

1. 巡回差集合による直交配列表の 生成に関する問題について

池 野 信 一 (電気通信研究所)
喜 安 善 市 (電気通信研究所)
鈴 木 誠 道 (鉄 研)
酒 井 一 義 (日 本 I B M)
高 橋 秀 俊 (東大理 高橋研)
石 橋 善 弘 (")

解 説

池 野 信一 (電気通信研究所)

0. はしがき

増

去る 1959 年 7 月の会議の際、松山氏の提出された直交配列表生成の問題*のうち、問題 2 中 $\lambda = 4$ の場合について

鉄道技術研究所 (Bendix G15-D)

日本 I B M (IBM 704)

東大高橋研究室 (PC-1)

電気通信研究所 (M-1)

の 4 箇所それぞれ独立にプログラムを作製して解をさがしたところ、このような条件を満足するものは一つも存在しないという一致した結論をえた。

ここに、代表として通研で行なわれたものを中心にそのプログラムの概要について報告する。協力された方々に心から感謝申し上げる。

1. 増山の問題

" λ が正整数で $(3\lambda - 1)$ が素数のとき

* 増山：巡回差集合による直交配列表の生成

数理科学第 4 班第 3 分科会 第 2 回資料

(E1)

$$\sigma \in \{1, 2, 3, \dots, 3\lambda-2\} \pmod{3\lambda-1}$$

を互に素な三つの集合 σ , $\mathcal{L}_1$, $\mathcal{L}_2$ に分割し,

$$s(\sigma) = \lambda, \quad s(\mathcal{L}_1) = \lambda, \quad s(\mathcal{L}_2) = \lambda - 2 \quad (1)$$

とし,

$$\sigma \cdot (\varepsilon + \mathcal{L})^* = \mathcal{L}_1 \cdot \mathcal{L}_2^* - \lambda \varepsilon \quad (2)$$

とすることができるか?"

ここに

$$\sigma + \varepsilon = 1.$$

参考のためにここで定義を簡単に復習しておこう。

- i) $s(\sigma)$: σ の含む元の数
 - ii) $\varepsilon = \{0\}$
 - iii) 1 : 空集合
 - iv) $\sigma + \mathcal{L}_1 = \mathcal{L}_2$: σ のすべての元の $\mathcal{L}_1$ のすべての元から $\mathcal{L}_2$ ができていてそれ以外の元を L が含まないこと。
- $$s(\sigma) + s(\mathcal{L}_1) = s(L)$$
- v) σ^* : σ の共役集合
- $$\sigma^* = \{a_i^* \mid a_i + a_i^* \equiv 0 \pmod{3\lambda-1}, a_i \in \sigma\}$$
- vi) $\sigma \cdot \mathcal{L}_1 = p$: σ の任意の一つの元と $\mathcal{L}_1$ の任意の一つの元の和が p に含まれ, それ以外の元を p が含まないこと。

$$s(\sigma) s(\mathcal{L}_1) = s(\sigma \cdot \mathcal{L}_1)$$

従つて $\mathcal{L}_1$ が相異なる λ 個の元だけからなるならば

$$\mathcal{L}_1 \cdot \mathcal{L}_1^* - \lambda \varepsilon$$

は $\mathcal{L}_1$ の相異なる 2 元について, あらゆる差をとつてえられる元の集合である。

問題にもどつて, $\lambda = 2$ の場合は完全に解けているが, λ が 4 以上のものは未解決であるので, これを計算機によつて解けないかということである。その方法としては, (1) のような分割のあらゆるものを次々と作りながら, その一つ一つについて (2) の条件が成り立つかどうかをチェックするというやり方が提案され, 4 者ともこれに従つたわけであるが,

この方法の問題点はいかにしてこのようなあらゆる分割を手ぎわよく作るかということで、4者ともそれぞれ違ったやり方を持ちいていて興味深い。

2. 問題の意味

これも参考までに復習しておこう。

$\lambda = 2$ の例について考えると、

$$\alpha = \{1, 4\}, \quad \beta = \{2, 3\}, \quad \mathcal{L} = 1$$

が解となるが、 $\mathcal{L} + \varepsilon$ には 0, α には 1, β には 2 をあてて、元素の値の番号の順にならべると

01221

となる。これから作られる巡回行列に 0 だけの 1 行 1 列を追加して

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 & 2 & 1 \\ 0 & 1 & 0 & 1 & 2 & 2 \\ 0 & 2 & 1 & 0 & 1 & 2 \\ 0 & 2 & 2 & 1 & 0 & 1 \\ 0 & 1 & 2 & 2 & 1 & 0 \end{bmatrix}$$

という行列を作る。ここで

$$0 \rightarrow 1, \quad 1 \rightarrow \omega, \quad 2 \rightarrow \omega^2$$

$$(\omega^3 = 1)$$

とおきかえると複素数の直交行列となるのであつて、このような行列から直交配列表が作られる。問題の条件は 5 水準の直交配列表で巡回性のあるものが存在するための必要十分条件なのである。

3. 4 者のプログラム

以下、それぞれの報告をそのまま引用させて戴いた。説明の都合上通研のものを最初にあげたが、他は原稿の到着順である。

これらの報告のなかから、参考のため計算時間をぬき書きしてみると、

Bendix G15-D 65 分 PC-1 20 分

IBM 704 35 秒 M-1 35 分

となつて、高速計算機の偉力がよく感じられる次第である。

プ ロ グ ラ ム I

喜安 善市・池野 信一（電気通信研究所）

0. 我々は二つ

0. 我々は二つの方法によつて解をさがしてみた。

まず，増山氏に従つて，1～10の整数を4個，4個，2個という三つの集合 $\mathcal{A}$ ， $\mathcal{B}$ ， $\mathcal{C}$ に分割し，

$$\alpha \cdot (\mathcal{A} + \mathcal{C})^* \equiv \mathcal{B} \cdot \mathcal{B}^* - \lambda \varepsilon$$

が成立するか否かを，そのあらゆる分割についてチェックするというプログラムを作つたところ，この条件を満すものは一つもないという結果がでた。

次に，これらの $\mathcal{A}$ ， $\mathcal{B}$ ， $\mathcal{C}$ から巡回行列を作り，これに0ばかりの1行1列を追加した行列について，その各行が互に直交するか否かを直接テストするプログラムを作つた。この場合相隣る2行以上が直交するものはその直交行数がわかるようにして全部印刷することにより，プログラムの正しさをチェックし易いようにした。しかし結果は同じであつた。

1. 整数の集合の分割

まず簡単な2群への分割を考える。整数を1行にならべ，一つの群に属するものに0を，他の群のものに1をつければ，一つの分割は一つの2進数に対応させることができる。この2進数を小さいものから順にならべて

1 2 3 4 5 6 7 8

0 0 0 0 1 1 1 1

0 0 0 1 0 1 1 1

0 0 0 1 1 0 1 1

.....

