# 文 献 紹 介

## 103. オートマトンの正規表現とその応用

J.A. Brzozowski: A survey of regular expressions and their applications [IRE Trans, EC-11 No. 3, June, 1962, pp. 324～335]

Kleene が 1956 年に有限オートマトンの動作と正規表現の間に 1 対 1 の対応関係が存在することを示して以来，この問題について多数の研究報告がある．本論文は，これらの諸報告をもとに，できるだけ統一的に正規表現の解説を行なおうとするものである．

まず正規表現の利点として，(1) 非常に一般的であること，(2) 言語表現に類似していること，(3) 正確であること，(4) 一行にかけること，を指摘している．

次に入力事象，正規表現の定義を述べ，正規事象の種類分けを行なっている．決定事象，初期単純決定事象，初期事象，単純決定事象，複合決定事象などを挙げている．

さらにオートマトンの定義と，決定事象をオートマトンで実現する方法を示している．次に一般の正規事象のオートマトンによる表現を論じている．

正規表現と，状態図による表現との対応関係を求めるために，状態図から正規表現，逆に，正規表現から状態図を導く手法を示している．

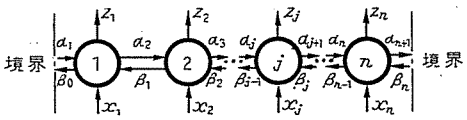
最後に，McNaughton & Yamada の提案にかかる正規表現での作用素の中に ( )′ と & を追加すること，変則な状態図，正規表現と可変長符号との関連などにも簡単に触れている． (戸田 巌)

## 104. 組合わせ単位回路で構成された繰り返し論理回路網

W. Kilmer: Iterative Switching Networks Composed of Combinational Cells [IRE Trans. EC-11 No. 2, April, 1962 pp. 123～131]

組合わせ論理回路を繰り返し，カスケードに接続し，双方向に情報を伝達する場合，ある条件を満たすと，その回路網は順序回路として動作する．すなわち，記憶作用がある．

Hennie と同様，図に示すような一次元の双方向繰り返し論理回路をモデルとして扱い，これが次の仕様を満足する時 BITN と呼ぶ．




1) 一つの論理単位回路は組合わせ的である．ただし，回路網としてはこの限りでない．つまり，$(z_j, \alpha_{j+1}, \beta_{j-1})$ は完全に $(x_j, \alpha_j, \beta_j)$ で定まる．

2) 各単位回路の入力が入ってから，出力が出るまでには単位時間の遅れがあり，同期的に働く．

3) 各単位間の結合線は離散的な情報を遅れなしに伝達する．

BITN には次の三つの型がある．

1) $\alpha$, $\beta$ 双方共に 相手に独立なもの (両独立 B-ITN)

2) $\alpha$, $\beta$ の片方は相手に独立で，片方はその影響を受けるもの (片従 BITN)

3) $\alpha$, $\beta$ 双方ともに，相手の影響を受けるもの(両従属 BITN)

1),2) は定常状態で唯一つの状態を持つから，記憶作用がない． 3) の場合には定常状態で記憶作用がない場合とある場合，巡還を生じて不安定な場合の三つがある．

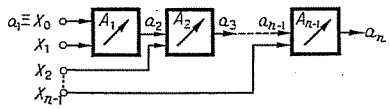
本論文では定常状態で記憶作用がある場合の性質を 5 個の定理の形で記述し，巡還におちいる場合の性質で，再帰的に解けないという Hennie が得たのと同じ定理を二つあげている． (伊吹公夫)

## 105. 2 入力可変単位のカスケード論理回路網

K.K. Maitra: Cascaded Switching Networks of Two-Input Flexible Cells [IRE Trans. EC-11 No. 2, April, 1962, pp. 136～143]

可変論理回路は将来，自己構成機械の分野に応用されるものとして有望である．この論文では任意の 2 入力 1 出力の函数値がとり得る可変論理回路の基本単位を図のようにカスケードにつないだとき，どのようなものが得られるかを論じている．

このような構成の機械を $n$-place 函数と呼ぶ．この方法では n 変数函数 $2^{2n}$ 個中，わずか $(2/5)6^n + 8/5$ 個しか実現できないことを示している．そして，




