4. U. L. の簡単な formula の機械によるproof の作成について

杉 林 益 太 郎

$$A_1 R_1 A_2 \cdots\cdots A_n R_n A_{n+1} \qquad \cdots\cdots\cdots\cdots\cdots\cdots \text{(F)}$$

をＵＬの formula とする。

但し

R_i ($i=1, 2, \cdots\cdots, n$) は∧, ∨, →, ≡, ∧¬, ∨¬, →¬, ≡¬, のうちのいずれかを示し $A \cdot B$ が formulasのとき $A \wedge \neg B$ は¬ ($A \wedge B$) を表わす。∨¬, →¬, ≡¬についても同様である。

A_i ($i=1, 2, \cdots\cdots, n+1$) は $a \tau b$ の形の formula であつて a と b は independent variable, τ は ε, $=$, $\subseteq$, $\not\varepsilon$, $\neq$, $\not\subseteq$ のうちいずれかを示す。

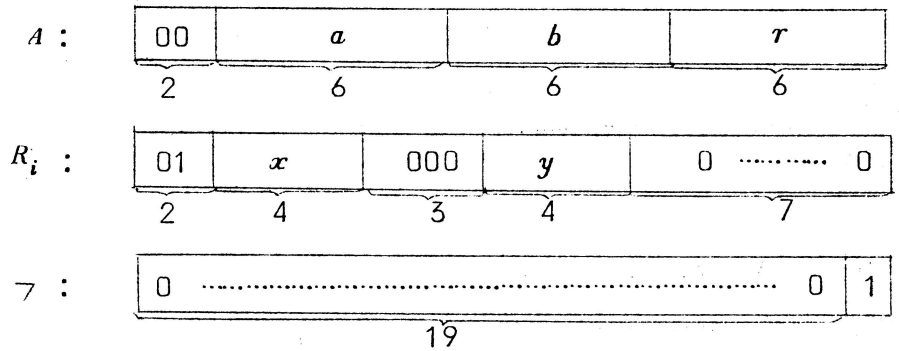
以下で A, B 又はそれらに添字をつけたものはすべて $a \tau b$ の形の formula をあらわす。また R_i には結合の順序により一番内側のものに番号1，以下外側のものにうつるに従つて，順次大きい番号（ x とする）を与える。但し，同一番号のものが二個以上出来るときはそれらについては例えば位置が左から右へ移るに従つて $x+1$, $x+2$, …… などと番号をつけかえる。それにともなつてその他の番号もつけかえる。

n の大きさは原理的に任意であるが記憶容量による制限から余り大きくとれない。(2048 words (half word) のもので約２０～３０である）

(F)は quantifier を含んでいないが簡単なものについては含んでいても取扱いは変らない。また assertion $\sigma \mid H$ は $\sigma \rightarrow H$ とかくことにし σ に属する個々の formulas を∧でむすべば $\sigma \mid H$ のあるもの

は(F)の一種と見做せる。この場合Hの proof constituent を最上段におく。

次にA, R_i を表わすには夫々20 bits(half word) ずつを用いる。＞（$A \wedge B$）のように＞を用いるときには＞にも20 bits を与える。そしてA, $\wedge$, B＞の順に格納する。夫々の構成は下図に示す。



但し下段の数字は使用桁数，xは R_i のもつ番号，yは R_i の種類を示す。

(F)は適当な storage locations l, $l+1$, ……, $l+2n$ へ, A_1, R_1, A_2, ……, A_{n+1} の順に store する。簡単のためl, $l+1$, ……, $2n+1$ を第一列とよぶ。従つて第L列は $l+2(L-1)(n+1)$, ……, $l+2L(n+1)-1$ を意味する。またlは規準点と仮称し偶数番地である。

proof の作成を次の5段階に分ける。

(1)……proof formula と proof constituent の association を各 string について行い R_i について分解可能な formula はすべて分解する。

(2)……(1)で得た tree を A_i（$i=1, 2, \ldots\ldots, n$）あるいはその否定のみを含む簡単な tree に reduce する。

