数式処理 *J.JSSAC (2004)*
Vol. 10, No. 3, pp. 49 - 62

論文

数学記号の認知速度
—実験心理学的計測方法と実例—

堀 幸雄* 後藤 英一†

神奈川大学理学研究科情報科学専攻 神奈川大学理学部情報科学科

佐藤 雅彦‡

京都大学大学院情報学研究科知能情報学専攻

(Received 2003/3/19 Revised 2003/11/18)

Abstract

We report basic experiment data of readability about three logic symbols "∨ ∧" in mathematics, ".OR. .AND." in Fortran and "`||` `&&`" in C(Java) first. As cognitive psychology measurement method, we propose three methods for readability in logic symbols. Those are a symbol count experiment, query with binomial expression experiment, and query polynomial expression experiment. These experiment's result was observed following power law of practice as the acquisition model of skill ($t = \alpha p^{-\beta} + \gamma$, t is execution time, p is total practice time, initial skill $\alpha+\gamma$, degree of proficiency β, skilled value γ). In a symbol count experiment result, it was 1.6:1.5:1 in the ratio of initial skill, at skilled value it was 2.2:2:1 in the ratio of mathematics, Fortran and C. In the same experiment "∨ ∧" in mathematics was 90% of ratio of a correct answer, others being 95% or more of rates of a correct answer. Therefore "∨ ∧" is easy to misidentify the symbol counts. "∨ ∧" is a bad symbols with low rate of a correct answer and cognitive time late twice as compared with "`||` `&&`". This paper also describes readability of inequality symbol, predicate logic symbol and negative symbol. The quantity-judging about readability as shown here should be added in the design of writing system of programming language, symbolic and algebraic computation.

1 はじめに

算術演算記号 「+, −, ·,×, /,÷」 は 500 年以上に渡り使われ，その意味は定着している．19 世紀後半に論理的推論を算術式と類似の記号を用いて形式的に表現する記号論理学が成

*horiyuki@goto.info.kanagawa-u.ac.jp
†goto@goto.info.kanagawa-u.ac.jp
‡masahiko@kuis.kyoto-u.ac.jp

表 1: 数学とプログラミング言語における論理と比較演算子

言語	論理演算子	比較演算子	真偽値定数
数学	$\vee$ $\wedge$ (その他付録表 7)	$> < \geqq \leqq = \neq$	
Fortran	`.OR. .AND.`	`.GT. .LT. .GE. .LE. .EQ. .NE.`	`.TRUE. .FALSE.`
C(Java)	`|| &&`	`> < >= <= == !=`	non-zero(真) 0(偽) `true false` (Java)

立してから約 150 年，計算機処理を手続きとして記述するプログラミング言語はまだ誕生してから 50 年余りである．数学やプログラミング言語の命題論理記号，不等号記号，述語論理記号には多種多様のものが使用されている．しかしそれらに対して読みやすさを示すデータは筆者らの知るところ報告はない．数学の述語論理の記号についてはその変遷は論じられているが，読みやすさについては論じられていない (付録 A 参照)．プログラミング言語 C の提案書には記号の変遷と読みやすさに関する記述はない [17].

我々は記号の読みやすさを計る認知心理学的測定方法として，記号単体の形を数え上げる記号対計数実験，記号両側二項の大小関係を問う二項実験，多項に渡る関係を問う多項実験の三種の実験方法を提案し，この結果について報告する．これらの実験の習熟過程は技巧獲得の修得モデルである練習のべき法則 ($t = \alpha p^{-\beta} + \gamma$, t: 遂行時間, p: 総練習時間) [7][2] に従うことが観測された．各実験における各記号の初期認知時間 ($\alpha + \gamma$)，上達度 (β)，到達時間 (γ) の計測結果を報告する．意味を問わない記号対計数実験において到達時間で数学の「$\vee$ $\wedge$」は 3.3 秒， C の「`|| &&`」は 1.5 秒であった．数学の「$\vee$ $\wedge$」は C の「`|| &&`」と比較して認知時間が 2 倍以上遅いうえに正解率も低い悪い記号対である．記号列の識別速度には非常に差があり，二項，多項実験においても「$\vee$ $\wedge$」が読みにくいことは変わらなかった．

本稿の議論の中心は命題論理記号で表現される読みやすさである．式で使用される命題論理記号の基本的な 3 つの関係 (一項関係，二項関係，三項関係) の組み方による読みやすさを実験的に求める．記号の読みやすさの要因には文字のサイズ，活字体種別，色付け等も考えられるがここでは組み方を基にした読みやすさのみを対象とする．

本稿で示す記号の読みやすさは今後さらに研究されるべき問題である．数式処理系やプログラミング言語の記号体系の設計に際してここで示したような実験を元に記号の読みやすさを定量的に考慮すべきである．