任意の函数が $n$-place 函数として実現できるか否かの検査法を具体例で説明している．

また実現可能な場合に，どのような函数を各基本単位に与えるべきか，表を用いて合成する手続きを説明している．これによれば，右側の基本単位から順に合成していくのであるが，隣の基本単位から入る入力に制限があるので，その性質を利用して，可能な函数をしぼっていく．つまり，$a_n$ と $A_{n-1}$ の値が一致しなければならない．$a_n$ と $A_{n-1}$ が不一致であるような部分を消していくのである．　（伊吹公夫）

## 106. 予測子修正子法の安定性

P.E. Chase: Stability Properties of Predictor-Corrector Methods for Ordinary Differential Equations 〔J.A.C.M., Vol. 9, No. 4, Oct. 1962, pp. 457～468〕

常微分方程式の初期値問題に対する数値解法として，予測子修正子法が好まれるのは，導函数計算の回数が少なくてすむのが主な理由であるが，もし収束するまでくりかえして修正子を使うとすれば，導函数計算の回数も増え，Runge-Kutta-Gill 法に対する利点も失われる．ところが，従来の予測子修正子法での安定論義は修正子公式についてだけ考えており，修正子公式を何回もくりかえした場合に対しての議論に相当する．この方法の特徴を生かすには，修正子公式の使用を，できれば1回ですませるようにしなければならないが，その場合には当然予測子のもつ誤差が問題になって来る．この論文はそのような場合の安定性を，Milne の方法と Hamming の方法とについて議論している．

その結果によると，Hamming の方法でも，修正子を1回だけ使用する場合（場合1）と，予測値を前段での予測された打切り誤差で補正して，あと1回だけ修正する場合（場合2）ともに不安定を示すような $\bar{h}$（きざみ幅 $h$ と $f_y$ との積）の領域が出て来る．また一方 Milne の方法では場合 1，2 ともに安定な領域も出て来ることがわかった（場合1で $-0.8<\bar{h}<-0.3$ の間）．結局修正を1回ですますためには Hamming 法での場合2が多分一番よいだろうということ，Milue法では場合1が此と同じ程度によかろうという結論である．

なお $y'=f(x, y)=-100y+100$, $y(0)=0$ という例での数値例がのべてある．　（伊吹公夫）

## 107. 切換え式分類――マージングの一手法

S. Sobel: Oscillating Sort: A new sort merge technique. 〔J.A.C.M. Vol. 9 No. 3, July 1962, pp. 372～374〕

内部分類とマージ分類とをまぜ合せて行なう方法で，その手順は次のとおりである．

まず，内部分類により $N-1$ 個の系連を作り，$N-1$ 個のテープに一系連ずつ置く．次にその $N-1$ 個のテープを逆方向読み取りを行ないながら $N$ 番目のテープにマージし，最後にテープマークを打つ．ここで再び内部分類に移って最初の系連をその $N$ 番目のテープへ入れ，続く $N-2$ 個の系連を残った $N-1$ 個のテープのうちの $N-2$ 個へ入れる．これらの $N-1$ 個のテープは残り1個のテープへマージされる．この手順を繰り返して $N-1$ 個のテープの各々がマージされた $N-1$ 個の序列を持つまで行なわれると，それをマージして，長さ $(N-1)^2$ の序列を得る．

テープ数 $N=4$ の場合に 27 個の序列をマージする例について，第1図に示す．また切換え式分類のマージパワーを他のマージ法と比較したものを第1表に示す．これから明らかな如く，切換え式分類は入力テープに1個だけ多くのテープを要するが，そのパワーは，他のものより大である．

OSCILLATING SORT

| | | | | |
|---|---|---|---|---|
| 1 | 1 | 1 | 0 | after first internal sort phase (3 internal sort sequences generated) |
| 0 | 0 | 0 | 3 | after first merge operation (3 internal sort sequences merged to form 1 tape sequence) |
| 0 | 1 | 1 | 3 | after next internal sort phase (6 internal sort sequences contained in 4 tape sequences) |
| | | | 1 | |
| 3 | 0 | 0 | 3 | after next merge operation |
| 3 | 3 | 0 | 3 | after next merge operation now no internal sort. |
| 0 | 0 | 9 | 0 | after next merge operation |
| 1 | 1 | 9 | 0 | after next internal sort phase |
| | | 1 | | |
| 0 | 0 | 9 | 3 | after next merge operation |
| 0 | 3 | 9 | 9 | after next merge operation |
| | | 3 | 3 | |
| 9 | 0 | 9 | 9 | after next merge operation |
| 0 | 27 | 0 | 0 | after final merge operation (all 27 internal sort sequences merged into 1 tape sequence) |

