長崎大学教養部紀要（自然科学篇）第35巻　第1号　1－7（1994年7月）

実二次体の類数の一つの計算法

工藤　愛知

（1994年4月28日受理）

A method of computation for class numbers of real quadratic number fields

Aichi KUDO

Abstract

The author improved the program "REALQF" version 2.3, which runs on UBASIC86 for PC-9801 and computes the class numbers of real quadratic number fields $K = Q(\sqrt{d})$. The new program "REALQF5" computes each class number of K with discriminant $d \approx 10^{12}$ in $1 \sim 1.5$ hours and each one with $d \approx 17 \times 10^{12}$ in $6 \sim 9$ hours by using PC-9801BA(40MHz).

1．序

二次体の類数はそれ自体が興味ある量であるだけでなく，その上で展開される整数論にも大きな影響を与える研究対象の一つである．二次体の類数や類群の生成元の表は代表的な整数論の書物の巻末に載せられ研究上の参考になっている．しかしたとえば判別式10桁までの実二次体あるいは虚二次体の類数の表を作るとなると印刷されるものは膨大な量になる．そこでパソコンなどで手軽に計算したり，知りたい系列の数表を作成できることが必要になってくる．木田祐司氏の開発による多倍長計算用 BASIC－UBASIC86 上の応用プログラム REALQF は実二次体の類数計算用のプログラムとして便利なものであるが，時間的あるいはメモリ容量的に判別式10桁あたりが限界となっていた．筆者は最近その改良を行いスピードおよび適用範囲の大幅な拡大の両面でかなり満足すべき成果を得たので本稿にその要点およびそれを支援するプログラムの概要をまとめることにする．

実二次体の類数および基本単数の計算用プログラム REALQF は，和田秀男氏のアルゴリズム［5］にしたがい、木田氏によってUBASIC用に書き直されたものである．今回の改良版 "REALQF5" はプログラムの本質的な流れは同じであるが，これまでネックになっていた部分の問題点を解消したものである．具体的には，既約実二次無理数をメモリへ登録する部分へ素因数分解を導入し，その利点を最大限に生かすため，組み込み関数 prmdiv に代わる機械語プログラム "pd31s" を作成したこと，また大量の既約二次無理数の記憶場所として EMS メモリを使用するため，そこでのサーチのための補助プログラムを用意したことことである．その結果，

EMS メモリを8MB以上とってあれば，判別式が14桁になったあたりまでの実二次体の類数が 4～5 時間台で計算できることになった（PC9801BA 40MHz 使用時）．パソコンの性能が上がればますます真価を発揮するものと思われる．

UBASIC86 は，数学的計算にとって特に優れた機能を備え，論文や研究発表の資料計算に幅広く利用されている評価の高いソフトである．ここに述べる補助プログラムでは，UBASIC86 の持つ機械語プログラムとのリンク機能，多倍長 IDIV ルーチン，MUL ルーチン，modpow，gcd，prmdiv などの組み込み関数，およびマクロファイル“UB.MAC”を利用した．また，必要な素数データも nxtprm 関数のおかげでごく短時間で作成することができた．UBASIC86 の最新バージョン（立教大学版）を送って下さった木田祐司氏に厚く感謝する．

2．REALQF version 2.3 のアルゴリズム

d を実二次体 $Q(\sqrt{d})$ の判別式とする．整数 a，b によって

$$\theta=\frac{b+\sqrt{d}}{2a},\qquad 0<b<\sqrt{d},\qquad \frac{\sqrt{d}-b}{2}<a<\frac{\sqrt{d}+b}{2},\qquad 4a\mid(d-b^2)$$

とあらわされる無理数 θ を，判別式 d に属する既約二次無理数という．このような θ の連分数展開の各項には判別式 d に属する既約二次無理数のみがあらわれ，またその連分数展開は循環することが知られる．判別式 d に属する既約二次無理数全体は連分数展開により同値類に類別され，その類全体と実二次体 $Q(\sqrt{d})$ の（広義）イデアル類全体の間に一体一対応が存在する．判別式 d に属する既約二次無理数の同値類のなかには上の条件に加え，さらに