1 1 1 1 0 0 0 0

のようにする。最小のものはすべて0が1の左側にきたものであり，

(E1)

最大のものはその逆である。いま、このなかの任意の相隣る2数 a , b ($a > b$) をとつてみると、たとえば

$a \dots\dots\dots 01\overset{*}{0}11100$

$b \dots\dots\dots 01100011$

のようになり、その左端から対応する桁の数を比較してゆくと、最初に不一致のおこつた桁では (* 印)、当然 a の方は0で b の方は1である筈である。それ以下に並ぶ数は、 a の方は、残つた1と0とを使つてできる最大の数であり、 b の方は最小の数でなければならない。このことから、次のようにして a から b を作ることができる。

a を右端からしらべてゆき、最初に0がならんでいたらその数を数える。次に1が来てから再び0の現れる最初の位置をみつける (* 印の点) この0を1に変え、その右に先刻数えた0の数より一つ多く0をならべ、そのさらに右側に空いた場所があつたら1をならべればよいのである。実際のプログラムでは都合により1のがわりに0を、0のかわりに0でない数をおいて表わした。なおこの2進数は1桁ごとに別の語に記憶させて操作を簡単にした。

以上の原理で8個を4個と4個に分割するプログラムと10個を8個と2個に分割するプログラムを作り、後者の8個を前者の分割でわけることにより、4個4個2個の分割を作つた。なお、会議席上の提案に従つて、01の最大数が02の最大数より小さい分割だけを作つて、数を半減した。

2. 条件のテスト

集合間の掛け算は次のような数の掛け算で表わすことができる。たとえば

$$\{0_2, 2, 5\} \cdot \{2, 3\} = \{2_2, 3_2, 4, 5, 7, 8\}$$

は

$$(2x^0 + x^2 + x^5)(x^2 + x^3) = 2x^2 + 2x^3 + x^4 + x^5 + x^7 + x^8$$

と表わされる。ここで各項の係数の最大のものより整数 x を大きく選んでおけば、桁上りによつて係数の変る心配はない。集合を mod m で考

えるときには x^m 以上の項を x^m で割ればよい。これは x 進法 m 桁シフトで簡単にできる。

各集合 $\alpha, \beta, (\varepsilon + \mathcal{L})$ の元の数はいずれも 4 を越えず、しかもすべての元は相異っているから、条件式のなかの積にあらわれる同一の元は高々 4 個である。従つて x は 5 以上であればよいわけであるが、2 進法計算機に便利のように $x = 8$ とした。なお、この条件式で左右の元の数は一一致しているから、 $\eta + \varepsilon = 1$ の条件は考える必要がない。つまり、上記のような数で表現して左右両辺を計数し、これをそのまま引き算して 0 になるかどうかをしらべれば十分である。

3. プログラムとその結果

以上のような考えに従つてプログラムを作り $M-1$ 計算機 (加算 5 m sec., 乗算 22 m sec.) にかけた。付表はその主要部のフローチャートである。実際にはこのほかにプログラムをチェックする目的で、任意の時間に一時停止させ、そのときの分割とその番号、掛算結果などを印刷するルーチンを含ませた。

特に技巧を用いず簡単に書き下した程度のプログラムであるが、全体で約 120 語、計算時間は約 35 分程度であつた。

4. 直交性を直接テストするプログラム

分割の発生には前と同じプログラムを使い、行列の 1 行を作る。これをつつと巡回させて次々と行を求めながら元の行との直交性をテストし、直交しない行がでるまで続ける。2 行以上が直交する場合にはその行数と分割の番号を印刷する。

直交性のテストをするには、元素を 0, 1, 2 で表わした形で 2 行をとり、対応元素間で mod 3 の引き算を行ない、1 が 4 個、2 が 4 個 (したがつて 0 も 4 個になる) であるかどうかをしらべればよい。

プログラムの語数は前のものと同程度、計算時間は丁度 2 倍であつた。結果は

相隣る 2 行だけ		237 個
” 3 行		48 個

相隣る4行		18個
" 5行		6個
" 6行以上		9個

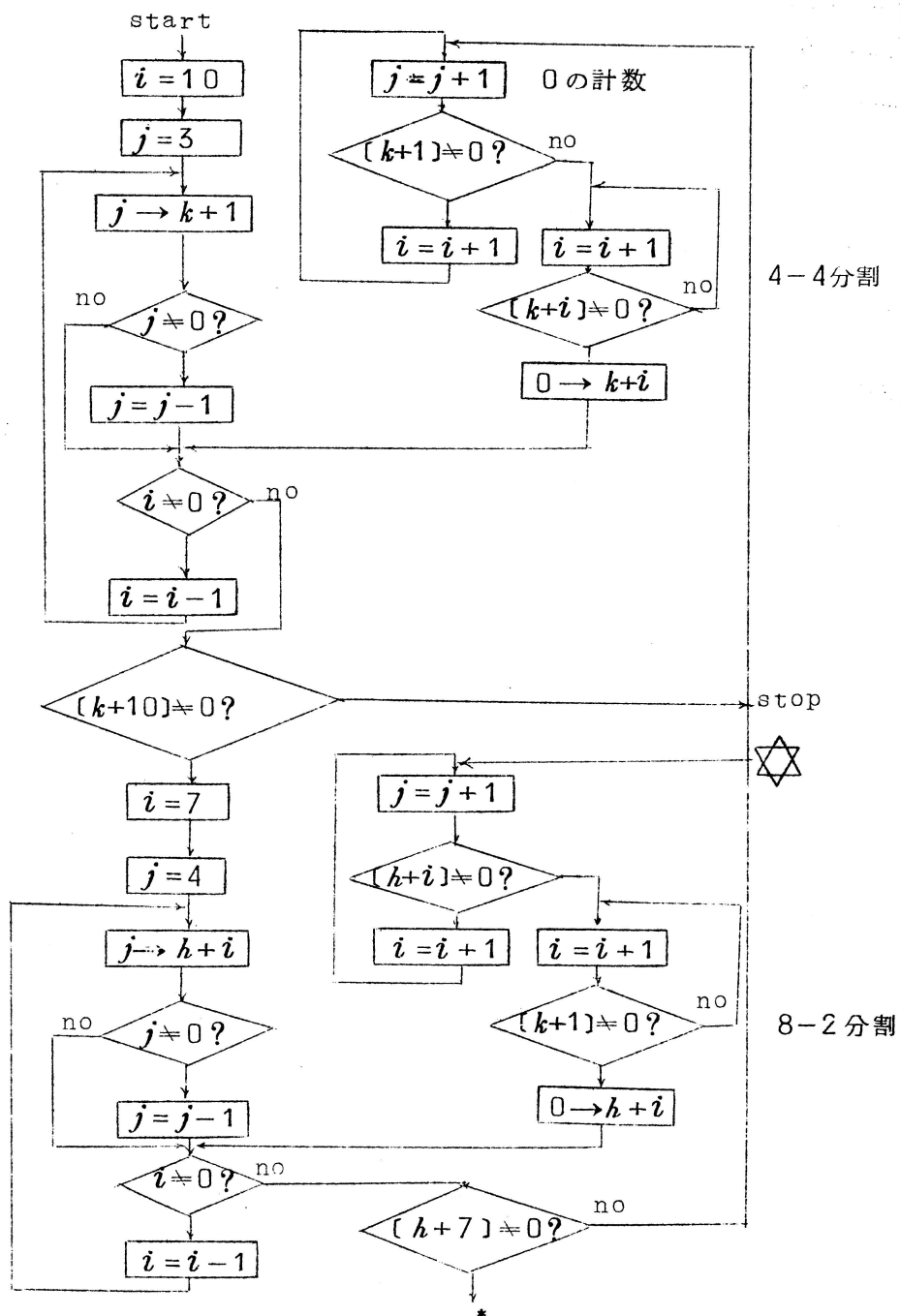
となつた。

5. $\lambda = 6$ の場合について

$\lambda = 4$ の上は $\lambda = 6$ であつて、この場合は16個の整数を6, 6, 4個に分割することになるわけであるが、その分割の数は