(3)……(2)で得た tree について cancelling pair を見つける。

(4)……cancelling pair のないすべての strings のAをrについて分解し primitive formulas のみからなる新しい tree

を作る。

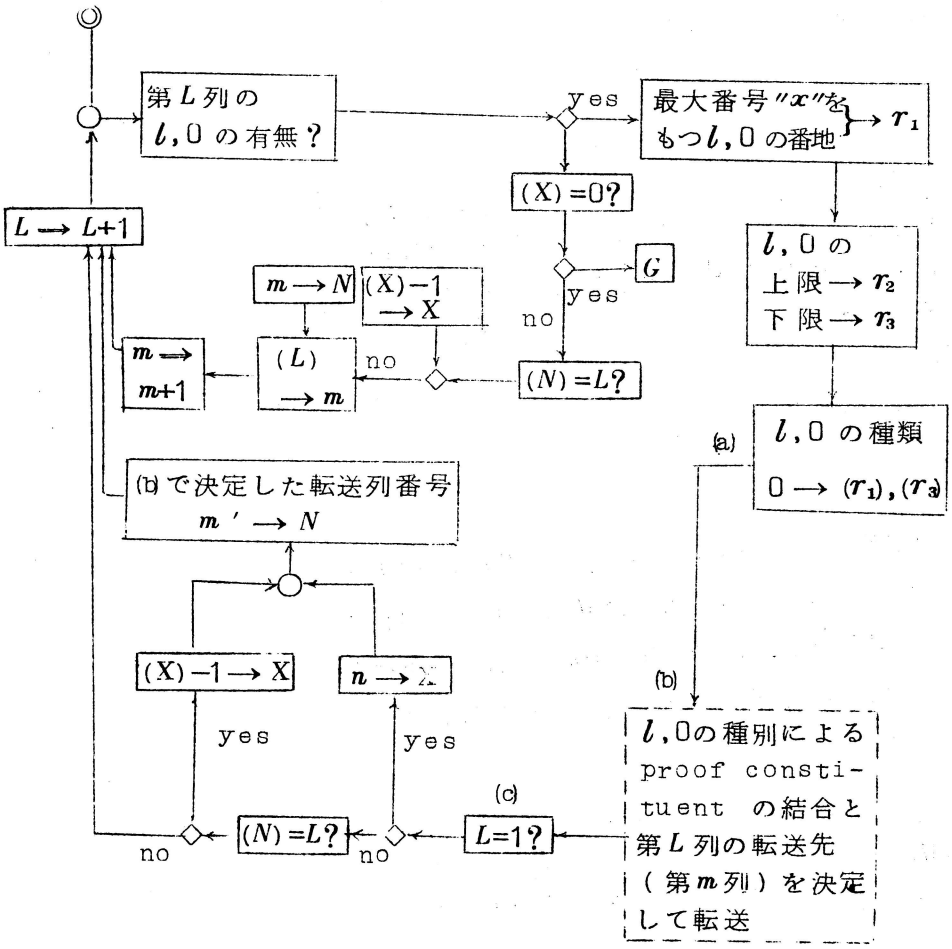
(5)……(4)で得たtreeについてcancelling pairを見つける。

印刷は(2)と(4)の作業と平行に行う。

(1), (2)について

(2)でいうreduce treeと，ある対応をもつ A_i （$i=1,2,\cdots\cdots,n$）あるいはその否定のみを含む図形の得られる過程を次の基本的流れ図(I)－(III)によつて示す。

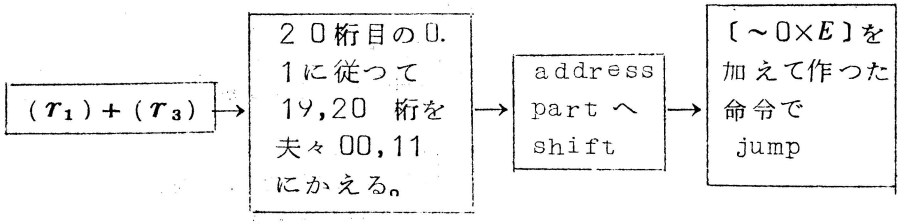
基本的流れ図 (I)