第1図

第1表

| Number of merging tapes | Power of merges | | | | |
|---|---|---|---|---|---|
| | Balanced | Cascade | Polyphase | Oscillating | Oscillating with one less merging tape |
| 3 | 1.5 | 1.64 | 1.65 | 2 | — |
| 4 | 2 | 2.32 | 2.10 | 3 | 2 |
| 5 | 2.5 | 2.99 | 2.42 | 4 | 3 |
| 6 | 3 | 3.71 | 2.66 | 5 | 4 |
| 7 | 3.5 | 4.39 | 2.82 | 6 | 5 |
| 8 | 4 | 5.09 | 2.95 | 7 | 6 |

（鵜飼直哉）

## 108. ＜概念＞を記述するリスト

R.B. Banerji: The Description List of Concepts [Comm. ACM, Vol. 5 No. 11, 1962, pp. 426～432]

本論文は，「人工知能」とまでは行かなくても，「心理現象のシミュレーション」として，＜概念＞の記述およびその処理の方法について述べている．

ここで＜概念＞とは，おおまかにいえば，ある性質をもった対象の集合を指している．たとえば ＜赤い物＞という概念は＜赤い＞という性質をもった物全体，ポスト，リンゴ，ホッペタなどのことである．そういった概念を定義するには，その本質上，回帰的（recursive）で，また定義の複雑さも可変であるから，リスト的な記述をしなければならない．言い換えれば，概念そのものが，回帰的でリスト的な構造を持っているとみられる．

さて本文に即していえば，対象のすべての集合を＜宇宙＞（universe）と呼び，その宇宙の，互に素で，網羅的な分割を＜性質＞（property）と呼ぶ．その分割に属する部分集合をその性質の＜値＞（value）と呼ぶ．たとえば，＜色＞という性質は全宇宙を，＜赤い物＞ ＜青い物＞ 等々に分ける分割であり，＜赤いもの＞，＜青い物＞（それぞれは宇宙の部分集合）などを＜色＞の値と呼ぶのである．

そこで＜概念＞の定義は

（i） 宇宙の性質の値は，概念である．

（ii） 二つの概念 A，B の和集合 A ∪ B もまた，概念である．

（iii） 二つの概念 A，B の共通集合 A ∩ B もまた，概念である．

このような概念は，リスト的に（たとえば IPL V などのリスト言語で）記述されるが，図的には次のように書かれる．

name of concept—universe
　　|
　　name of property— name of value
　　　　(name of concept)

例をあげれば

ball—universe
　　|
　　shape—round (ball)

もっと複雑な例として nice の定義（ここで nice なものとは，赤くて丸いものとする）

nice—universe
　　|
　　color—red (nice)
　　|
　　shape—round (nice)

A ∪ B，A ∩ B などの複合概念のリスト的記述を，もとの A，B のリスト的記述から形成するプログラム（の流れ図）が与えられている．

次に，このような記述法で生じる冗長（非能率）の問題を検討し，また，今後の問題を論じている．

（淵　一博）

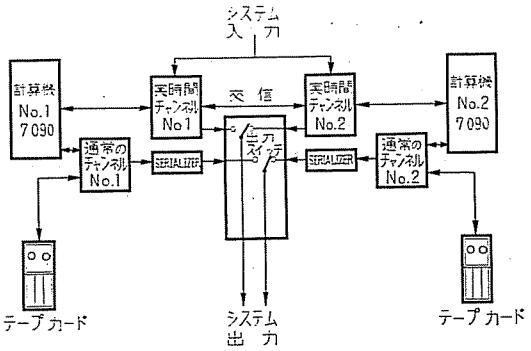
## 109. 2台の計算機を複合した組織のためのプログラム

J. Dow: Programming a Duplex Computer System [Comm. ACM, Vol. 4 No. 11, Nov. pp. 507～513]

この論文は，コアからコアへの情報直接移送装置を使い2台の計算機を同時に働かせるプログラムに関して述べたもので，BMEWS（米空軍弾道ミサイル早期発見組織）用の計算機（IBM 7090 2台）のプログラムとして Sylvania Elec. Co. で開発したものである．

計算機を複合させるのは極度に高い信頼度を得るためである．3台の計算機を用いれば多数決論理で自動訂正が行えるがコストの点で難点があり，2台でも実用的には十分である（この場合はどちらが正しいかは人間が判断する）．