2 本稿で扱う記号

本稿で取り扱う記号を表 1 に示す．現在多くのプログラミング言語が存在するが，ここでは代表的な言語として Fortran[15]，C[17] と Java[35] の各言語について取り上げる．

Fortran では文字集合を PC(パンチカード) に準拠するように設計され，使用可能な特殊記号は「空白 `= + - * / ( ) , . $`」に限っていたため表 1 のような演算子になったと

思われる．C は処理系を ASCII コード [1] に準拠するものとして設計された．論理演算子を「|| &&」に決めた理由は文献 [17] に論じられていないが ASCII コードに「∧」がないためと思われる．Java ではコード体系として Unicode[36] が使用できるが論理演算子には C と同様に「|| &&」を採用している．これは C と ASCII コードとの下位互換性を持たせた結果と思われる．

Unicode の注目すべき特徴に記号の重ね打ち (結合文字 combining character) がある．Unicode ではアクセント付き文字などの動的な合成を許している．これは図 1 のようにベースの文字コードに続いてアクセント文字などの結合文字のコードを並び続けることによって 1 つの文字を合成するというものである．結合文字は入力方式，重複符号化，無制限文字品目などに対する処理がまだ確立されておらず実用的に普及されていない [18].

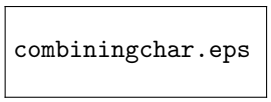

図 1: 結合文字の例

3 記号認知速度計測実験

3.1 実験目的

数学記号やプログラミング言語で用いられる論理積和記号と不等号記号について定量的読みやすさの指標を示すことを目的とする．認知心理学的測定方法としてここでは記号の読みやすさを下記の 3 種の方法に分けて実験を行った．

(1) 記号対計数実験 (単記号実験): 記号対のいずれかの形を数えあげる実験

(2) 二項実験: 記号両側二項の大小関係を問う実験

(3) 多項実験: 多項に渡る関係を問う実験

3.2 実験方法

被験者を情報科学科の卒研生 20 名とし，実験は計算機端末上で以下の手続きを行なう実験プログラムを作成して行なった．実験環境と被験者に関するデータを表 2 に示し，以降実験 (1), (2), (3) における実験手順を説明する．

(1) 記号対計数実験

記号対計数実験とは一列に記号対を並べ，指定された記号の個数をなるべく速く数える実験である．記号の個数は答えがキーボードの 1 打鍵で入力できるように 6〜9 個とし，問題ご

表 2: 実験環境と被験者のデータ

実験期間:	2002 年 5 月 〜 2002 年 10 月.
被験者:	情報科 4 年生 15 名，情報科大学院生 5 名.
使用環境:	CRT 17inch カラーモニタ，解像度 1280x1024 で使用.
実施方法:	各実験を順番に行ない，各実験 1 回の出題数は 20 問.
	各実験の間には休憩時間 1 分が入る.
	1 日の実験時間は 1 時間とする.

とに記号の個数や記号間の空白の個数を変えることで，記号列毎に正確に記号の個数を数えなければならないようにする.

この実験を数学の各論理記号「∨ ∧」,「∨ ·」,「∨ &」，プログラミング言語の論理演算子「.OR. .AND.」,「|| &&」に加え集合演算子「∪ ∩」，量化記号「∃ ∀」，そして真偽値定数である「.TRUE. .FALSE.」,「1 0」,「true false」で行なう．また「∨ ∧」に「·」や「|」やを重ね打ちした記号「⩒ ⩓」,「⩛ ⩕」についても合わせて実験を行った.

実験時の問題例を図 2 に示す．被験者は記号対の中から問題毎に指定された記号の個数をなるべく速く数える．この実験では記号の意味は問わず，記号対における認知されやすさを計測する.

```
          ∨   ∧ ∨   ∨ ∧   ∨   ∧   ∨ ∧
 .OR.    .AND.    .AND. .OR.    .OR. .AND. .AND.
           &&     ||  &&    && ||    &&
```

図 2: 記号対計数実験問題例 (実際の問題出力は 1 行毎)

(2) 二項実験

不等号記号「$>, <, \geqq, \leqq, =, \neq$」を用いて二項間の大小比較の真偽を問う実験を行なう．比較される数字は 1 桁の 10 進数とし，比較演算子に表 1 の各記号を用いる．この実験では記号単体の意味を含めた読みやすさの計測を行なう.

問題例としては「$1 < 2$」(真) や「3 .GE. 4」(偽) のようになる．このような問題を 1 問ずつ出力していき，被験者は問題毎に真偽をなるべく速く判別していく．答えはキーボード 1 打鍵で入力できるように表示された問題が真ならば t，偽ならば f キーを使用する.