(*)はstorage location *に入つている内容，□内の"→"は数値は転送（従つて → の左側はmemoryの内容，右側は番地又は列番号である）"→"は数値の変更 □ の間にある矢印は作業の進行方向を夫々示す。l，0はlogical operaterの略である。以後も同様に用いる。図中のm'は転送列が2個あるとき大きい方の列番号をあらわす。

1.0. の上限とは$L=1$のときの規準点，Lア1のときは「第L列の先頭」又は「（r_1）から先頭の方向へ"0"もしくは"ア"に出合うまで進み最初に出合つた"0"もしくは"ア"の入つている番地に1を加えたもの」下限とは（r_1）から前記と反対方向へ進み最初に出合う"0"または"ア"の入つている番地のことである。

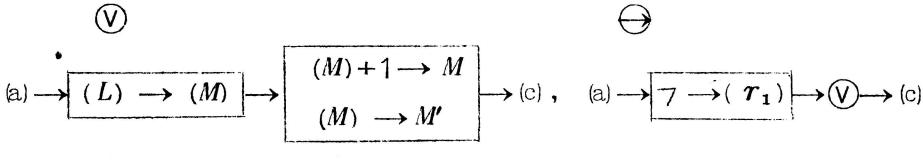
(II)−(a) の説明



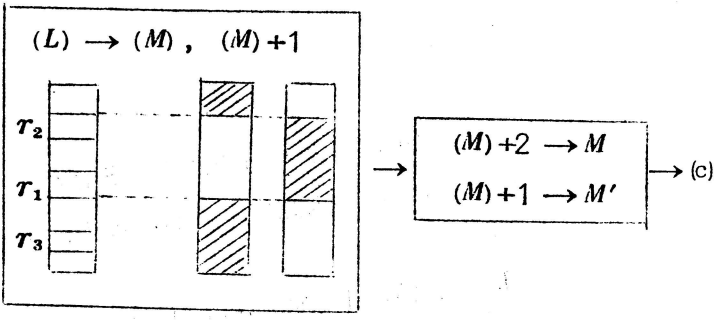
（r_3）は"0" か ">" である。

(III)−(b) の説明

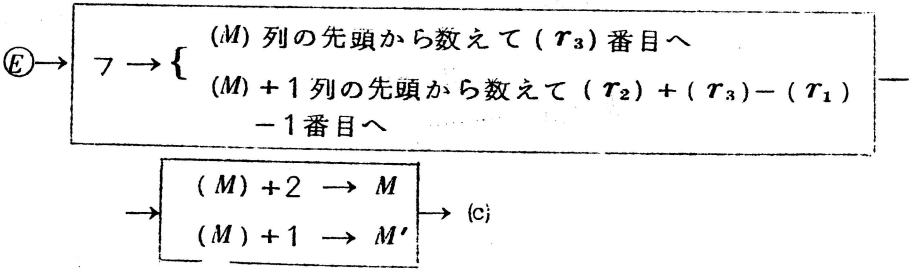
logical operatorを処理する命令群mを(R)で示すRの処理結果を転送すべき列番号（2個あるときは小さい方）はMにstoreされている。Mの初期値は2，またM'はW.S.で（M'）$=m'$である。更に(R)の図で転送後の斜線の部分は転送前と同じもの，空白は0であることを示す。



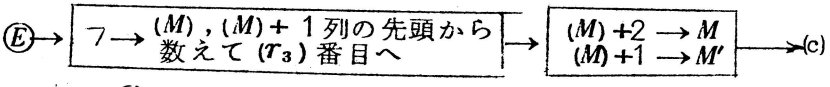
Ⓐ



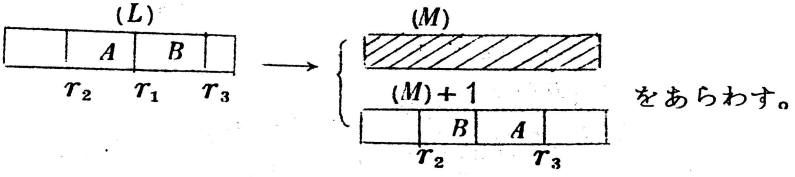
Ⓔ



≡ （≠と同じ）

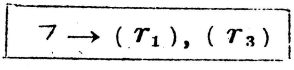


但し Ⓔは

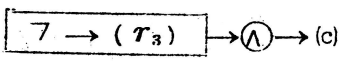


をあらわす。

∨ㄱ



→ㄱ



(A7)