$$\frac{\sqrt{d}-b}{2}<a<\sqrt{\frac{d-b^2}{4}}$$

をみたすものが含まれる［1］，［4］，［5］．REALQF version 2.3 における実二次体の類数計算のアルゴリズム［5］は次のようになる．

PART 1．平方因数を持たない正整数 M に対し実二次体 $Q(\sqrt{M})$ の判別式 D を計算する．

PART 2．$0<B<\sqrt{D}$，$B\equiv D\pmod 2$ をみたす整数 B の各々について

$$\frac{\sqrt{D}-B}{2}<A\leq\sqrt{\frac{D-B^2}{4}},\qquad A\mid\frac{D-B^2}{4}$$

なる整数 A を求め，B を一定ビットずらせて変数 X の上位へ，A を X の下位へおさめ，このような値 X を大きさの順にメモリへ登録する．

PART 3．X をメモリから取り出し X の上位と下位から復元した既約二次無理数を連分数展開する．展開に現れる既約二次無理数に一致するメモリ内の値をチェックして，展開が循環したら類を 1 つカウントする．メモリ内にまだチェックされない別の X があれば同様のことを繰り返す．カウントが終了したら類数を表示する．

3．REALQF5 での改良点

REALQF では AB 平面の上記の領域に含まれる B と A のすべての対（つい）に対して $\frac{D-B^2}{4}$ を A で割る割り算をおこなう．ただし被除数が奇数とわかるときは奇数の A のみで割る

が，この方法ではDが大きくなると領域の面積に比例（すなわちDに比例）して計算時間が増加する．しかも実際は大きなワード数の割り算が多くなってくるためさらにスピードダウンする結果となっていた．そこで次のように考えることにする．

$0 < B < \sqrt{D}$，$B \equiv D \pmod 2$ をみたす整数Bについて $y = \dfrac{D - B^2}{4}$ の素因数分解を

$$y = \prod_{i=1}^{n} p_i^{a_i}, \quad p_1 < p_2 < \cdots < p_n : \text{素数}, \quad a_1, a_2, \cdots, a_n \geq 1$$

とする．このとき，

$$\prod_{i=1}^{r} p_i^{a_i} \leq \frac{\sqrt{D} - B}{2}$$

なる素因数 p_i $(1 \leq i \leq r)$ だけからは PART 2 の条件をみたす y の約数 A を作ることはできない．また，もし $\sqrt{\dfrac{D - B^2}{4}} < p_n$ ならば，p_n が A の因数とならないことはあきらかである．

そこで，いま

$$z_m = \prod_{i=1}^{m} p_i^{a_i}, \quad 0 \leq m \leq n, \quad (z_0 = 1,\ z_n = y)$$

とおき

$$I(B) = \left\{ m : \frac{\sqrt{D} - B}{2} < z_m,\ p_m < \sqrt{\frac{D - B^2}{4}} \right\}$$

を考える．これらを用いて，

$$J(B, m) = \left\{ A : p_m \mid A,\ A \mid z_m,\ \frac{\sqrt{D} - B}{2} < A \leq \sqrt{\frac{D - B^2}{4}} \right\},\ (m \in I(B))$$

と定義すると

$$J(B) = \bigcup_{m \in I(B)} J(B, m)$$

が，PART 2 で求める各Bに対するAの集合になっていることがわかる．しかもこの右辺は disjoint な和集合である．なぜなら，Bに対するAは p_m $(m \in I(B))$ の少なくとも1つを因数に持たなければならず，また $J(B, m)$ は p_m を因数に持ちそれより大きな p_j $(m < j \in I(B))$ を因数に持たないAの集合だからである．

したがって，新しい PART 2 を次のようにする．

PART 2a. y を素因数分解し必須の素因数 p_m と被除数 z_m $(m \in I(B))$ を決める．

PART 2b. $J(B, m)$ の元 A を求める $(m \in I(B))$．

これらを $0 < B < \sqrt{D}$，$B \equiv D \pmod 2$ について実行する．

こうすれば，割り算の回数を減らすことができ，被除数も小さくできる．とくに，Dが多くの小さな素数を法として平方非剰余となっているような場合には $I(B)$ が空集合ということもあり得るので非常に効果的である．ただ，こうして得られたBとAの対から作るXが自然に大きさの順に並ぶということがないので並べ換えの操作が必要になる．そのためEMS上のデータをソート・サーチする機械語プログラム“qf55”と y の素因数分解のためのプログラム“pd31s”を