(3) 多項実験

複数の記号を組み合わせた式の関係を問う問題は色々考えられるが，ここでは論理記号の双対性 [16] を表わす式の真偽を問う実験を行なう．この実験においても (1) の実験と同様の論理記号対を用いる.

分配則，吸収則，ド・モルガンの定理における双対性の問題例を図 3 に示す．否定記号には重ね打ちの「¯」を用いた．この図 3 の例をもとに式の論理記号をランダムに変化させ，被

験者は出力された問題毎に式の真偽を判別する．(2) の実験と同様に答えの回答には t,f キーを使用する．

$$x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z), \quad x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z),$$
$$x \wedge (x \vee y) = x,\ x \vee (x \wedge y) = x, \quad \overline{x \wedge y} = \overline{x} \vee \overline{y},\ \overline{x \vee y} = \overline{x} \wedge \overline{y}$$

図 3: 双対性を表わす問題例 (「$\vee$ $\wedge$」 の場合)

(3.1) 否定解釈実験

多項実験では否定記号を用いるが否定記号の読みやすさとして同変数を結ぶ二項間に否定記号を用いた式とド・モルガンの定理の真偽の判断を行なわせる実験を考える．二項間における否定記号は 0 個から二重否定までをランダムに変化させる．否定記号には前置の「$\neg x$」，「$\sim x$」，重ね打ちの「$\overline{x}$」，後置の「x'」を使用する．回答はこれまでと同じく t,f キーの打鍵で行なう．この実験における問題例を図 4 に示す．

$$\neg\neg x = x, \quad \neg x \neq \neg x, \quad \neg(x \wedge y) = \neg x \vee \neg y, \quad \neg(x \vee y) = \neg x \wedge \neg y$$
$$\overline{\overline{x}} = x, \quad \overline{x} \neq \overline{x}, \quad \overline{x \wedge y} = \overline{x} \vee \overline{y}\ , \quad \overline{x \vee y} = \overline{x} \wedge \overline{y}$$
$$x'' = x, \quad x' \neq x', \quad (x \wedge y)' = x' \vee y'\ , \quad (x \vee y)' = x' \wedge y'$$

図 4: 否定解釈実験問題例

本稿で提案した全ての実験は計算機端末上に出力し，入力は端末のキーボードを用いて被験者に応えてもらう形とした．そのため特別な専門の装置は必要としない．今回は出来るだけ回答の負担を軽減するために回答を英字 1 打鍵で行なえるようにした．認知の速さと正確さにはトレードオフがあるが今回の場合回答の負担をなるべく軽減するようにし，なるべく早く認知してもらうように全被験者に教示した．また被験者は全員キーボードをタッチタイプできる学生である．

この実験のプログラム実行時の出力例を図 5 に示す．実験プログラムは文字の大きさは 16pt のゴチック体を用い，背景色を白，文字色を黒に統一した．

3.3 実験結果の要点

前節で説明した実験 (1),(2),(3) のそれぞれを被験者に 3 ヶ月程定期的に実行してもらった．これら実験の習熟過程は技巧獲得の修得モデルである練習のべき法則に従うことが観測された．それぞれの実験の到達時間が十分に習熟した実験開始 200 時間までの結果について見てみる．

図 5: 実験プログラム実行図

実験 (1) の記号対計数実験の習熟結果と正解率の推移を図 6 に示す．横軸は累積実験時間であり，縦軸に記号の数え上げにかかった全被験者の平均計測時間，正解率をプロットしてある．また誤差棒はこの記号対計数実験における個人差を表わしている．