$$\frac{16!}{6!6!4!} = 1681680$$

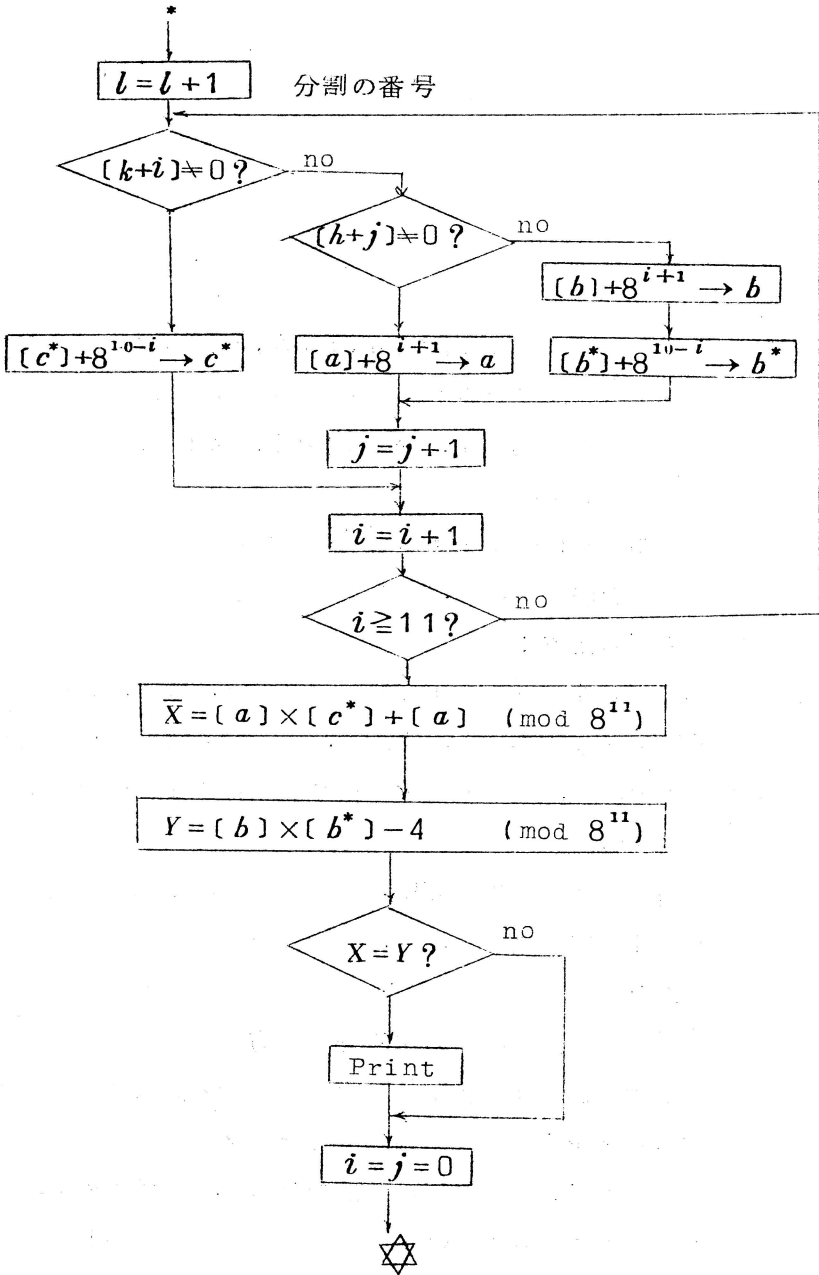
となる。実際にはこの半分でよいが、 $\lambda = 4$ の場合の時間から推定すると300時間余り必要と思われる。何らかの工夫をして短縮しない限り実行はむづかしい。



(注) $[k+i]$: $k+i$ 番地の内容

$j \rightarrow k+1$: 数値 j を $k+1$ 番地に入れる

(E1)



プ ロ グ ラ ム Ⅱ

鈴木 誠 道 (鉄 研)

1. 問 題

$$g^d = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$$

を互に素なる3つの集合 α, β, γ に分割し、

$$s(\alpha) = 4, \quad s(\beta) = 4, \quad s(\gamma) = 2$$

として、

$$(1) \quad \alpha(\varepsilon + \gamma)^* = \beta \cdot \beta^* - 4\varepsilon$$

を満たす α, β, γ を求める事。

2. 考え方：

1 から 10 までの 10 ケの数に 4 ケ、4 ケ、2 ケに分ける場合の数は、3150 通りであるが、(1) は α, β を交換しても成り立つから、3150 通りの内、 α, β の交換によつて得られる 1575 通りについては、調べなくてよい。

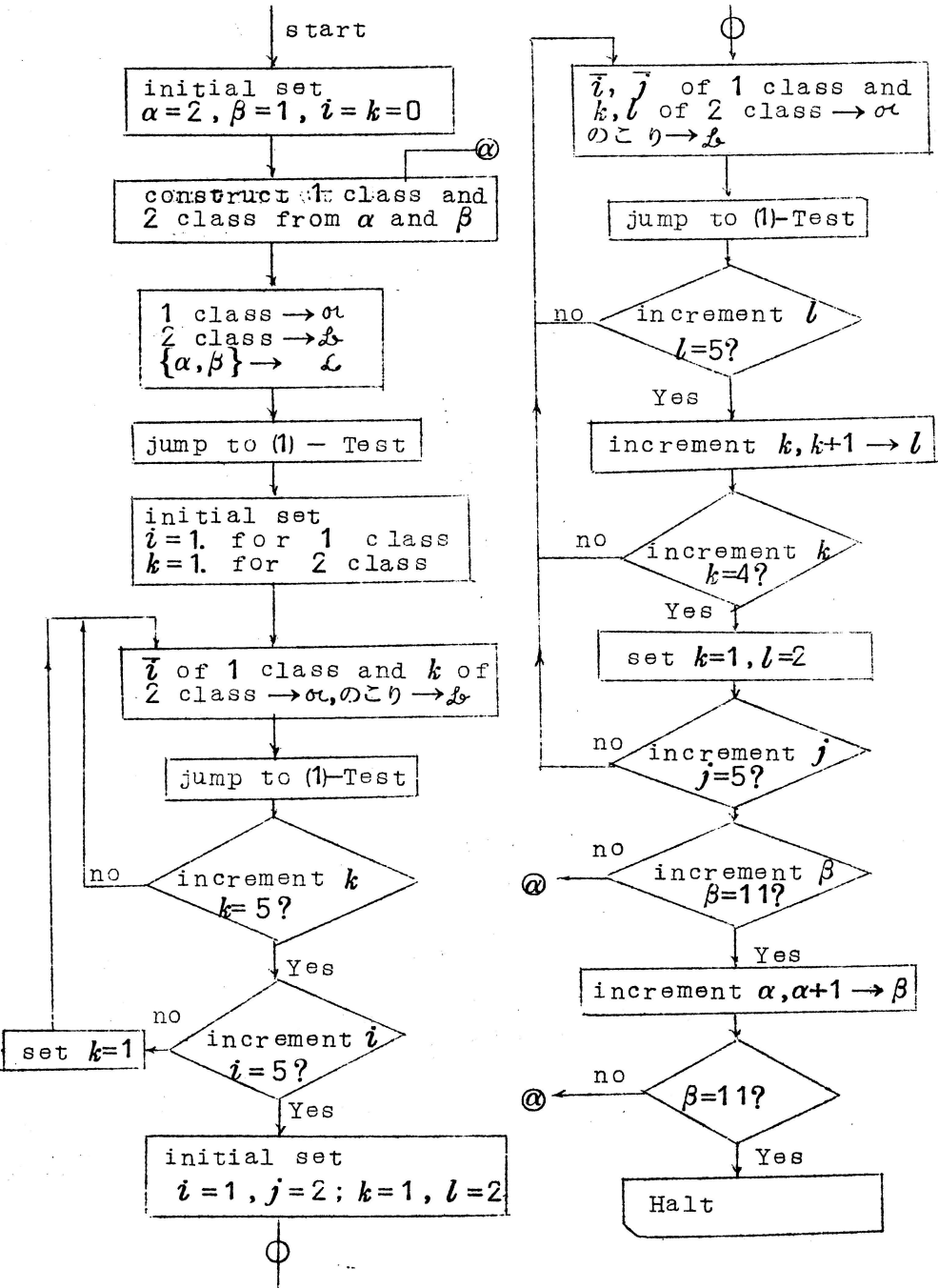
結局後に述べる 1575 通りについて、(1) が成り立つかどうかを調べて、もし(1) が成り立てば、 α, β, γ を typeout するようにプログラムを組んだ。