機械の構成は第1図に示すとおりで，全く同一の2



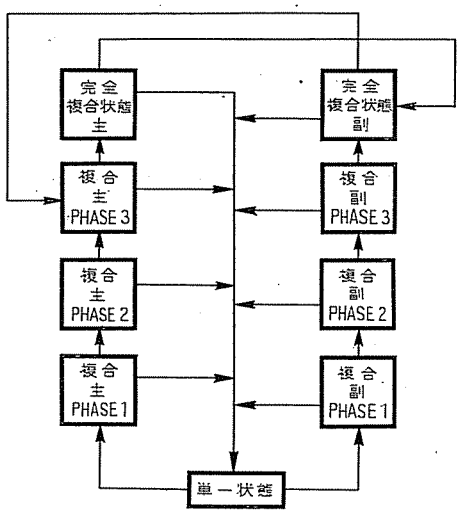

第1図　複合計算機の機械構成

組の計算機からなっている．各計算機は通常のI/Oチャンネルのほかに実時間チャンネルを持っていて，これを通じ外部入力が入ってくると同時に互いに他を自己のI/O装置であるとみなす．すなわちコアの中の1語36ビットを他方のコアへ33 μsで移すことができる．なお，おもな命令の処理時間は4.4 μsである．伝送は同時には一方向のみである．

出力はスイッチでどちらかの計算機に切りかえられる．出力に接続されている方を主計算機(active)，そうでない方を副計算機(stand by)と呼ぶ．計算は双方で同時に行なわれるが，主計算機は計算がすむと結果を送る．副計算機はそれを自身の結果と比べてチェックする．

動作の状態は三つあり，プログラムも異なっている．第1は複合(duplex)プログラムの状態で，これが正常の状態である．第2は単一(simplex)プログラムの状態で，前に両計算機の不一致があったなどのため，一時的に両計算機が完全に独立に動いている状態である．第3は複合準備(duplex initialize)プログラムの状態である．

複合準備の場合，片方は過去のデータを失っているので，それを補わなければ双方の結果は一致しない．方法は三つあって，(1) 双方とも過去のデータを無視させる，(2) 古い計算機と同じ情報を新計算機に移す，(3) 新計算機が古い計算機と同程度の情報を得るまで待つ．これは古いデータがあまり長い間影響しない場合に使える．ここでは第3の方法を使っている．




第2図　複合プログラムの状態
(ただし，矢印は各種原因による状態の変化を示す)

最初両計算機は単一プログラム状態でスタートする．数ms経って結果がない方が副計算機となり，続いて複合準備状態に入る．ここではphaseが三つあって，phase 3で計算結果は等しくなる．複合状態では，主計算機で結果が得られる度に，それが副計算機に送られ比較される．不一致の時は信号を発し，オペレータは切りかえを決定する．

なお両計算機に共通あるいは他方から制御可能なインデケータ，レジスタ，コアがあれば極めて有効であろう．　(岩城三郎)

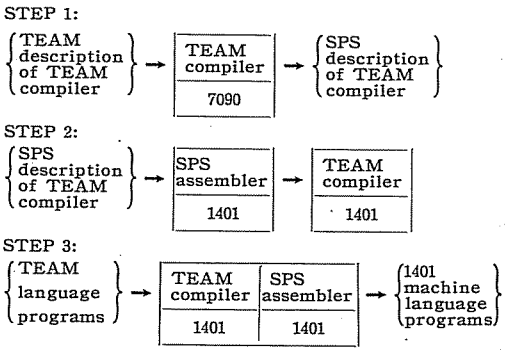
## 110. NELLIACで発生された7090-1401コンパイラ

J.B. Watt and W.H. Wattenburg: A NELLIAC-generated 7090-1401 Compiler [Comm. ACM, Vol. 5 Feb. 1962, pp. 101～102]

NELLIACというのは，ALGOLの一方言であり，7090 NELLIACコンパイラは既に開発されている．この段階で，IBM 1401のための新しい(言語のための)コンパイラTEAMをどう開発したか，その手順がここで報告されている．

眼目は，TEAMコンパイラをNELLIAC(という問題向言語)で書き，それを7090-NELLIACコンパイラでコンパイルした，ということである．その結果できたものは，7090でのTEAMコンパイラである．