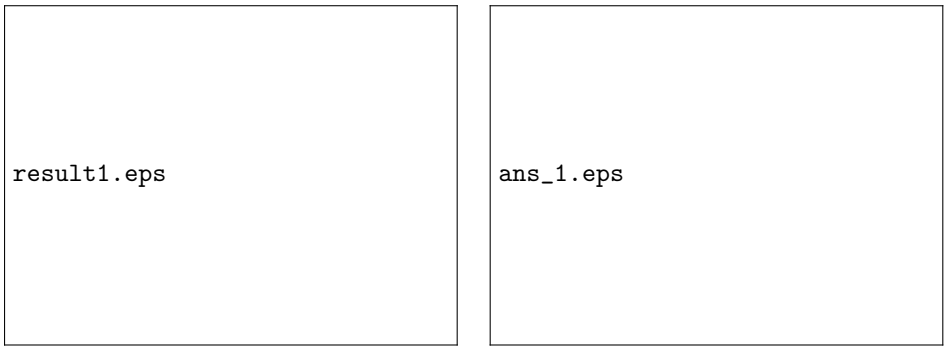

図 6: 記号対計数実験の習熟曲線と正解率の推移

全記号において実験時間が長くなるにつれ，個人差は小さくなっている．累積実験経過時間が 100 時間を超えたあたりで各被験者の個人差は 1 秒以内と非常に小さいものであることがわかる．最も記号の数え上げが早かったのは「|| &&」であり，実験が 100 時間を超えたあたりでほぼ習熟し 1.5 秒程で数え上げを行っている．それに対し最も遅かった記号は「∨ ∧」であり，到達時間で最も早かった「|| &&」に比べ 2 倍近い 3.3 秒程かかっている．集合演算に対する記号「∪ ∩」も「∨ ∧」程悪くはないが 3.0 秒程である．Fortran の「.OR. .AND.」，および Fortran，Java の真偽値定数である「.TRUE. .FALSE.」,「true false」も同じ 3.0 秒程であったが，これらの記号類は文字数が他の記号列に比べて長いためにこの実験においては不利であったと思われる．習熟曲線より最小二乗法 [20] を用いて練習のべき法則にあてはめ求めた記号対計数実験における全記号の初期時間，上達度，到達時間，および誤差を表 3 に示す．

表 3: 記号対計数実験における初期時間，上達度，到達時間

真偽値定数	(初期時間，上達度，到達時間)
true false:	$(6.22 \pm 2.11, 0.18 \pm 0.13, 3.03 \pm 0.79)$
.TRUE. .FALSE.:	$(6.35 \pm 2.47, 0.22 \pm 0.15, 2.89 \pm 0.80)$
1 0:	$(4.76 \pm 1.38, 0.21 \pm 0.20, 1.86 \pm 0.55)$
論理積和 / 量化記号	
$\vee\ \wedge$:	$(4.82 \pm 1.68, 0.29 \pm 0.14, 3.39 \pm 1.09)$
.OR. .AND.:	$(4.56 \pm 1.64, 0.23 \pm 0.16, 3.10 \pm 0.95)$
$\cup\ \cap$:	$(4.45 \pm 1.46, 0.18 \pm 0.15, 3.07 \pm 0.93)$
$\Vee\ \Wedge$:	$(4.22 \pm 1.22, 0.11 \pm 0.09, 2.42 \pm 0.59)$
$\dot{\vee}\ \dot{\wedge}$:	$(4.13 \pm 1.15, 0.11 \pm 0.08, 2.37 \pm 0.53)$
$\forall\ \exists$:	$(3.68 \pm 0.99, 0.15 \pm 0.11, 2.14 \pm 0.51)$
$\vee$ &:	$(3.72 \pm 1.10, 0.22 \pm 0.10, 1.65 \pm 0.43)$
$\vee\ \cdot$:	$(3.43 \pm 1.09, 0.21 \pm 0.14, 1.64 \pm 0.52)$
|| &&:	$(2.92 \pm 0.89, 0.43 \pm 0.07, 1.49 \pm 0.37)$

「$\vee\ \wedge$」に「$\cdot$」や「$|$」を重ね打ちした記号,「$\dot{\vee}\ \dot{\wedge}$」,「$\Vee\ \Wedge$」は到達時間で 2.7 秒程で記号の数え上げを行なうことができた．元の記号「$\vee\ \wedge$」から 25 % 程の計測時間の改善である．

また「$\vee\ \wedge$」は記号の数え上げの誤りが最も多かった．実験時間が習熟しても正解率はあまり変化せず，正解率が習熟しない常に誤りの起こりやすい記号対であることがわかった．しかし重ね打ち記号では読みやすさと正解率が改善されていることがわかる．

実験 (2) の二項実験の習熟結果と正解率の推移を図 7 に示す．今回の被験者らにとって Fortran の不等号記号は他の記号に比べ馴染がなく，また字数も多いため数学記号，C(Java) のプログラム記号よりも初期認知時間が非常に遅かったが，実験経過時間が 100 時間を超えたあたりで他の記号と差がなく認知されていくようになった．二項実験の初期時間，上達度，到達時間を表 4 に示す．

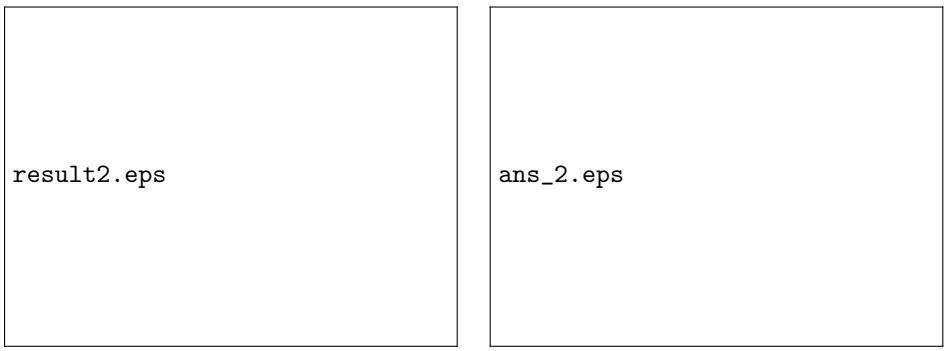

図 7: 二項実験の習熟曲線と正解率の推移

二項実験では二項間の真偽を答えるだけであるので，実解開始初期 Fortran の比較演算子を除き正解率はほとんど 100% というものであった．Fortran の比較演算子も習熟していくと他の記号と同じくほとんど誤りが無くなっている．