別に作成した．なお，

$$\frac{\sqrt{D}-B}{2} < z_m \leq \sqrt{\frac{D-B^2}{4}} \quad \text{かつ} \quad z_m \neq p_m$$

のとき，z_m の最小の素因数を q_m とすると，

$$J(B, m) = \left\{ A : p_m \mid A,\ A \mid z_m,\ \frac{\sqrt{D}-B}{2} < A \leq \frac{z_m}{q_m} \right\} \cup \{z_m\}$$

であり，また，$\sqrt{\frac{D-B^2}{4}} < z_m < \frac{D-B^2}{4}$ のとき

$$J(B, m) = \left\{ A : p_m \mid A,\ A \mid z_m,\ \frac{\sqrt{D}-B}{2} < A \leq \min\left\{ \frac{z_m}{q_m},\ \sqrt{\frac{D-B^2}{4}} \right\} \right\}$$

であることに注意すると計算量がさらに 5～20%減少する．以上が UBASIC86 の言語で書き表す部分である．

4．pd31s プログラム

上記の改良は UBASIC86 の組み込み関数 prmdiv を用いて実行しても十分効果があらわれる．実際，半年ほど前に，メインメモリのみで使用できる REALQF3 と，EMS メモリを用いる REALQF4 を作成した．計算できる限界はともに，判別式 $d \equiv 1 \pmod 8$ のとき $d = 2^{37}$ まで，そのほかのとき $d = 2^{36}$ までである．ただし REALQF3 は既約二次無理数の個数がメモリの限界をこえることが多く，REALQF4 は判別式が大きくなるにつれスピードが鈍った．

pd31s は引数の最小素因数を高速に求め，同時に取り扱える引数の範囲を広げることを目的として作成した 32ビット機専用の機械語プログラムである．2^{21} までの素数データとフェルマ素数判定法によって 2^{42} 以下の引数の最小素因数を求めることができる．これを併用する REALQF5 が実二次体の類数を計算できる限界は，判別式 $d \equiv 1 \pmod 8$ のとき $d = 8 * E$，そのほかのとき $d = 4 * E$（E は 2^{21} の直後の素数 2097169 の平方 4398117814561）である．

プログラムは UBASIC86 の dim 命令によって定義された配列変数が確保するメインメモリ上の領域に load され，call 命令によって実行される．用意した 2^{21} までの素数データをすべて用いるには “pd31s”から “pd36s”までの 6 つのファイルを別々の配列変数上に load する．次いで call PD%(W,P)（PD% は pd31s を load した配列変数名）を実行すると，変数 W の最小素因数が変数 P に返される．

pd31s はプログラム部と 2 から $5 * 2^{16}$ までの素数の下位ワード 16 ビット，pd32s は $5 * 2^{16}$ から $10 * 2^{16}$ までの素数の下位ワード，…というように構成され，それぞれおよそ 60000～50000 バイトの大きさである．メインメモリが不足する場合などは，はじめの数ファイルのみの利用も可能である．pd32s 以降に当初はプログラム部も入っていたが，現在はデータのみである．最初の 8 バイトは “pd31s”がそのファイルの存在を認識するための情報で，‘pd’ という 1 ワードの文字列定数，そのファイルの先頭データの上位ワード，そのファイルの次のファイルの先頭データの下位ワード，上位ワードという順に入っており，その次の 8 バイト目から素数データの下位ワードが 2 進数で並ぶ．上位ワードが切り替わる場所は 1 ワードの 0 で区切る．各ファイルの終わりは，区切りの 0 である．

pd31s は call されると，後続ファイルの所在を確認（初回のみ）したのち，

STEP 1．引数 W の絶対値を読み込み 2^{16} までの素数で割る．W が 2^{32} 以下の数であればこの STEP で答が P へ返される．

STEP 2．引数 W が 3 ワード数で，STEP 1 では答が得られなかった場合，W を対象に素数判定を行う．ただしそれに手間取るときは判定を打ち切って次の STEP 3 へ移る．合成数と判定されたときも同様である．なお，W が $2^{32}+2^{28}$ 以下の場合は，単純に割って行った方が早いためこの STEP へは入らず STEP 3 へ進む．