1) α, β, γ の組み分けの方法

まず g から α, β を選び之を ε とする。次に $g - \varepsilon$ の 8 ケの内小さい方の 4 ケを 1 class 残りの 4 ケを 2 class とし、各 class の element に 1, 2, 3, 4 と番号をふり、(プログラム上で云えば、4 ケの element が次々の location に store されている事に対応する。) この 1 class, 2 class から 4 ケを選んで α とし残りを β とする。

1 class, 2 class からの 選び方は、 α と β の交換で得られるものを除き、36 通りある。之が終つたら、 α, β を変更して新しい、1 class, 2 class を構成して、上の操作をくりかえし、 α, β の選び

方のすべてをつくすまで、之を行う。(1)が成り立つかどうかは、 α , β , γ が構成される都度行う。以上をFlow Diagram に表わせば、



組み分け，の仕方を見やすくするために，始めの数例を表にする。

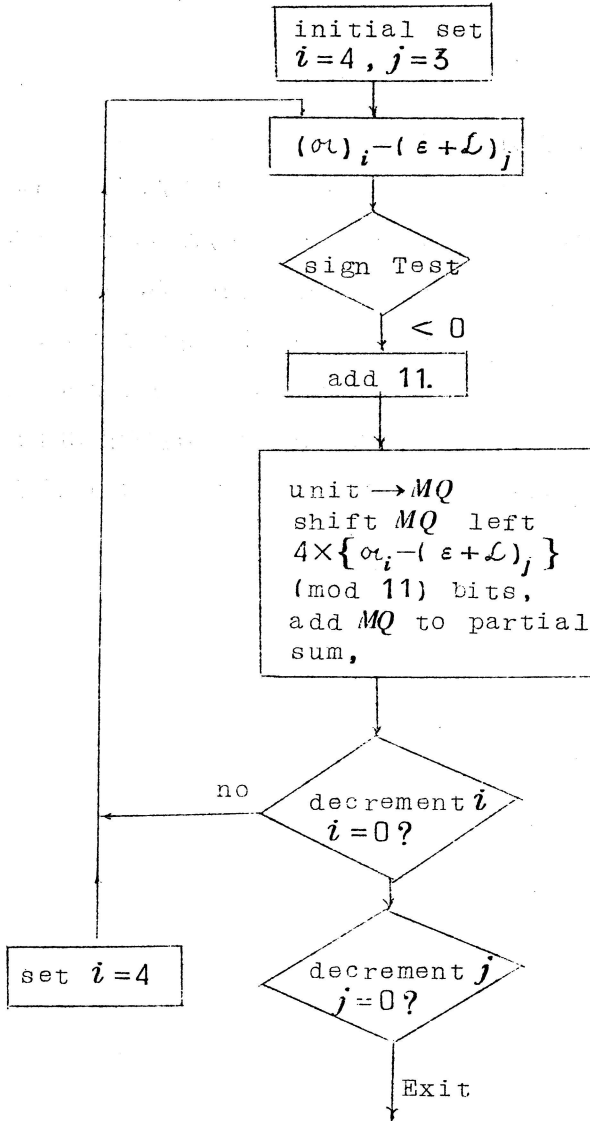
class no	L		1 class				2 class			
	1	2	3	4	5	6	7	8	9	10
1			○	○	○	○				
2				○	○	○	○			
3				○	○	○		○		
⋮										
6			○		○	○	○			
7			○		○	○		○		
⋮			⋮							
17					○	○	○	○		
18					○	○	○		○	
⋮			⋮							
36			○	○					○	○
⋮										

○は， α に属する。

ii) $\alpha\{\varepsilon + L\}^* = L L^* - 4\varepsilon$ が成り立つかどうかのテスト

この部分は subroutine をなしている。さらに之は，3つの部分に分けられている。第1は， $\alpha\{\varepsilon + L\}^*$ を計算する部分，第2は， $L L^* - 4\varepsilon$ を計算する部分，第3は，それが相等しいかどうかを判断する部分である。 $\alpha\{\varepsilon + L\}^*$ や $L L^* - 4\varepsilon$ の値は，どの要素がいくつあるかを，1つの element に対して4 bits を割り当て，計40 bits で表わし，2 word に store させた。

$\alpha\{\varepsilon + \mathcal{L}\}^*$ を求める Flow Diagram



$\mathcal{L}\mathcal{L}^* - 4\varepsilon$ を求めるには、之が任意のあらゆる二元について、あらゆる差をとつて得られる元の集合である事を用いて、enumerative に $\mathcal{L}_i - \mathcal{L}_j \pmod{11}$ を求め、 $\alpha\{\varepsilon + \mathcal{L}\}^*$ を求めるのとはほぼ同様な方 (E1)

法を採った。

$\alpha\{\varepsilon + \mathcal{L}\}^* = \mathcal{L}^* - 4\varepsilon$ が成り立つかどうかのテストは、各々 2 words に store されている左右両辺の値の差を求め 0 test を行うだけである。

3. 結 果

2 の述べた方法によつて Bendix G15-D の machine language でプログラムを組んだ結果、命令の数 197, 定数の数 26 から成るプログラムを完成した。(7・25・'59) このプログラムでは、1 回テストする毎に Bell がなり、(1) が満たされたら、その結果が typeout されるようにしてある。計算の結果、(1) を満たすような組み合わせがないという事がわかつた。なお所要時間は、約 65 分であつた。