表 4: 二項実験における初期時間，上達度，到達時間

各不等号記号	(初期時間，上達度，到達時間)
Fortran：	$(8.56 \pm 2.40, 0.42 \pm 0.10, 1.83 \pm 0.40)$
数学記号：	$(3.35 \pm 0.55, 0.21 \pm 0.07, 1.40 \pm 0.28)$
C(Java)：	$(3.32 \pm 0.50, 0.29 \pm 0.14, 1.25 \pm 0.33)$

実験 (3) の多項実験の習熟結果を図 8，表 5 に示す．この双対性の真偽を問う実験では，記号対計数実験に比べ式の真偽を考える時間が入るためかそれぞれの記号間の認知時間差が小さいものであった．ここでも記号「$\vee$ $\wedge$」は最も認知が早かった「`||` `&&`」と比較して 2 倍程度遅い結果である．

図 8: 多項実験の習熟曲線と正解率の推移

表 5: 多項実験における初期時間，上達度，到達時間

各論理記号	(初期時間，上達度，到達時間)
$\vee$ $\wedge$:	$(3.98 \pm 0.94, 0.48 \pm 0.05, 1.75 \pm 0.50)$
`.OR. .AND.`：	$(3.95 \pm 0.89, 0.44 \pm 0.12, 1.74 \pm 0.50)$
$\veebar\barwedge$:	$(3.23 \pm 0.59, 0.35 \pm 0.13, 1.69 \pm 0.53)$
$\underset{\cdot}{\vee}\dot{\wedge}$:	$(3.14 \pm 0.60, 0.42 \pm 0.10, 1.54 \pm 0.42)$
$\cup$ $\cap$:	$(2.95 \pm 0.56, 0.40 \pm 0.14, 1.58 \pm 0.43)$
$\vee$ &:	$(2.67 \pm 0.49, 0.35 \pm 0.13, 1.29 \pm 0.49)$
$\vee$ $\cdot$：	$(2.24 \pm 0.44, 0.41 \pm 0.07, 1.07 \pm 0.31)$
`||` `&&`:	$(2.83 \pm 0.54, 0.39 \pm 0.15, 0.95 \pm 0.29)$

実験 (3.1) の否定解釈実験の習熟結果を図 9，表 6 に示す．否定記号としては重ね打ちがわずかに早い結果となり，後置，前置は少し遅れてほぼ均衡している．

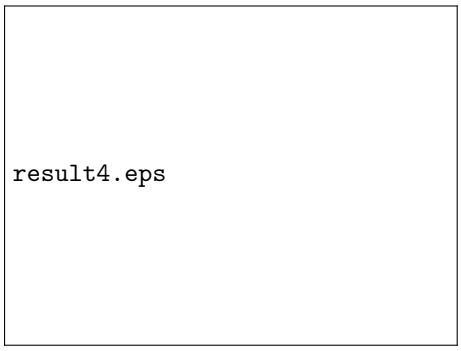

図 9: 否定解釈実験の習熟曲線と正解率の推移

表 6: 否定解釈における初期時間，上達度，到達時間

各論理記号	(初期時間，上達度，到達時間)
前置 $\neg x$:	$(2.38 \pm 0.56, 0.41 \pm 0.14, 1.54 \pm 0.41)$
前置 $\sim x$:	$(2.37 \pm 0.54, 0.40 \pm 0.13, 1.45 \pm 0.39)$
後置 x':	$(2.24 \pm 0.49, 0.38 \pm 0.11, 1.39 \pm 0.43)$
重ね打ち $\bar{x}$:	$(2.15 \pm 0.44, 0.35 \pm 0.10, 1.17 \pm 0.31)$

4 考察

4.1 我々の提案

今回我々の行なった全ての実験において数学で使用される「$\vee$ $\wedge$」が最も認知されにくく，また誤認されやすい記号であることがわかった．しかしこれに「|」や「·」を重ね打ちした記号では読みやすさ，認知の正確さという面で改善が見られた.「$\vee$ $\wedge$」以外の記号には特に目立った欠陥は発見できなかった．

今回のような読みやすさの評価の指標を裏付けるために英文や日本文等のいくつかの文の相対的な読みやすさを測る方法でクローズ法 [34] がある．クローズ法は次のような手続きを取る．

