv\quad (i.e.\ [ruv]),$

$bb,\quad xy[rxy][ryx]uv\quad (i.e.\ [ruv][rvu]),\quad a,$

$bc,\quad [rvu],\quad ba,\ bb,$

$bd,\quad xyz[rxy][ryz][rxz]uvu\quad (i.e.\ [ruv][rvu][ruu]),\quad a,$

$be,\quad [ruu],\quad ba,\ bc,\ bd$

Even if we describe this proof in a single line deleting explanations enclosed in the parentheses and without inserting any space between symbols, we can take out the meaning correctly. Namely, the whole sequence of the symbols are divided into a sequence of series of symbols denoting index-words or propositions.

Series of symbols denoting index-words are characterized by that they have no bracket at all. If we represent series of symbols denoting index-words and those denoting propositions in the example proof by *Iij* and *Pij* (*i*-th line, *j*-th series), respectively, our proof can be denoted by

$P01,\ I02,\ I03,\ I10,,\ P11,\ P12,\ I20,\ P21,\ I22,\ I23,\ I30,,\ P31,\ I40,\ P41,$
$I42,\ I50,\ P51,\ I52,\ I53,\ I60,\ P61,\ I62,\ I70,\ P71,\ I72,\ I73,\ I74$

The sequence is divided into eight lines beginning with introductory index-words except for the first line. Introductory index-words can be taken out from the above sequence, because the introductory index-words can be characterized as those index-words standing just before propositions.

I will extend our formalism so as to be able to show conversely that any

[7] See Ono [2]. The class of index-words of the form αl is called the frame-work of the index-word α.

series of brackets, commas, and variables can be regarded as a proof except for a few trivial exceptions. The formalism can be established in such way that we can decide quite mechanically whether any given proof is valid or not.

In (1), I will extend our formalism so that any series of brackets and variables makes sense as a proposition as far as it contains at least one bracket and as an index-word otherwise.

In (2), I will show how to reduce any series of brackets, commas, and variables to a proof-like figure, taken as a sequence of lines, each line being a sequence of index-words and propositions.

In (3), I will describe the primitive logic in such way that we can decide whether any proof-like figure as a sequence of lines is a valid proof in the primitive logic or not. It is remarkable that we can unify the notions IMPLICATION and UNIVERSAL QUANTIFICATION as well as their inference rules in our formalism.

(1) Sentence

We employ head- and tail-brackets. a comma, and a series of variables of the same category. A sequence α of symbols is called PROOF, SENTENCE, or INDEX according as α contains commas, contains no commas but some brackets, or contains neither commas nor brackets. The i-th symbol of α is denoted by α_i, and the subsequence of α from the i-th symbol to the j-th symbol *i.e.* $\alpha_i \cdots \alpha_j$ is denoted by α_{ij} $(i \leq j)$.

Any sentence α is called NORMAL if and only if in every α_{1i} the number of tail-brackets does not exceed the number of head-brackets and the total number of tail-brackets is equal to that of head-brackets. Naturally, in any normal sentence, every bracket can be coupled with its partner as a pair of head- and tail-brackets.

Any variable α_i in a sentence α is called HEAD-VARIABLE or TAIL-VARIABLE according as α_i is immediately followed by a head- or tail-bracket in '$\alpha]$', respectively, skipping over variables.

Any tail-variable α_i of a sequence α standing just after a head-bracket in '$[\alpha$' without skipping over variables, is called PREDICATE VARIABLE. Variables other than predicate variables are called OBJECT VARIABLES[3]. We

[3] The distinction between predicate and object variables has been introduced so as to describe our formalism along the line of lower predicate logics.

distinguish always predicate- and object-variables even when they are denoted by the same letter. We deal with them as if they are denoted by different kinds of letters.

Any object variable in a sequence is called QUANTIFYING VARIABLE if and only if it is a head-variable. Any object variable α_i in a sentence α is called BOUND TO a quantifying variable α_j $(j<i)$ if and only if α_i and α_j are denoted by the same letter and the number of head-brackets in α_{jk} exceeds the number of tail-bracket in the same sequence for any k satisfying $j<k\leq i$ as far as it contains any brackets, but no quantifying variable of the same letter between i-th and j-th places satisfies the condition for α_j. Any object variable in a sentence is called BOUND if and only if it is a quantifying variable or it is bound to a quantifying variable. Otherwise, any object variable in a sentence is called FREE.

