4　エラステネスのふるいによる素数の計算について

新　井　克　彦（通　研）

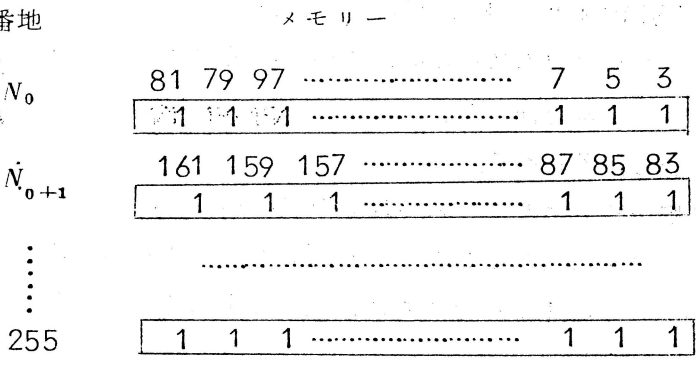
多くの電子計算機で素数を計算するプログラムは作られているが，その多くは繰返し除算を行なうことによつて計算を行つているため計算時間が非常に長いのが常である。計算時間を短縮するためにエラトステネスのふるいによつてプログラムを組みよい結果が得られたので，これについて報告する。

Ⅰ．整数のリストの記憶と方法

2進法の計算機のメモリーの各桁に仮想的に番号をつけておき，この番号によつて整数を代表する（下図）。

始めに各桁を皆1にしておき，順次合成数に対応する番号のついた桁に0を書き込む。このふるい演算の終了後0でない桁をピックアップして，対応する番号を素数として印刷する。

番号のつけ方（1語40ビットの場合）



3以上の素数は皆奇数であることから，上図のように奇数番号をつけ

ておく。合成数に対応する桁に0を書き込むためと0でない桁をピックアップするためにある特定の桁が1で他の桁は皆0であるような語を用意し，これをパイロットということにする。こうすればある番地のある桁に0を書き込むにはこれに対応するパイロットと番地との論理積を作り更にこれと番地との2を法とする和をつくればよく，0か1かを知るには，論理積をつくり結果が0か0でないかを判定すればよい。

今nがpの倍数であることが分かつたとすればnより大きいpの倍数は，nに対応する桁をp桁ずつ左へずらすことによつて知れる。プログラミングに際してはシフトした際オーバーフローすることを考え，$p/40$の商だけ番地を進め剰余でシフト桁数を指定すればよい。

次に，素数pの倍数をふるい落すとしたときp^2以下のp倍数は，すでにp以下の素数の倍数としてふるい落されている筈であるから，ふるいの操作は，p^2から始めればよい。p^2の位置は$(p^2-3)/80$の商をp_1，剰余をp_2とすればN_0+p_1番地の左から$p_2/2+1$桁目であることが分るから，ここから始めればよい。

Ⅱ．13,601以下の素数計算

以上の考察に従つてプログラムを組んだ（別図フローチヤート参照）。

プログラム：85語（一時記憶，印刷ルーチンを含む）

計算時間：3～13,597迄1,608ケの素数を約1時間10分

Ⅲ．任意のn_0から$15040+n_0$までの範囲の素数計算

上で得た素数をデータとして読み込むことによつて，任意のn_0以上の素数を計算させるプログラムを作つた。

Ⅰの場合と異なる点を2，3述べる。

A) Ⅰでは始めに各桁を皆1にしてふるい演算を行なつたが，ここでは始めに各桁を皆0にすることにした。これは0を書き込むには2回の論理演算が必要だが，1を書き込むには論理和1回で済むからである。