1. ほぼ同じ長さの文の一部の語を，その文の中の働きや意味を考えずに一定の割合で抜き取る．抜き取る割合は 10% 〜 20% の間とし，ランダムに抜き取るかまたは一定の割合で抜き取る．語の抜き取った場所には一定の長さの空白を入れる．

2. このようにして作った文を何人かの被験者に見せ，残りの文脈から空白に入るべき語を書き込ませる．

3. 各文ごとにもとの語が正しく復元された数の被験者全員についての合計を求める．これをクローズスコアという．またクローズスコアを (空白の個数 $\times$ 被験者の数) で割ったものを求める．これを平均的中率という．平均的中率が高い文ほど読みやすい文である．

日本語や英語の文については同様に語を抜き取りの単位とすれば良いことがわかっている [29]．日本文や英文の場合，読みやすさの評価実験とこのクローズ法を用いて読みやすさの指標が正しいことを裏づけている [33][5].

しかし算術式，論理式の項は冗長性がなく，クローズ法を適用することはできない．論理記号対の読みやすさの指標は今回我々の行なった実験で簡単に各記号の相対的読みやすさを求めることができ，結果として有意差が認められる．

また重ね打ち記号は今回の実験でわかったように読みやすさ，誤認を防ぐ効用がある．命題論理積和記号は「|| &&」で良いが，集合演算子の「∪ ∩」は「∨ ∧」に次いで読みにくい記号であった．これに「| ·」をそれぞれ重ね打ちし,「⊍ ⋒」といった記号対を使用することをお勧めする．それぞれに別の記号を重ね打ちするのは「| ·」がそれぞれ論理和，論理積を表す記号であることと，180 度回転した記号対は読みにくいという本実験での結果からである．この他に重ね打ち記号は BNF (Backus Naur Form)[6][4] のメタ記号に用いることも考えられる．BNF のメタ記号の「<>」や「|」,「∗」といった記号に「¨」を重ね打ちし,「<̈ >̈」や「|̈」,「∗̈」といった記号をメタ記号として使用すれば他の記号と間違えずに済む.「::=」は他に使用された例を見ないので今回は特に考えないことにする．このようにメタ記号のように特殊な意味を持つ記号を重ね打ちすることで他の記号との違いを明確に表すことが可能となる．

命題論理記号，述語論理記号及びプログラミング言語等の処理系の表記体系は歴史が浅く各記号が十分に定着しているとはいえない．これら記号が変わることは十分に考えられる．特にプログラミング言語の表記体系のコード変換は容易に実現可能で，言語処理系に渡す前の段階でプログラミング言語の予約語，識別子を別の記号 (日本語等) に変更することができる [31]．また重ね打ち記号は Unicode で規格化されているが，現在入力方式が一般的になっていないため標準的に使えるようにはなっていない．入力方法が一般的になれば重ね打ち記号を用いた表記体系も使えるようになる．新しい演算子記号が今までの記号と比較して相対的にどの程度読みやすいかは今回行なった実験が参考になるだろう．

4.2 今後の展望

ここで示した実験のような記号の読みやすさを定量的に計る実験は今後より研究されるべきテーマである．プログラミング言語の論理記号は「∨ ∧」が ASCII コードにないということで読みやすくなった．Unicode を使うことで「∨ ∧」を使うことができるが C, Java で普及した「∨ ∧」でない記号が生き残ることを切望する．今回は命題論理記号，不等号記号等の記号単体の読みやすさの定量的指標を実験的に求めたが，述語論理記号，プログラムコード全体に対し読みやすさの指標をどのように表わすのかは今後の残された課題である．

表 7: 論理記号の変遷

著者 (文献番号)	発行年	連言	選言	含意	否定	全称	特称
ペアノ [23]	1889	$\cap$	$\cup$	$\supset$	$-$	なし	なし
ホワイトヘッド・ラッセル [37]	1910	$\cdot$	$\vee$	$\supset$	$\sim$	(x)	$(\exists x)$
ヒルベルト・アッケルマン [13]	1928	$\&$	$\vee$	$\rightarrow$	$\overline{}$	(x)	$(\mathrm{E}x)$
ハイティング [12]	1930	$\wedge$	$\vee$	$\supset$	$\neg$	(x)	$(\mathrm{E}x)$
ゲーデル [10]	1931	$\cdot$	$\vee$	$\supset$	$\sim$	$x\Pi$	$(\mathrm{E}x)$
ゲンツェン [8]	1935	$\&$	$\vee$	$\supset$	$\neg$	$\forall x$	$\exists x$
クリーネ [19]	1952	$\&$	$\vee$	$\supset$	$\neg$	$\forall x$	$\exists x$
ヒルベルト・アッケルマン [14]	1958	$\wedge$	$\vee$	$\rightarrow$	$\neg$	$\forall x$	$\exists x$
シュッテ [24]	1960	$\wedge$	$\vee$	$\supset$	$\neg$	$\bigwedge x$	$\bigvee x$
シェーンフィールド [30]	1967	$\&$	$\vee$	$\rightarrow$	$\neg$	$\forall x$	$\exists x$
竹内 [32]	1975	$\wedge$	$\vee$	$\supset$	$\neg$	$\forall x$	$\exists x$
シュッテ [25]	1977	$\wedge$	$\vee$	$\rightarrow$	$\neg$	$\forall x$	$\exists x$