Now, we explain BRACKET-TRANSFORMATION, BRACKET-EQUIVALENCE, SUBSTITUTION, and FORMAL EQUIVALENCE of sentences.

(1.1 D) *For any sentence α, any one of 'α', '$\alpha]$', and '$[\alpha$' can be transformed into any other of them. This transformation is called BRACKET-TRANSFORMATION.*

(1.2 D) *Two sentences are called BRACKET-EQUIVALENT if and only if the one can be transformed into the other by finite steps of bracket-transformations.*

(1.3 D) *Any normal sentence $\alpha x\beta]y\gamma$ can be transformed into $\alpha\delta]\gamma$ and vice versa, if δ is the sequence obtained on replacing all the variables x of β by y which are bound to the quantifying variable x of $x\beta$, assuming that α is a series of normal sentences of the form $[\cdots]$, γ is a series of variables, and no quantifying variables y occur in β. This transformation is called SUBSTITUTION*[9].

(1.4 D) *Two sentences are called FORMALLY EQUIVALENT if and only if the one can be transformed into the other by finite steps of bracket-transformations and substitutions.*

Lastly, I will write down some theorems concerning these notions.

(1.5 T) *If two sentences are bracket-equivalent to each other, they are also formally equivalent.*

[9] Notice that $r[rr]u$ is not $[uu]$ but it is $[ru]$, for example.

(1.6 T) *Any sentence is bracket-equivalent to a normal sentence.*

(1.7 T) *Let α and β be bracket-equivalent sentences. Then, for any variable α_i of α, there is a well-determined variable β_j of β which corresponds to α_i. Naturally, α_i and β_j are denoted by the same letter.*

(1.8 T) *Any substitution transforms any normal sentence into a normal sentence.*

(1.9 T) *Let α and β be normal sentences formally equivalent to each other, α_i be a head- (or tail-) bracket having the tail- (head-) bracket α_j as partner in α, and β_k be the bracket corresponding to α_i. Then, the partner bracket β_l of β_k by the coupling of brackets in the normal sentence β corresponds to α_j of α.*

(1.10 T) *Let α and β be sentences formally equivalent to each other, and let α_i and β_k as well as α_j and β_l be a pair of corresponding variables of both sentences. Then, α_i is a head-variable, tail-variable, predicate variable, object variable, quantifying variable, bound variable, or free variable in α according as β_k is so in β, and α_j is bound to α_i in α according as β_l is bound to β_k in β. In short, notions of head-variable, tail-variable, predicate variable, object variable, quantifying variable, bound variable, free variable, and variable being bound to another variable are all invariant with respect to formal equivalence.*

(2) **Proof.**

Any proof π is divided into a sequence ϕ of indices and sentences by commas disregarding the number of successive commas. We delete at first indices and commas standing at the top of π. Indices standing just before a sentence or followed by a successive series of commas no less than two are called INTRODUCTORY INDICES. Other indices are called REFERENCE INDICES.

The sequence ϕ of sentences and indices is divided into a sequence λ of LINES by introductory indices. Each line begins with an introductory index followed by a series of sentences and a series of reference indices (both series can be vacant) except for the first line which lacks introductory index. To unify the forms of lines as possible, I will regard the first line as beginning with the introductory index being a null series.

It is our policy to disregard the number of successive commas more than two. Accordingly, replace every series of successive commas more than two by double commas. Introductory indices followed by double commas are called ASSUMPTION INDICES. Lines beginning with assumption indices represent assumptions, so delete all reference indices in these lines. Delete also all reference indices which do not occur as introductory indices.

Now, the sequence of introductory indices must be arranged in the lexicographic order with respect to the order of letters. Even when we are not given such order of letters beforehand, we can usually settle a possible order between them (but, in general, not uniquely) so that the introductory indices of ϕ are arranged in the lexicographic order with respect to it.

The class of all indices of the form $\eta_{1(i-1)}n$ whose n stands before η_i in the alphabetical order is called GROUND OF η[10], and the class of all indices of ϕ of the form ηn is called FRAME-WORK OF η.