又 $\lambda = 6$ に対する問題に以上と同じ方法を採用すれば、数 100 時間を要する事になるので、もつと elegant な方法を採る必要があると思う。

(9・8・'59)

プ ロ グ ラ ム Ⅲ

酒 井 一 義 (日本 I B M)

I 計算の方法

$$\alpha(\epsilon +)^* \equiv \epsilon \cdot \epsilon^* - \lambda \epsilon \quad \lambda = 4 \quad \dots\dots (1)$$

を変形して

$$\alpha\alpha^* + \alpha\epsilon + \epsilon^* (\alpha + \epsilon) + \epsilon + \epsilon^* \equiv 3 + 3\epsilon \quad \dots\dots (2)$$

を満足する $\alpha, \epsilon, \epsilon^*$ の組が存在するか否かを調べる。

$$g = \{d_1, d_2, \dots, d_i\} \rightarrow 2^{d_1-1} + 2^{d_2-1} + \dots + 2^{d_i-1}$$

として表現する。

1. 凡ての $\alpha, \alpha^*, \epsilon, \epsilon^*$ を上記の表現法によつて生成し記憶する。
2. 一つの α について $\alpha\alpha^* + \alpha$ を計算し、各元の個数を記憶する。
3. 一つの ϵ の元と α の元が重複していれば次の L に移る。
4. $\epsilon^* (\alpha + \epsilon) + \epsilon + \epsilon^*$ を計算し、各元の個数を求める。
5. 一組の α, ϵ が②式を満足しないならば次の L に移る。
6. α が既に②式を満足した組の L に一致するならば次の ϵ に移る。
7. ②式を満足した場合は $\alpha, \epsilon, \epsilon^*$ の順序でその元を印刷する。
8. 一つの α について凡ての ϵ の test が終つたならば次の L に移る。
9. 凡ての α について凡ての ϵ の test が終つたならば計算を止める。

II Program

上記のような計算方法による Program (別紙 Flow chart 参照) を SAP (IBM 704 Symbolic Assembly Program) で assembly した。その内訳は

命令の数	Main Program	243
	Output Subroutine Program	404
Data field	解の個数	+572

III 計算結果

上記の Program を IBM 704 型計算機で test run したところ Card read の時間を含め 35 秒後に計算を終ったが何も印刷されなかった。

そこで Change Card を挿入して、 $\alpha \cdot \alpha^* + \alpha + L^* (\alpha + L) + L + L^*$ の各元の個数を 3150 組求め、Dump out して調べたところ (2) を満足する組は一組も存在しなかった。又一部手計算した結果と比較してみたが完全に一致し Program の誤りは発見できなかった。