付録

A 論理記号の変遷

命題や判断を記号を用いて形式的に表現することにより，現代的な記号論理学が成立したのは 19 世紀後半のことであった．この節ではその頃から 1970 年代までに出版された論理学関係の書籍や論文でどのような記号が用いられてきたかを，学問的に影響力が大きかった文献について見てみることにする．

表 7 では，命題論理の論理記号で「連言 (かつ)」,「選言 (または)」,「含意 (ならば)」,「否定 (でない)」を表すものと，述語論理の量化記号で「全称 (すべて)」と「特称 (ある)」を表すものを示してある．量化記号については，量化される変数として「x」を選んで明示してある．

ペアノは [23] で自然数の公理系を与えているが，述語の代りに集合を用いて理論を展開している．そのために，全称，特称に対する記号を用いていない．注目すべきは「ならば」に対して,「C」を 180 度回転させた記号「$\supset$」を用いていることである．この記号を導入する直前に彼は，命題 a,b に対して，b が a からの帰結である (b is a consequence of a) ことを $b\,\mathrm{C}\,a$ で表わすと定め，その直後に，同じことを，$a \supset b$ で表わすと定めている．[11] によれば，この記号が変化して，現在標準的に用いられている記号「$\supset$」になったとしている．

ハイティングは論文 [12] で否定の記号としては「$\sim$」でなく新しい記号「$\neg$」を用いると述べている．

ゲンツェンは論文 [8] の中で論理記号の選択について次のように述べている．記号「$\vee$」,「$\supset$」,「$\exists$」はラッセルにしたがう[1)]が，ラッセルの「$\cdot$」,「$\sim$」,「( )」(全称) は数学で別の意

[1)] これは [37] を指すものと思われる

味に用いられることがあるので採用しない．その代り，連言にはヒルベルト[2)]の「&」を用いるが，ヒルベルトの「( )」,「¯」はやはり数学で別の意味に用いられることがあるので採用しない．そこで否定についてはハイティングの記号を用い，全称については「∃」に対応する「∀」を用いる．

このようにして，ゲンツェンによってほぼ現在標準的に用いられている記法が確立したと考えられる．[9] でも指摘しているように全称の記号「∀」はゲンツェンが導入したものであるとされている．

以上のような変遷を経て，現在では竹内 [32] で用いられている論理記号が標準的なものとなっている．[3)]

一方，形式的な論理学の成立とほぼ同時期に，ブールにより，ブール代数が考案された．ブールは彼の体系をもともとは論理の体系として提案したが，後に計算機等で用いられる論理回路の記述に便利な代数系として用いられるようになった．文献 [3] では論理を集合に対する操作として扱っており，全体集合と空集合をそれぞれ「1」と「0」で表し，集合の和 (合併) と積 (共通部分) をそれぞれ「+」と「·」で表している．これらの記号は代数で既に用いられていた記号であったので，論理記号の場合と異なり，彼の体系が代数系として認識されるようになってからもそのまま採用されてきている．(ただし，集合の代数という意識がある場合には集合算の記号である「∪」と「∩」が用いられることもある．) 例えば，シャノンは文献 [26], [27] でブールと同じ記号を用いており，日本人として電気回路の代数的性質を早い時期に考察した中嶋も文献 [21], [22] で同じ記号を用いている．

またシェッファは NAND を表わす彼の演算記号「|」を導入した論文 [28] で，論理和と論理積にぞれぞれ記号「⊙」と「⊕」を用いている．シェッファはこの「|」と 「( )」を用いてすべての論理式が表現できることを示した．

参 考 文 献

[1] ANSI X3.4-1986 (R 1992) Coded Character Set—7-Bit American National Standard Code for Information Interchange

[2] Blackburn, J.M.: Acquisition of Skill: An Analysis of Learning Curves, IHRB Report, No. 73, 1936.

[3] Boole, G.: The Calculus of Logic, *The Cambridge and Dublin Mathematical Journal*, Vol. 3, pp. 183–198, 1848.

[4] Bray, T., Paoli, J. and Sperberg-McQueen, C.M.: Extensible Markup Language (XML) 1.0 (Second Edition), 6. Notation, W3C Recommendation, 6 October 2000.

[5] Chall, J.S. and Dale, E.: *Readability Revisited: The New Dale-Chall Readability Formula*, Brookline Books, 1995.