$\boxed{\neg \rightarrow (r_1),\ (r_3)} \rightarrow$ (V) $\rightarrow$ (c)

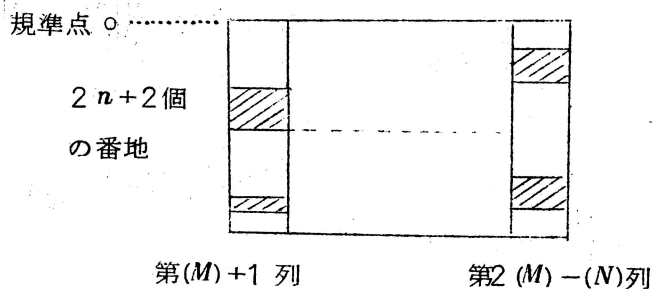
(a) の説明の補足

R が以上に示したものと異なる種類のときはすべて同じ番地へ jump する。これは(F)がU.L. の formula でないことを意味する。

流れ図(I)に要する命令は約２３０語であつて，G 点で終る。これ以後の program は未完であるが，以下にその方針について記す。

1. 第 $(N)+1$ 列より第 (M) 列までを順に第 $(M)+1$ 列より第２ $(M)-(N)$ 列へ送る。A，¬ と続いているものについては，¬ を A に加えて ¬ を取去る。第 $(M)+1$ 列 ～ 第 $2(M)-(N)$ 列に下に例示するような図形 T を得る。

図 形 T

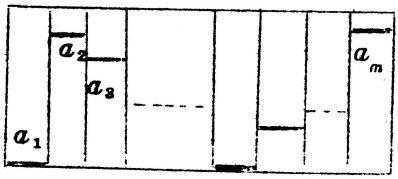


斜線の部分は A を示す。

2. T の整理

T の第 p 行（第 q 列）（各列の先頭からの位置が同じものを，列の順序に並べて「行」とする（注意参照））にある A を A_{pq} とする。今第 q 列で A_{pq} の上方に A がないとき，あるいは A_{rq} と A_{pq} （$1 \leqq r < p$）の間に A がないとき l の増加と共に r が減少すれば A_{rq+l} を第 p 行へ下げる。（l を増加したとき $r \geqq p$ となるときも含めて）r が増

加すれば $A_{r\,q+l}$ を出発点として上の操作をくり返す。これを「列」については左（$q=1$）から順に，「行」については下（$p=n+1$）から順に行い各 A_{ij} の移動を１回に限れば例えばＴは下図に示す如く配列される。



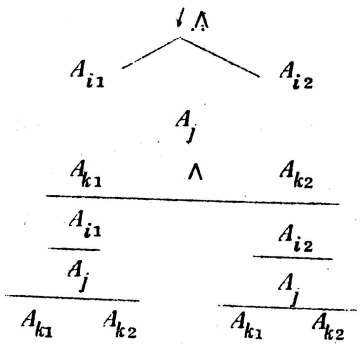
太線 a_1，……，a_m は同じ行に下りて来た A_{ij} の集りを表わす（a_i には左から番号をつける）

次に第 $n+1$ 行の a_i と a_j（$i<j$）の間に a_k（$i<k<j$）がなければ a_{i+1}，a_{i+2}，……，a_{j-1} を a_i の上方に平行移動する。整理の結果を印刷して⑵を終る。