B) 素数pを読み込んで，pの倍数をふるい落す際，$p^2-n_0>0$であればⅠと同様に$(p^2-n_0)/80$の商と剰余とでp^2の位置が指

定できる。$p^2-n_0<0$ の場合 n_0 以上の最小の p の倍数を得るには，次のようにすればよい。

n_0/p の剰余を r とすれば，r が偶数のときは n_0+2p-r が，r が奇数のときは，n_0+p-r が，n_0 以上の最小の p の奇数倍である。従つて n_0 の桁からそれぞれ $(2p-r)/2$，$(p-r)/2$ 桁目からふるい演算を始めればよい。この場合も各々を40で割つた商および剰余で番地と位置が指定できる。

C) IIのプログラムと異なり，このプログラムではふるい演算が完了してから始めて素数か否かの判定を行なうので，両者のプログラムを分離して語数の節約をはかつた。

D) このプログラムでは，3から $\sqrt{n_0+15{,}040}$ までの素数をデータとして読み込む必要がある。

以上の考察のもとにプログラムを組んだ。その内訳は

プログラムの長さ：ふるい演算　67語
印刷　48語
いずれも定数，一時記憶を含む。後者はふるい演算完了後読み込まれる。

計算時間：12,301~27,337 迄1,521ヶの素数計算に約1時間10分，このうちふるい演算に要する時間は約11分
10,004,201~10,019,241 迄935ヶの素数，約1時間，ふるい演算は約12～3分。

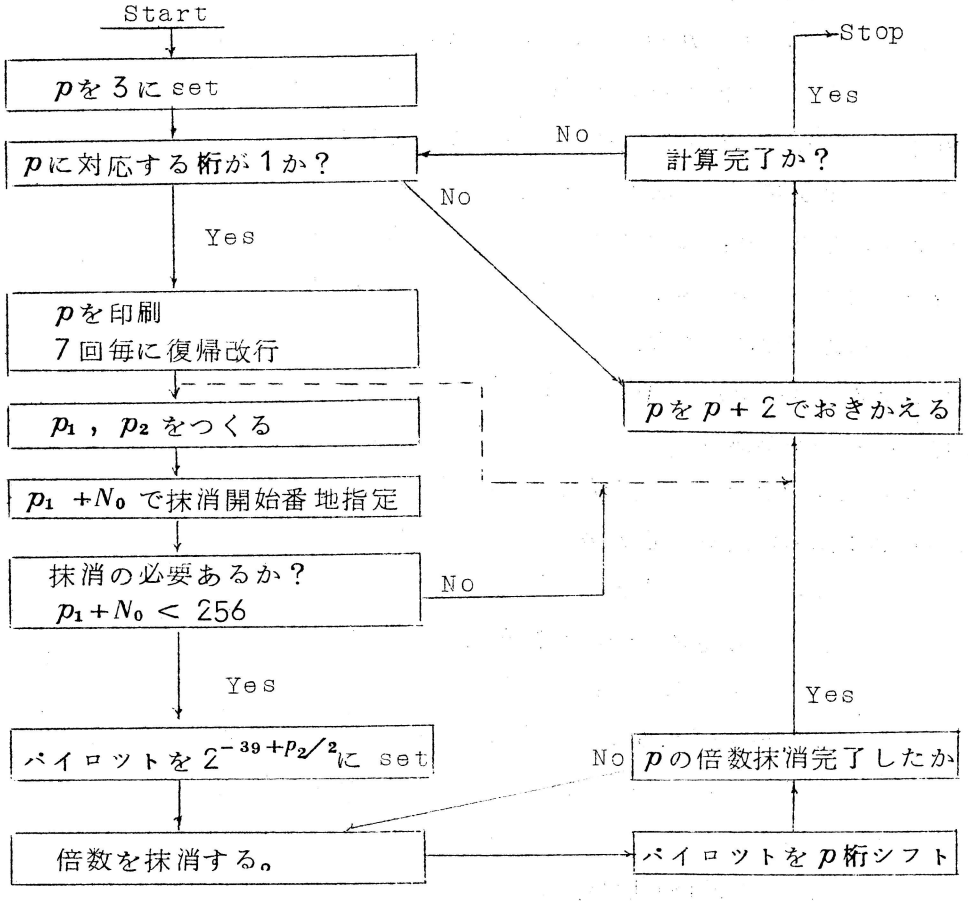
上限：原理的に計算できる上限は $2^{38}-1$ であるが，現在実際に可能な範囲は大略 2.8×10^{11} 程度である。