[2)] [13] を指すものと思われる

[3)] 「ならば」については「→」も用いられている．

[6] Crocker, D. and Overell, P.: Augmented BNF for Syntax Specifications: ABNF, RFC 2234, November 1997.

[7] Crossmann, E.R.F.W.: A Theory of the Acquisition of Speed-Skill, *Ergonomics*, Vol. 2, pp. 153–166, 1959.

[8] Genzten, G.: Untersuchungen über das logische Schließen, *Mathematische Zeitshrift*, **39**, pp. 176–210, 405–431, 1935.

[9] Genzten, G.: *The Collected Papers of Gerhard Gentzen*, North-Holland, 1969.

[10] Gödel, K.: Über formal unentscheidbare Sätze der Principia mathematica und verwandter Systeme I, *Monatshefte für Mathematik und Physik*, **38**, pp. 173–198, 1931.

[11] Heijenoort, J.: *From Frege to Gödel: A Source Book in Mathematical Logic, 1879–1931*, Harvard University Press, 1967.

[12] Heyting, A.: Die formalen Regeln der intuitionistischen Logik, *Sitzungsberichte der Preussischen Akademie der Wissenshaften, physikalisch-mathematische Klasse*, pp. 42–71, 158–169, 1930.

[13] Hilbert, D. and Ackermann, W.: *Grundzüge der theoretischen Logik (1. Aufl.)*, Springer, 1928.

[14] Hilbert, D. and Ackermann, W.: *Grundzüge der theoretischen Logik (4. Aufl.)*, Springer, 1958.

[15] JIS X 3001-1994 プログラム言語 Fortran (ISO/IEC 1539:1991)

[16] 情報科学辞典 第一版, pp. 481–499, 「双対性」, 岩波書店, 1990.

[17] Kernighan, B.W. and Ritchie, D.M., 石田晴久訳: プログラミング言語 C 第 2 版, 共立出版, 1994.

[18] 清兼義弘, 末廣陽一: 国際化プログラミング, 共立出版, 1998.

[19] Kleene, S.C.: *Introduction to Metamathmatics*, North-Holland, 1967.

[20] 中川徹, 小柳義夫: 最小二乗法による実験データの解析, 東京大学出版, 1982.

[21] 中嶋章, 榛澤正男: 継電器回路に於ける単部分路の等価変換の理論, 電信電話学会雑誌, Vol. 165, pp. 1089–1093, 1936.

[22] 中嶋章: 継電器回路に於ける負性四端子回路網の Transfer Impedance に就いて, 電気通信学会雑誌, Vol. 179, pp. 90–98, 1938.

[23] Peano, G.: Arithmetices principia, nova methodo exposita, Turin, 1889. ([11] pp. 83–97.)

[24] Schütte, K.: *Beweistheorie*, Grudlehren der mathematischen Wissenschaften, Band 103, Springer, 1960.

[25] Schütte, K.: *Proof Theory*, Grudlehren der mathematischen Wissenschaften, Band 225, Springer, 1977.

[26] Shannon, C.: A Symbolic Analysis of Relay and Switching Circuits, *Trans. AIEE*, Vol. 57, pp. 713–723, 1938.

[27] Shannon, C.: The Synthesis of Two-Terminal Switching Circuits, *The Bell System Tech. J.*, Vol. 28, pp. 59–98, 1949.

[28] Sheffer, H.M.: A Set of Five Independent Postulates for Boolean Algebras, *Trans. Amer. Math. Soc.*, Vol. 14, pp. 481–488, 1913.

[29] 芝祐順: 読み易さの測り方 クローズ法の日本語への適用, 心理学研究, Vol. 28, No. 2, pp. 67–73, 1957.

[30] Shoenfield, J.R.: *Mathematical Logic*, Addison-Weseley, 1967.

[31] 鈴木康彦, 野口健一郎, 後藤英一: Java をターゲットにした自国語プログラミングの実験, 情報処理学会 第 58 回全国大会, 5M-8, March 1999.

[32] Takeuti, G. (竹内外史): *Proof Theory*, Studies in Logic and the Foundations of Mathematics, Vol. 81, North-Holland, 1975.

[33] 建石由佳, 小野芳彦, 山田尚勇: 日本文の読みやすさの評価式, 情報処理研究報告, HI-18-4, 1988.

[34] Taylor, W. L.: Cloze Procedure: A New Tool for Measuring Readability, *Journalism Quarterly*, Fall 1953.

[35] The Java Language Specification: http://java.sun.com/

[36] The Unicode Consortium: *The Unicode Standard, Version 3.0*, Addison-Wesley, Reading, MA, 2000.

[37] Whitehead, A.N. and Russell, B.: *Principia Mathematica*, Vol. 1, Cambridge University Press, Cambridge, l910.