故に (1) を満足する α , L , L^* の組は存在しないものと思われる。

IV $\lambda \geq 6$ の場合

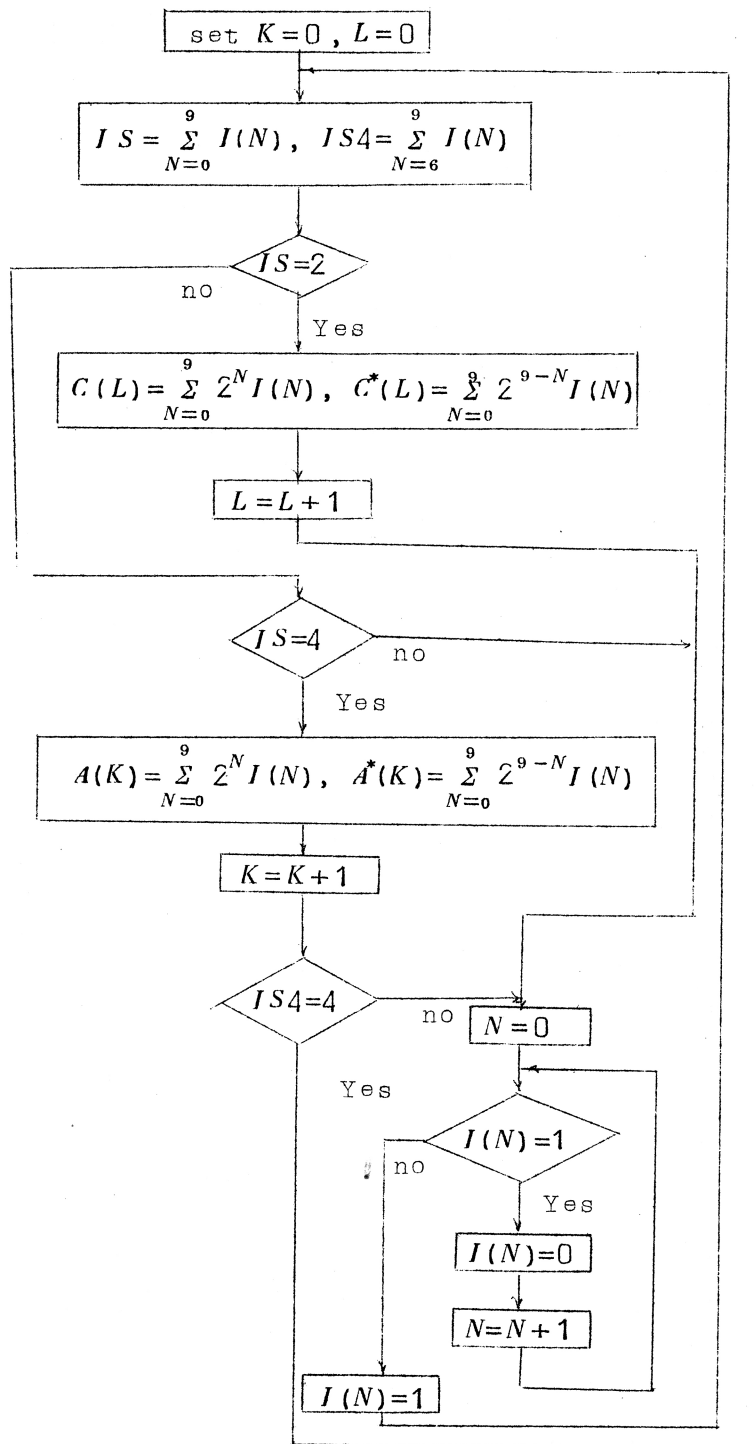
$\lambda = 6$ について考察すると

$$(3\lambda - 2)! / \lambda! \cdot \lambda! \cdot (\lambda - 2)! = 1,681,680$$

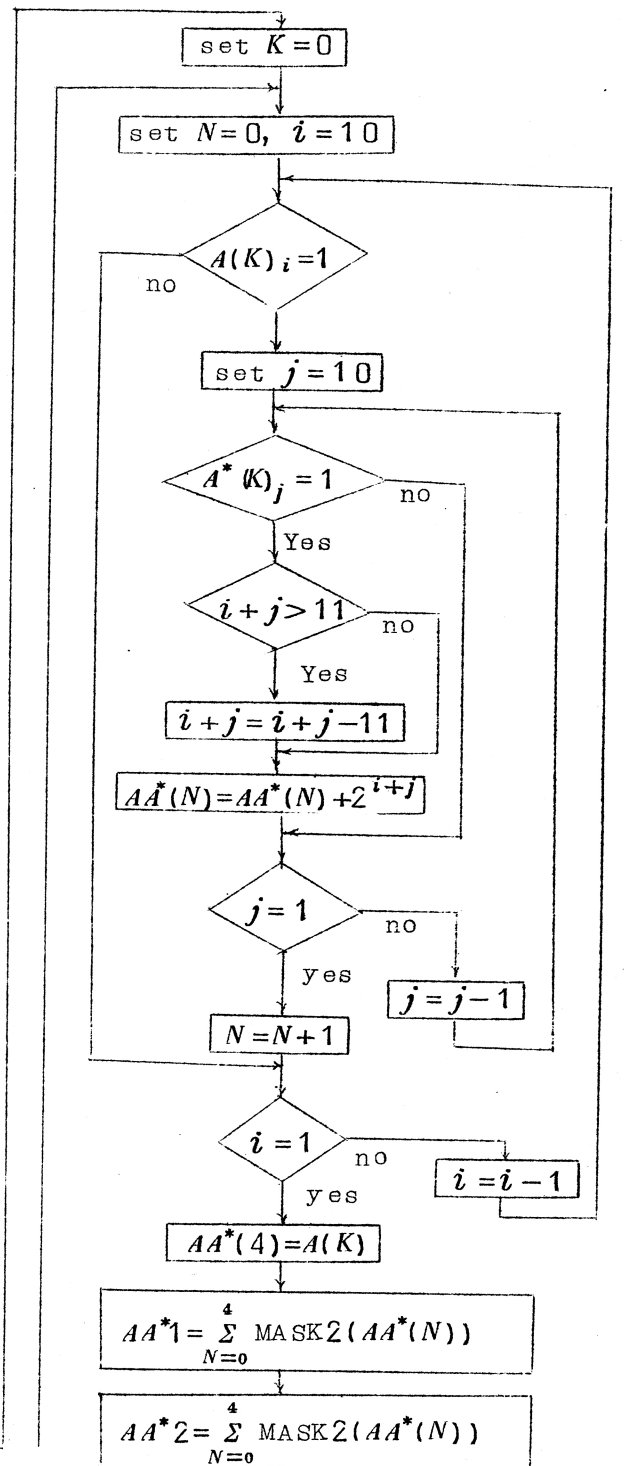
となり IBM 704 型計算機 (演算速度; 加減算 $24 \mu s$) で計算して約 2 時間要するものと思われる。

Program は記憶容量, 計算時間の短縮等のため一部変更しなければならないが上記の如き方法で可能である。

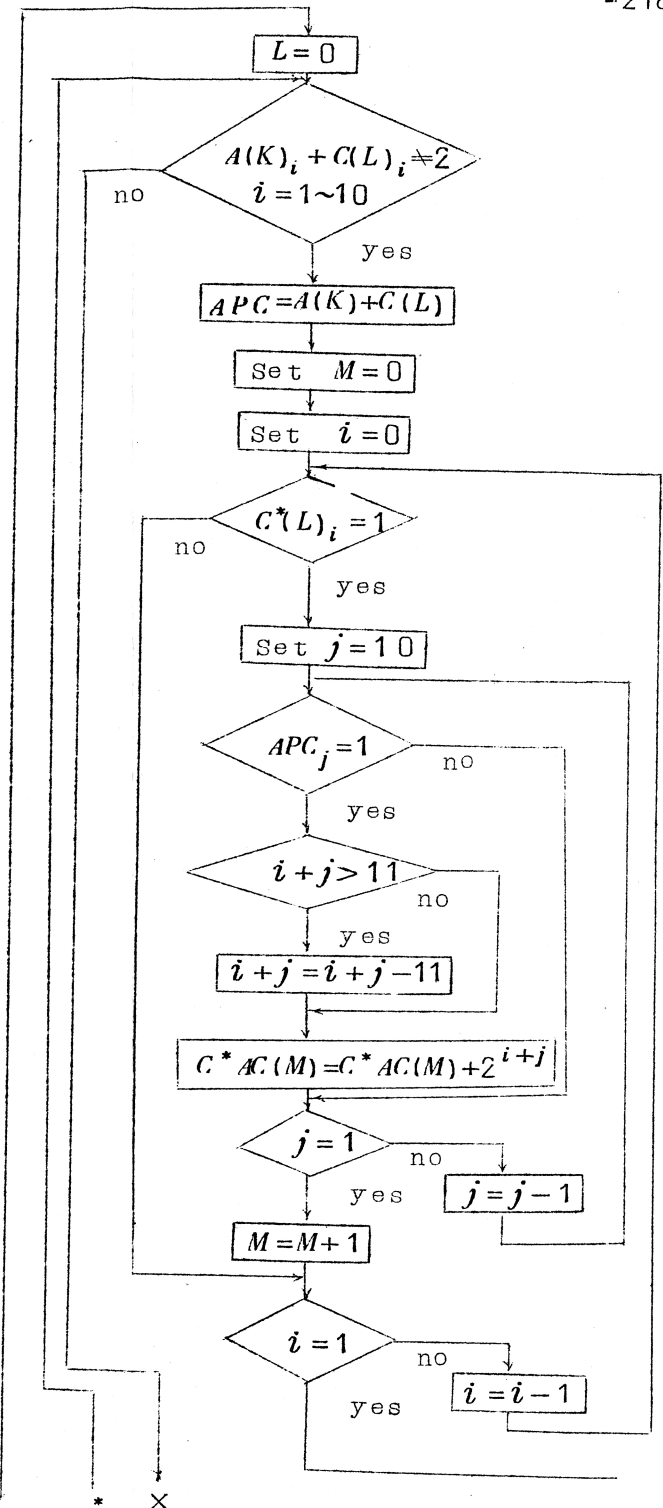
$\lambda \geq 8$ については記憶容量, 計算時間等のため特別の方法を考えない限り不可能と思われる。



