TM-0475

整式処理システムNOR
－YOKOSYMAの現況

野呂正行、竹島　卓、横山和弘

March, 1988

ICOT

Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan

(03) 456-3191～5
Telex ICOT J32964

Institute for New Generation Computer Technology

整式処理システムNOR-YOKOSYMAの現況

富士通(株)国際研

野呂 正行 竹島 卓 横山 和弘

1. はじめに

我々のグループでは、ICOTからの委託研究として約2年前から数式処理システムの研究開発を行っている。以下で、現在までの進行状況および今後の目標などについて述べる。

2. システムの概要

我々の当面の目標は、不定積分アルゴリズム(Rischアルゴリズム)のインプリメントである。このことは、何も不定積分自体が特に必要とされているという意味ではなく、単に到達目標として手頃なものであるという理由による。(これは既にあちこちで述べて、その都度思想がない、などとたたかれている通りである。)もちろん、単に計算機能を豊富に持つだけではシステムとはいえず、それらを使うための言語、インタフェースおよび環境などが整って初めてシステムと呼べるものになる。その意味では我々のものはまだまだ名前をつけるのもおこがましいくらいである。特に現状では数式処理というよりは整式処理といったほうが当たっている。

さて、初期の段階ではプロトタイプはPrologにより記述されたが、速度、効率その他の点からすぐに限界を悟り、現在のところUNIX上でC言語(および一部assembler)により記述されている。machineはSUN3およびμ VAXIIである。SUN3(260)はwork stationとはいうものの公称4MIPSのスピードを持ち、ある程度実用的なものができる可能性はある。現段階でインプリメントされている機能はおおよそ次の通りである:

- (有理数係数)多項式の四則演算
- 多項式のGCD
- 因数分解
- 超簡易言語
- その他(入出力切り換えなど)

実際のところ、上で述べたようにまだ全くシステムなどと呼べるものではなく、単なるサブルーチンの寄せ集めといったところである。そろそろ、アルゴリズムを単にインプリメントするだけでなく、システムとしての構成の検討を真剣にやらねばならぬ時期にきているようである。

3. 内部表現、メモリ管理

数式の内部表現は大きな問題であり、システムの効率、拡張性などを直接左右するものであるが今のところ、対象を有理係数多項式に限っているため、基本となる構造体は比較的単純な構造をしている。やや特殊と言えそうなのは、リスト構造より配列を基本としていることである。これは、lisp などの上にでなく、直接C言語で書きはじめたため、その方が書き易かったという、これもまたややなさけない理由からである。これにも関係することであるがC言語で直接書くということは、メモリ管理を自分でしなければならないということにつながる。これも、いずれはなんとかしようと思っているが、今のところ、函数の内部で生成され不必要になったものはその函数の責任で始末する(即ちfreeする)ということになっている。このままでは、このレベルでのインプリメントは自分しかできないことになるので、独自のメモリ管理ルーチンを書くか、或いはそのへんを自動的にやってくれるcompilerのようなものを書く必要があるだろう。半面、不必要な物がメモリに残ることが少ないためメモリ不足で計算不能になったりすることが(少なくともPrologなどに比較して)少ない、という利点がある。

4. 言語

これについては今のところほとんど手付かず、といった状態である。yaccで書いたbc並みのお手軽interpreterを作って使っている。まだ仕様も決定していない。

5. 入出力、ヒューマンインタフェース

UNIXの長所である入出力の容易さは、数式処理システムを使う上でも、また開発する上でも有利に働く。途中結果をファイルに出したり、ファイルから読み込んだりすることは極めて容易である。また、特別な努力なしに、マルチウィンドゥ環境で使用することもできる。これは、interactiveな使用においては大変心地良い。これらの長所を十分に生かしたシステム開発を目指している。

6. 因数分解、GCD

以下で、現在インプリメントされている因数分解、GCD 演算について、そこで用いられているアルゴリズムや、他のアルゴリズムとの比較などについて述べる。

a) 因数分解

• 整数上一変数多項式の因数分解

これについては、ごくごく普通の Berlekamp-Hensel アルゴリズムを採用している。しかし、相当初期に作ったため、つい凝り過ぎてしまったところもあり、もっと素直にやったほうが速い物ができたかも知れない。MACSYMA 、REDUCE が手元にないため同じ machine 、sample で比較ができないので残念だが、VAX11/780 上の VAXIMA の timing data が[1]にあるので、それと比較してみると、まずまずの物であると言える。(具体的な数字は別紙参照のこと。以下同様)

• 代数体上一変数多項式の因数分解

ここでいう代数体とは、有理数体に整数上既約な多項式の一根を添加して得られる体のことをいう。この体の上の一変数多項式は、形の上では二変数多項式として扱われる。これに対する因数分解は、Lagrange 補間によるもの、lattice によるもの、norm によるものがあるが、今のところ norm によるもののみがインプリメントされている。これは、単にインプリメントが簡単であったためであるが、予想以上に良く動く。しかし、ものの本にもあるとおり、最も時間がかかるのは、真の因子の主係数を有理数に正規化する部分であって、この部分をより最適化する必要がある。

b) GCD

最初に PRS によるものを作った。そして現在 EZ 法によるものを試作している。この 2 つの比較は興味ある問題である。大雑把にいって、GCD が元の多項式に"近い" ならば PRS の方が有利であろうし、1 に近ければ EZ による方が良さそうである。しかし、その境目がどのあたりにあるかは見当がつかない。現在実験中である。さて、EZ 法の長所は、中間表式膨張がある程度抑えられることであるが、unlucky な evaluation を行った場合、bound まで計算してはじめてハズレと分かる場合が多い。更に、その Hensel 構成は、項数も多くなることが多く時間がかかる。このように運に左右されるのを少しでも減らすため、並行して他の evaluation で一変数での GCD の計算を行わせ、現在計算中の因子の候補の次数より小さい次数の候補が出た時点で Hensel 構成を中止し、新たに Hensel 構成をはじめる、という方法を考えている。これにはメモリ管理の問題なども絡んでくるが、試してみる価値はあるだろう。

7. おわりに

今後何をすべきか、については既に幾つか述べてきたのでここでは繰り返さない。とにかく人様に使って頂けるようなものができあがるまでにはまだまだ相当の期間がかかることだけは間違いない。

参考文献

[1] Wang,P.S.: Implementation of a *p-adic* Package for Polynomial Factorization and other Related Operations.