○注意 A_{pq} は $\boxed{\begin{array}{c}A\\ \hline 0\end{array}}$ の形である。従つて行の数は $n+1$ 行である。
また整理の前に $\boxed{\begin{array}{c}A_i\\ \hline 0\\ \hline A_j\\ \hline 0\end{array}}$ の形をしているものは整理の際位置を（最下段にある）A_j の位置で代表させる。（移動にあたつては，組にしたまま移動する。３段以上になつているときも同様である）

3. ⑶について

例えば整理の結果 a_i の上方に a_j，a_k があるとし a_i が A_{i1}，A_{i2}，a_j が A_j，a_k が A_{k1}，A_{k2} を夫々含むとする。このとき各 a 内の A を Λ で結合したものを top formulas とする。tree



あるいはinterchange の ruleに従つて変換したtree を対応させる。但し(1)の一番外側の R_i が→ のときにはinterchange は premissesに属するものと conclusion に属するものの間では行わない。そして conclusion に属するものから出来るtree の bottom formulas の下へ premises の formula を接続する。

かくして出来たtree について cancelling pair をもつstring を消去する。この操作を第 $n+1$ 行にある各 a_i について行う。以上で(3)を終る。

4. (4), (5)について

A は $a\,r\,b$ なる形の formula であるから容易に primitive formula に直せる。A のみを含む tree から primitive formula のみを含む tree への変換及びその後の cancelling pair の決定は(1)～(3)の方法を殆んどそのまま使用出来る。

(1)～(5)の programming の主要部分は(1)と(2)である。

1，2，3，4項についての詳細及び quantifier を含んだ任意の formula については次の機会に報告致します。

参　考　文　献

Sigekatsu Kuroda
An investigation on the logical structure of mathematics.
(I) A logical system, Hamb. abh. 22(1958)
(III) Fundamental deductions, Nagoya Math.Journ.13(1958)

本 PDF ファイルは 1960 年発行の「第 1 回プログラミング–シンポジウム報告集」をスキャンし、項目ごとに整理して、情報処理学会電子図書館「情報学広場」に掲載するものです。

この出版物は情報処理学会への著作権譲渡がなされていませんが、情報処理学会公式 Web サイトの `https://www.ipsj.or.jp/topics/Past_reports.html` に下記「過去のプログラミング・シンポジウム報告集の利用許諾について」を掲載して、権利者の捜索をおこないました。そのうえで同意をいただいたもの、お申し出のなかったものを掲載しています。

過去のプログラミング・シンポジウム報告集の利用許諾について

情報処理学会発行の出版物著作権は平成 12 年から情報処理学会著作権規程に従い、学会に帰属することになっています。

プログラミング・シンポジウムの報告集は、情報処理学会と設立の事情が異なるため、この改訂がシンポジウム内部で徹底しておらず、情報処理学会の他の出版物が情報学広場 (=情報処理学会電子図書館) で公開されているにも拘らず、古い報告集には公開されていないものが少からずありました。

プログラミング・シンポジウムは昭和 59 年に情報処理学会の一部門になりましたが、それ以前の報告集も含め、この度学会の他の出版物と同様の扱いにしたいと考えます。過去のすべての報告集の論文について、著作権者（論文を執筆された故人の相続人）を探し出して利用許諾に関する同意を頂くことは困難ですので、一定期間の権利者捜索の努力をしたうえで、著作権者が見つからない場合も論文を情報学広場に掲載させていただきたいと思います。その後、著作権者が発見され、情報学広場への掲載の継続に同意が得られなかった場合には、当該論文については、掲載を停止致します。

この措置にご意見のある方は、プログラミング・シンポジウムの辻尚史運営委員長 (`tsuji@math.s.chiba-u.ac.jp`) までお申し出ください。

加えて、著作権者について情報をお持ちの方は事務局まで情報をお寄せくださいますようお願い申し上げます。

期間：2020 年 12 月 18 日 〜 2021 年 3 月 19 日
掲載日：2020 年 12 月 18 日

プログラミング・シンポジウム委員会

情報処理学会著作権規程
`https://www.ipsj.or.jp/copyright/ronbun/copyright.html`