IV. 割算による方法との比較

通研において，かつて割算による素数発生のプログラムが組まれているので，この結果と比較すると，

5～6桁程度の素数の計算については1/3~1/4程度
7～8桁　〃　〃　1/10かそれ以下
} の時間で計算できる。

A. 13,601 以下の素数計算フローチャート



Notations :

p : 考えている整数

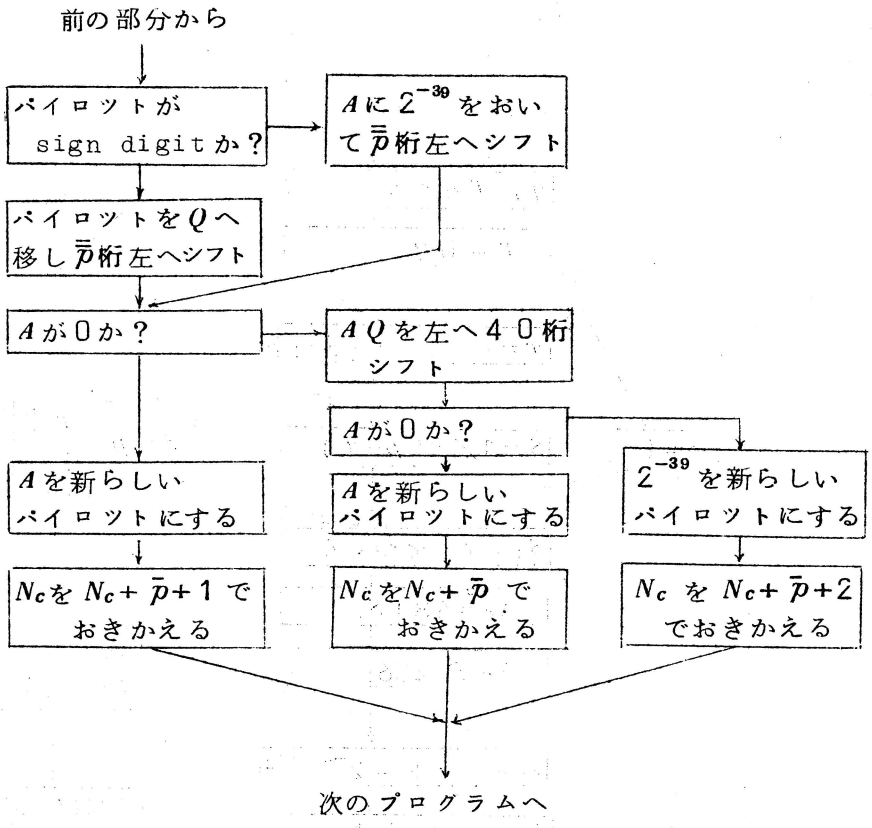
p_1 : $(p^2-3)/80$ の商

p_2 : 〃 〃 の剰余

N_0 : 整数のリストの入つている最初の番地。($M-1$の場合86)