Anyway, we can normalize proofs so that any normalized proof has the following characteristics:

i) *There is a fixed order of letters for variables.*

ii) *At the top, stands a sentence or sentences unless the proof is reduced to a null sequence by normalization.*

iii) *All the introductory indices are arranged in the lexicographic order.*

iv) *In any line beginning with an assumption index lies no reference index.*

v) *In any line beginning with an introductory index other than assumption indices lies at least one sentence.*

(3) Primitive Logic

Let π be any normalized proof and λ be the sequence of lines of π. π is called a VALID PROOF of the sentences of the first line in the primitive logic **LO** if and only if every introductory index ξ in λ satisfies the following conditions.

i) *The case where the frame-work of ξ is not vacant:*

The frame-work of ξ should contain one and only one assumption index ξm. In the line beginning with ξ, there should be a single sentence of the form $\mu\alpha^1 \cdots \alpha^k\alpha$, where μ is a series of variables and $\alpha^1, \ldots, \alpha^k$, and α are normal sentences of the form $\sigma[\text{---}]$, σ being a series of variables (including the case $k=0$). The reference indices in the line should be ξm and ξn (or ξm

[10] In Ono [2], this is called ASSUMPTION OF η.

only if n coincides with m). The line beginning with ξm should be a sequence of the form

$$\xi m,, \mu\alpha^1\omega, \ldots, \mu\alpha^k\omega,$$

where ω is a series of the same number of mutually distinct variables as μ and no variable of ω occurs as free in any line beginning with ξ or beginning with an index in the ground of ξ. In the frame-work of ξ, there should be an index ξn such that the line beginning with ξn is a sequence of the forms

$$\xi n, \beta, \cdots, \quad \text{or} \quad \xi n,, \beta,$$

where β is a sentence formally equivalent to $\mu\alpha\omega$ for the series μ and ω introduced just before, and '$\cdots$' represents a sequence of indices. (In the latter case n coincides with m.)

ii) *The case where the frame-work of ξ is vacant.*

For any sentence γ in the line beginning with ξ, there should be a sequence of indices $\eta, \eta^1, \ldots, \eta^k$ in the ground of ξ such that there is a sentence formally equivalent to $\mu\alpha^1 \cdots \alpha^k\alpha$ in the line beginning with η, the sentences $\alpha^1, \ldots, \alpha^k$ are normal ones of the form $\sigma[---]$, for a series σ of variables, γ is formally equivalent to $\mu\alpha\omega$ for a series ω of the same number of variables as μ, and there is a sentence formally equivalent to $\mu\alpha^i\omega$ in some line beginning with an index η^j.

Theoretically, I could describe the inference rules for **LO** more simply. However, our rules described here enable us to write down formal proofs shorter.

Formal proofs thus far can be easily checked mechanically. Namely, we can check mechanically whether given proofs are valid or not. In our real thinking, however, we use skipped description and further rely upon metalogical understandings. Skipped deductions can be also formalized in our formalism. For mechanical checking of skipped deductions, we need a programming for proving any provable sentences. However, we need not decide whether the proofs are valid or not as a skipped proof. It is enough to give an alarm mechanically if a certain step can not be followed easily. Programming for mechanical proofs must be simpler in the primitive logic compared with those in other logics. Reduction of proofs in the other logics to proofs in the primitive logic can usually be done mechanically. Reduction to the primitive logic seems practical even for mechanical checking of skipped proofs, since we need not interpret back into original logics.

Concerning metalogical understanding, I have not to say much. I would like to point out only that the usage of predicate variables introduced in this paper may have some connection with the matter.

References

[1] Church, A. [1] Introduction to Mathematical Logic, I, Princeton, 1956.

[2] Fitch, F. B., [1] A basic logic, J. Symb. Log., vol. **7** (1942), pp. 105–114.

[3] Johansson, I., [1] Der Minimalkalkül, ein reduzierter intuitionistischer Formalismus, Compositio Mathematica, vol. **4**(1936), pp. 119–136.

[4] Lukasiewicz, J. and Tarski, A., [1] Untersuchungen über den Aussagenkalkül, Comptes Rendus des séances de la Société des Sciences et des Lettres de Varsovie, vol. **23** (1930), Cl. III, pp. 31–2.

[5] Ono, K., [1] On universal character of the primitive logic, Nagoya Math. J., vol. **27-1** (1966), 331–353.

[6] Ono, K., [2] A certain kind of formal theories, Nagoya Math. J., vol. **25** (1965), pp. 59–86.

[7] Ono, K., [3] On a practical way of describing formal deductions, Nagoya Math. J., vol. **21** (1962), pp. 115–121.

Mathematical Institute
Nagoya University