今後の予定は．TEAMコンパイラをTEAMという言語で書く．そして，それから1401における1401-TEAMコンパイラを作ることであるとして，その手順を述べている(第1図)．




第1図

コンパイラを問題向言語で書くことの利点は，生産性の向上(1人8週間でTEAMコンパイラをNE-

LLIAC で書き検査した．コンパイルのための計算時間は 7090 で計 50 分)．文書化の利益（問題向言語で書かれているから，それで許されるコメントと合わせて，コンパイラの保守，改訂が容易になる）などである．

また，このような方式が，ソフトウェアの開発の能率的な方式を示唆し，一般に資するであろうともいっている．

（紹介者註：プログラミングのためのプログラム，すなわち，コンパイラやインタプリータの相互関係を，それらの表現，表現形態の変換の点から，一般的に論ずることが可能である）　（淵　一博）

## 111. ATLAS の管理プログラム

T. Kilburn, R.B. Payne and D.J. Howarth: The ATLAS Supervisor [Proc. E.J.C.C. Vol. 20, Dec. 1961, pp. 279～294]

Manchester 大学と Ferranti 社が共同で開発している大形高速計算機 ATLAS の管理プログラム（Supervisor）のシステムを金物と関係づけて説明している．

管理プログラムが動作状態になるには，次の三つの原因がある．すなわち，（1）目的プログラムが自分で管理プログラムに入る（割出し），（2）依頼された仕事が終った金物（主として入出力機器）からの終了信号（割込み），（3）予測できない事故または誤働作（割込み）である．

割出しの場合は，目的プログラムを制御していた主制御装置は一時休止し，管理プログラムを制御する別の extracode control が動作状態に入る．割込みの場合には，割込信号によってまず interrupt control が働き，割込みの原因を調べ，それによって必要な処置をする．interrupt control が動作状態にある時には，別の割込原因が生じても，割込は禁止される．割込み原因に対する処置が簡単な場合は interrupt control だけを使った，割込ルーチンを経て，再び目的プログラムに戻るが，処置が複雑な時は，extracode control が動作状態になるように要求される．

extracode control は管理プログラム以外に，固定記憶装置内のサブルーチンを使う時にも動作状態になる．

管理プログラム全体を総括している co-ordinator ルーチンがあり，このルーチンは管理プログラムの各部分（S.E.R. と略称する）の順番や待ち合せなどの情報を記録したリストを持っている．管理プログラムへの出入の場合には必ず，この co-ordinator ルーチンを経由する．

管理プログラム全体が洩らさず説明されているが，特に説明の重点が置かれているのは次のルーチンである．

（1） 磁心記憶装置と磁気ドラム記憶装置との間の情報の転送に関するルーチン

ATLAS はページ・アドレス方式を採用しているから，それに対応して管理プログラムには ATLAS 特有のルーチンがある．ただし，ページの入れ替えに関する学習プログラムの詳細は，ここでは述べられていない．

（2） 磁気テープ装置の実施ルーチン

入出力機器を制御するのは管理プログラムの役目であり，目的プログラムは管理プログラムに仕事を依頼する．

その他，ATLAS を運転するための operating system の概要も述べられている．　（西野博二）

## 112. ALGOL 60 の構造と用法

H. Bottenbruch: Structure and Use of ALGOL 60 [J.A.C.M. Vol. 9 No. 2, April 1962. pp. 161～221]

論理的，形式的に簡潔に記述された ALGOL レポートに対して，ALGOL の書き方およびその言語構造をわかりやすく講義式に解説する．本文の構成は次のとおり．

(1) ALGOL の非形式的記述　(2) 種々の特徴
(3) ALGOL の形式的記述　(4) Procedures
(5) ALGOL Procedure の数例．

はじめの 2 章は procedure 以外の概念を簡単な例題を掲げながら，特にブロック構造については，それの導入されねばならぬ理由も入れて，詳しく説明される．

3 章では ALGOL 文法の形式的定義を与えるが，メタ言語は用いない．たとえば，無条件 statement ***U*** は，

“条件 statement でもなく，ラベル付条件 statement でもない statement” であり，条件 statement は，“***B*** を論理式，***U*** を無条件 statement，***S*** を statement として，if ***B*** then ***U*** または if ***B*** then ***U*** else ***S*** なる形をした statement” と定義される．これらの定義が回帰的になされざるをえないことは本質的である．

4 章は procedure の説明で，specification と declaration と混同するようなことがないように明確に

述べられる.