* この部分は次に詳しいフローチャートを示す。

B. 前図* の部分の詳細なフローチャート



Notations：

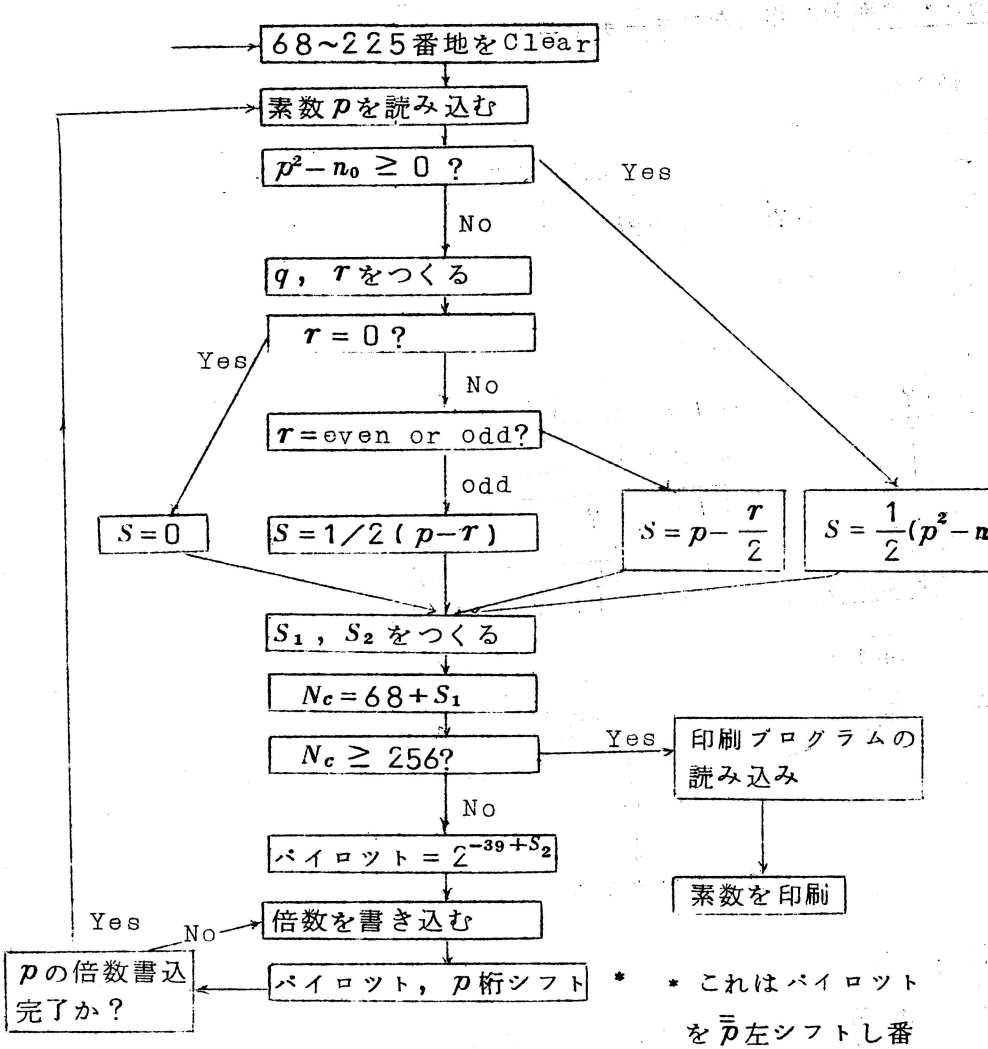
$\bar{p}$, $\bar{\bar{p}}$：$p=40\bar{p}+1+\bar{\bar{p}}$　但し，$2\leq\bar{\bar{p}}\leq 40$

A, Q：A = accumulater,　Q = quotient register

N_c　：抹消すべき番地

パイロット：今抹消しようとする桁が1で，他の39桁は皆0であるような単語を便宜上こうよぶ。

C. 任意の n_0 から $n_0+15,040$ までの素数計算フローチャート



Notations :

N_c, $\bar{p}$, $\bar{\bar{p}}$ は先のフローチャートと同じ。

q, r : n_0/p の商を q, 剰余を r とする。

S :

$$S = \begin{cases} 1/2(p^2-n_0) & (p^2 \geq n_0) \\ 1/2(p-r) & (r=\text{odd},\ p^2 < n_0) \\ 1/2(2p-r) & (r=\text{even},\ p^2 < n_0) \end{cases}$$

S_1, S_2 : $S/40$ の商を S_1, 剰余を S_2 とする。

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