※

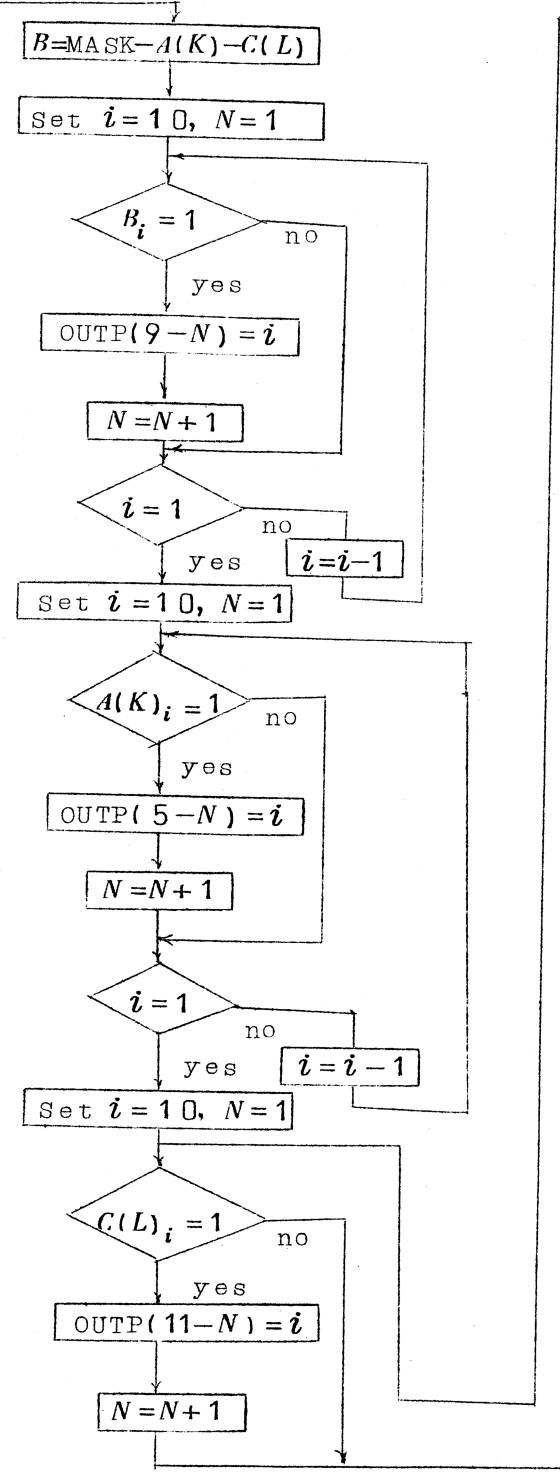
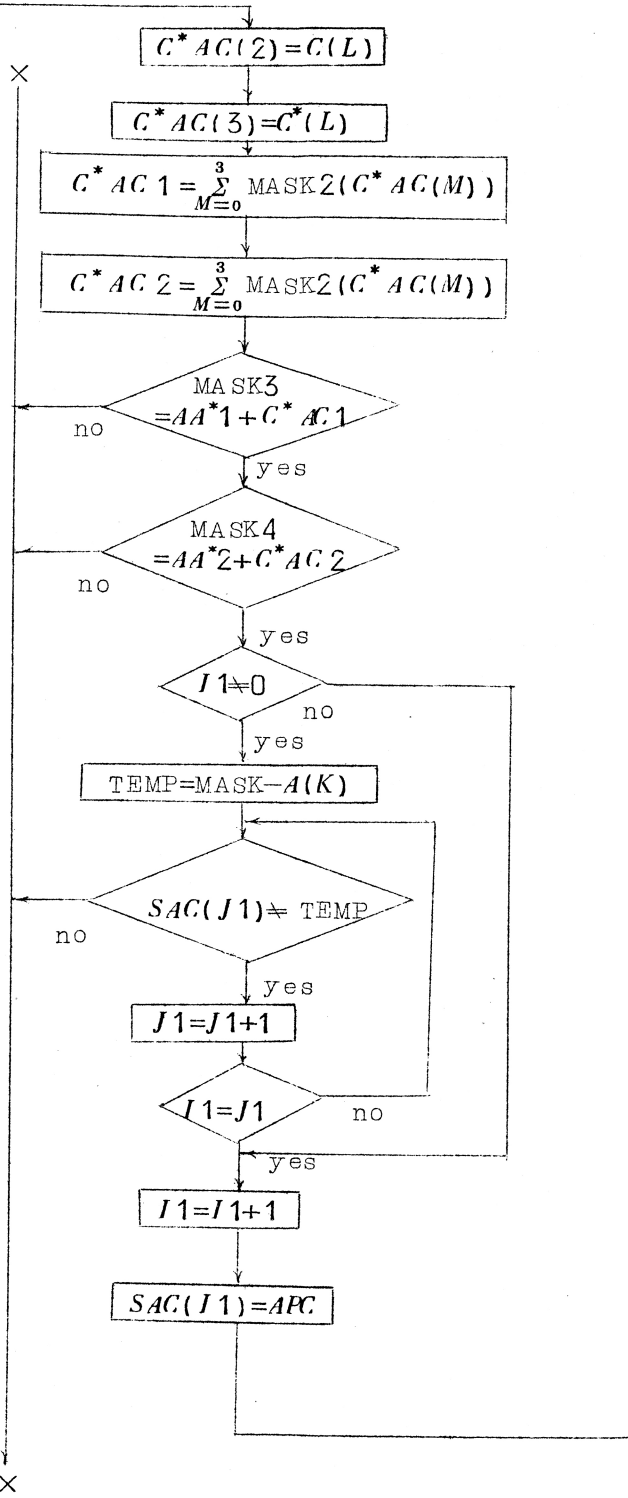


(E1)

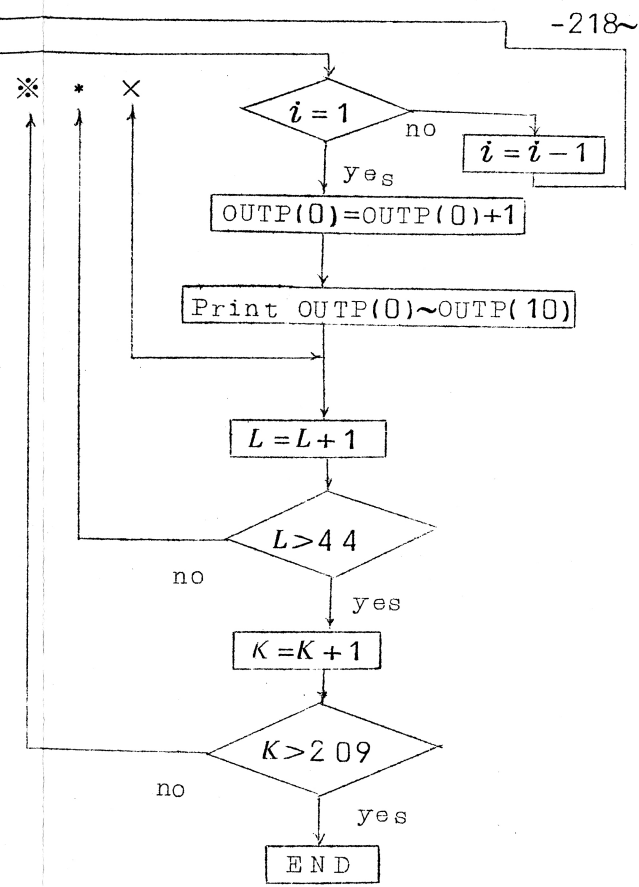


*

X



(E1)



MASK =000001111111111
 MASK1=000010101010101
 MASK2=000001010101010
 MASK3=001101111111111
 MASK4=000011111111110

プ ロ グ ラ ム IV

高 橋 秀 俊・石 橋 善 弘 (東大理)

問題 1, 2, ………, 10, を元の数が4個, 4個, 2個であるような α , β , γ の各グループに分け

$$\alpha(\gamma + \varepsilon)^* = \beta\beta^* - 4\varepsilon$$

とすることが出来るか?