最後の procedure の例題はすべて“まぜ合わせ”と“分類”とにあてられている. (丸山 武)

## 113. ALGOL 60 の文法チャート

W. Taylor, L. Turner and R. Waychoff: A Syntactical Chart of ALGOL 60 [Comm. ACM, Vol. 4 No. 9, Sept. 1961, p. 393]

ALGOL 60 の文法は16頁よりなる“報告”により完全に記述されたが, B 5000 のコンパイラを作る際に, もっとコンパクトな便覧が要求された. ALGOL 60 の下記のような性格を考慮して, 流れ図の形をとった(この流れ図を一面の折込みとして添付している).

1. ALGOL の構造はメタ言語の式によって記述される.
2. この式はメタ言語の変数と基本記号とよりなる.
3. メタ言語の各変数には幾通りかの定義があり, その多くは回帰的である.
4. メタ言語の変数のとる値は, 基本記号によって表わされる.

チャートは, 長円, 長方形, 円で囲まれたエレメントおよびこれらを結ぶ縦線, 横線よりなる.

メタ言語の変数は長円または長方形で囲む. 長円⬭は, 変数の定義がここで与えられることを示す(したがって, この下には縦線が出ている). 長方形▭は, 別の場所で定義された変数を示す. 基本記号は, 円○で囲む.

長円から出る1本または数本の縦線↓は変数の定義を示す. 縦線の先には, この変数を構成する唯一の(または最初の)エレメントがくる. 変数がいくつかのエレメントからなるときは, これらを必要な順序に横線→で結ぶ.

なお参照の便のために, 長方形の右端には, 定義の存在場所がチャート上の座標で指示されている. また長円の左端には, この変数が他の場所で言及された回数が指示されている.

このチャートは, コンパイラ作成時に大変役立ったが, その他, ALGOL プログラムのチェックおよび ALGOL 言語の学習にも有効であろう. (西村恕彦)

## 114. ALGOL を用いたベクトル心電図の診断

G.E. Forsythe, J.D. Groeben and J.G. Toole: Vector Cardiographic Diagnosis with the Aid of ALGOL [Com. ACM, Vol. 5 No. 2, Feb. 1962, pp. 118~122]

心電図はその発現機序が論理的に解明されやすく, 波形も比較的単純で利得も大きいのと, この検査法の普及が著しいので, 常に電子計算機の医学的応用の好対象となる.

心電図法とは体表の異なった2点の電位差, あるいは1点の電位の時間的変動を記録して, その調律の解析から心筋亢奮の神経伝導経路を検索し, あるいは波形の分析によって活動電流源である心筋の活動性を検索, 心筋障害を診断しようとするものであるが, 記録される各波形(通常, 12種類の誘導による)はいずれも三次元的な心電図ベクトルが一面に投影されたものであるに過ぎないから, これより心電図ベクトルを類推する診断に高度の論理性が必要とされ, その診断能にも限度がある.

これに対してベクトル心電図は, 心電図ベクトルをブラウン管上にリサージュ図形として表現するもので, 通常の心電図法に比べれば情報量が多く, したがって診断能にも高いものが期待されるが, 一面, 時間経過に関する情報に乏しい欠点がある.

本報告は, このベクトル心電図に時間情報を付加して診断能を高める試みに関するものであって, 電子計算機は直交座標系から球座標系への変換ならびに変換諸量のプロット, 患者の臨床データ記入などに用いられる.

使用電子計算機は Burroughs-220 (スタンフォード大学, 計算センタ), ブラウン管に display される直交座標系の心電図を5 m s の間隔で Sampling し, Benson-Lehner Model Oscar-K で量子化された心電図をパンチカードに記入, 時間に対して心電図の空間振幅 (S.M.M.V) と球座標系の球角($\alpha$), 傾斜($T$)などを計算, 変換してプロットする.

この他, ALGOL での計算プログラムについては, 座標系の変換ならびに変換された上記の諸量を時間に対してプロットする他, 患者に関する臨床データの記入などについて述べてあり, 正常心電図の分析結果および左室の肥大, 拡張があって心電圧が増大している例につき, 正常範囲からの deviation において診断する可能性を示している.

臨床的有意性については未だ研究途上にあるためであろうか詳しくは示されていないが, ベクトル心電図学の推進に電子計算機が有用である一面を紹介したものとして興味深い. (三浦 茂)