解 答 そのような組合せは存在しない。

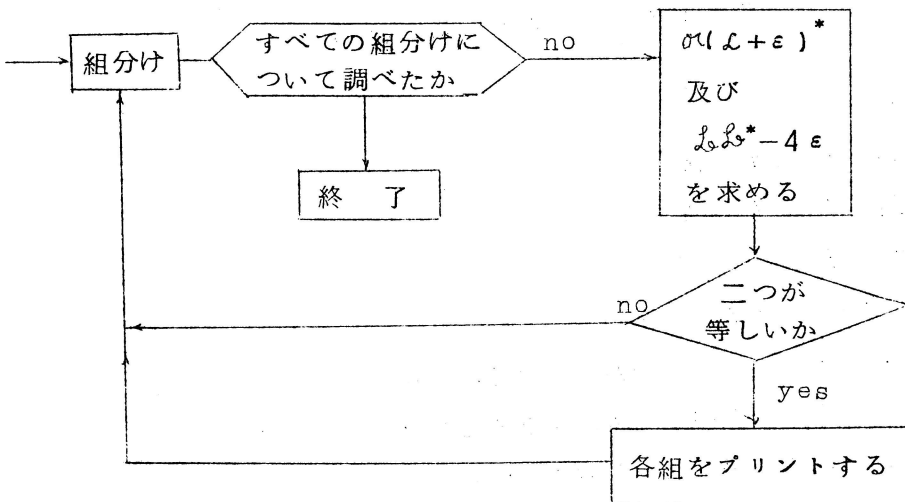
計算方法

計算に用いた計算機 PC-1

所要時間 約20分

プログラムの概略

1.



2. $\alpha(\gamma + \varepsilon)^*$ と $\beta\beta^* - 4\varepsilon$ が等しいかどうかの判定

$\alpha(\gamma + \varepsilon)^*$ の element に 0, 1, ………, 10 がそれぞれ何個含まれているかを覚える register に $\alpha(\gamma + \varepsilon)^*$ の element が求まる

(E1)

毎に相当する場所に 1 を加えて行く。

逆に $\mu^* - 4\varepsilon$ の element を求めてこの場合は register の内容から 1 ずつへらしてゆく。

その結果 register がすべて 0 になつていれば二つのものは等しいと判断される。

3. 組分けの方法

組合せの数は $10! / 2! 4! 4! = 3150$ 個あるが、 $\alpha = (a_1 a_2 a_3 a_4)$, $\mu = (b_1 b_2 b_3 b_4)$, $\lambda = (c_1 c_2)$ において $a_1 < b_1$ としてもかまわないので実際には 1575 個のものについて調べるとよい。

この組合せを作る際に g 番地にある内容が $8g$ 番地からプログラムパラメーターで指定された番地までにある内容のどれかと一致するかしないかを調べるような Comparison subroutine を使った。

4. 組合わせの作り方を示すフローチャート

$\lambda = (c_1 c_2)$ を $8g, 9g$
 $\alpha = (a_1 a_2 a_3 a_4)$ を $10g, 11g, 12g, 13g$
 $\mu = (b_1 b_2 b_3 b_4)$ を $14g, 15g, 16g, 17g$

につくことにする。(フローチャートは次頁へ)

但し

$$c_1 < c_2$$

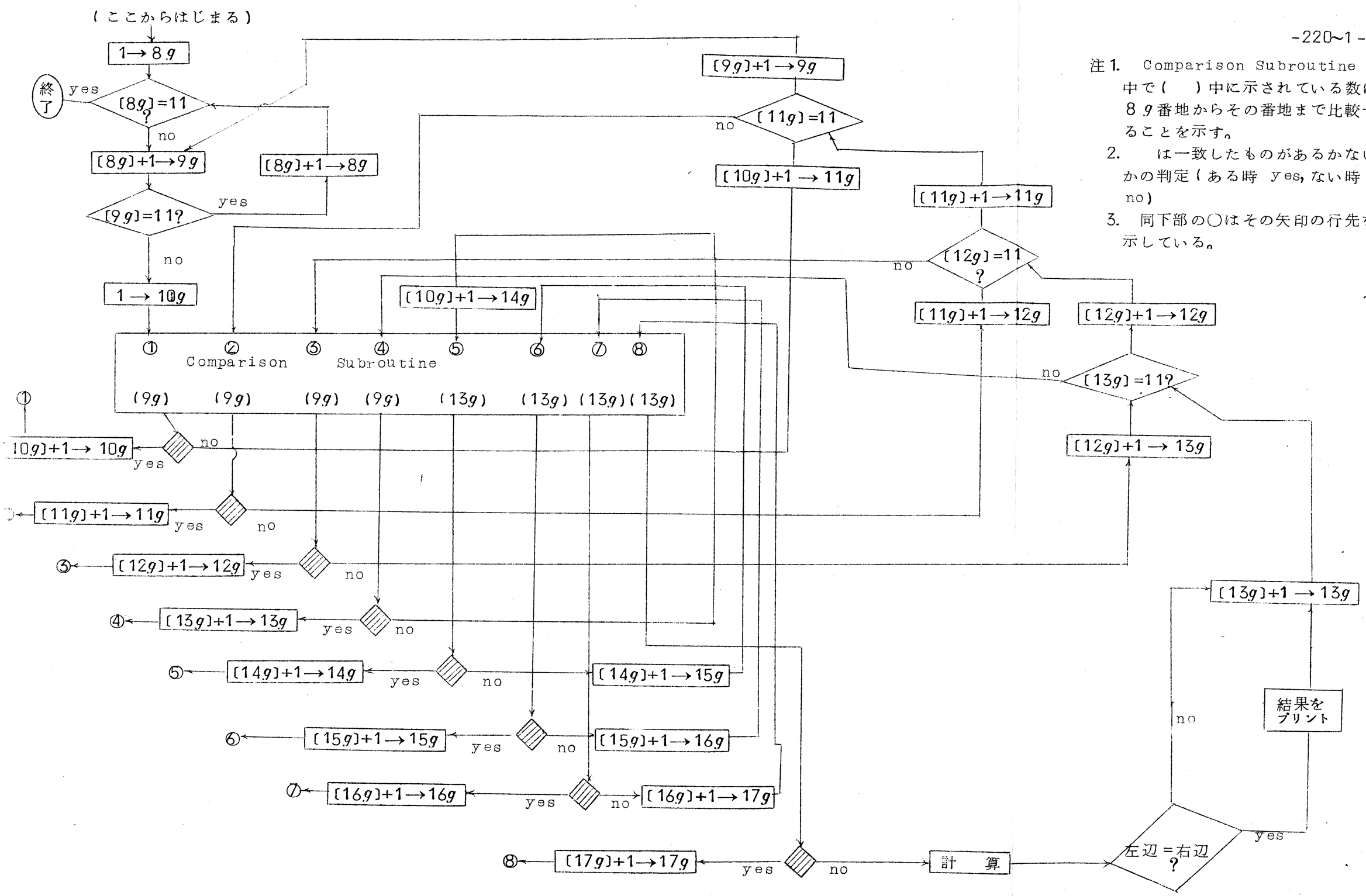
$$a_1 < a_2 < a_3 < a_4$$

$$b_1 < b_2 < b_3 < b_4$$

$$a_1 < b_2$$

フローチャートでわかる通りこの方法が最良とは思えない。というのは μ は α , λ でとつたものの残りでよく、それは自動的にきまるわけでここでやつたような面倒な方法をとる必要はない。従つてそれを改良すると時間は更に早くなると考えられる。

- 注1. Comparison Subroutine の中で () 中に示されている数は 8g 番地からその番地まで比較することを示す。
2. は一致したものがあるかないかの判定 (ある時 yes, ない時 no)
3. 同下部の○はその矢印の行先を示している。



(E1)

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、権利者の検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なからずありました。

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、一定期間の権利者搜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>