STEP 3．2^{16} 以上の素数で割ってみて答を返す．このように，最後には素数データで必要なだけ割るので，引数 W が最後に load された最後のデータの次の素数の平方（この値は初回に計算して保持）以下であれば必ず求められた値を返す．それより大きくて，かつ素因数が決定できなかったときは prmdiv 関数と同様に 0 を返す．

それぞれの STEP は 親のルーチンから CALL でよばれ RET で終了する．終了にあたって，W の素因数が見つかったときは，レジスタ SI に見つかった素因数のメモリのアドレスを返し，見つからないとき キャリーフラグ CF をセットする．このようにしたのは，これらの STEP を他の STEP や自分自身からも呼ぶことができるようにするためである．

STEP 1 と STEP 3 は単純なので，STEP 2 について述べる．この STEP は引数 W が W 自身の素因数であるかどうかをテストする．そのために用いるのは次の事実である［3］．

定理． 自然数 w に対して $w-1$ の素因数分解を

$$w-1=R\prod_{i=1}^{m}p_i^{e_i}$$

とする．ただし p_i は互いに異なる素数で，R はどの p_i とも互いに素な整数とする．このとき，各 i について，

$$a_i^{w-1}\equiv 1 \pmod{w}, \qquad (a_i^{\frac{w-1}{p_i}}-1,\ w)=1$$

なる自然数 a_i が存在し，かつ $R<\sqrt{w}$ であれば，w は素数である．

いま引数 W の値 w がこの STEP に入ってきたとする．w は $2^{32}+2^{28}$ 以上で，2^{16} 以下の素因数を持たない自然数である．さらに w は 2^{42} 以下で，2^{21} までの素数データは使用できる状態であるものとして見て行く．まず

FT 1． $2^{w-1}\equiv 1 \pmod{w}$

をテストする．もし不成立ならば w は合成数であるので STEP 3 へ行く．

FT 2． $w-1$ を素因数分解する．

素数データが十分なことより，とにかく素因数分解はできるのでそれを

$$w-1=\prod_{i=1}^{n}p_i^{e_i}, \qquad p_1<p_2<\cdots<p_n$$

とする．これについて，

$$\prod_{i=1}^{r-1} p_i^{e_i} < \sqrt{w} \leq \prod_{i=1}^{r} p_i^{e_i}$$

なる p_r から p_n までと，それに対応する巾指数 $\frac{w-1}{p_r}$ から $\frac{w-1}{p_n}$ までを記憶しておく．

$2^{w-1} \equiv 1 \pmod{w}$ は既に確かめてあるので

FT 3． $(2^{\frac{w-1}{p_r}} - 1, w) = 1$

をテストする．これが正ければ p_r に対する定理の条件は $a_r = 2$ によって満たされるので r を $r+1$ に変えて FT 3 へ行く．そうでないときは $b = 3, 5, \cdots, 31$（素数）を用いて

FT 4． $b^{w-1} \equiv 1 \pmod{w}$ かつ $(b^{\frac{w-1}{p_r}} - 1, w) = 1$

となる b があるかどうかをテストする．見つからなければ上記以外の b を用いるテストはせずに STEP 3 へ行く．また，FT 4 の前半が成立しないような b があったときは w は合成数であるから，同様に STEP 3 へ行く．

逆に，FT 4 を満たす b があったときは p_r に対する定理の条件は $a_r = b$ によって満たされるので r を $r+1$ に変えて FT 3 へ行く．

このようにして，記録しておいたすべての p_i $(r \leq i \leq n)$ に対して FT 3（または FT 4）のテストをパスすれば w は素数であるから CF = 0，SI = w のアドレスとして STEP 2 を CALL した親のルーチンへ戻る．

以上が STEP 2 のあらましであるが，なお次の 2 点を注意しておきたい．第一に，FT 2 の $w-1$ の素因数分解において再び STEP 2 を用いた方がよい場合がしばしばあらわれる．STEP 2 のそれまでの作業によるデータを保存し，現在の $w-1$ の素数判定をしたい因数をあらたな w として STEP 2 を行うことを 4 重まで許容した．2～3 重まで必要とする場合はかなりあり明瞭な効果がある．第二に，素数判定法はほかにもいろいろあるが，あまり強力なものはこの程度の小さい数の素因数を大量に迅速に求めようとする目的のためには不向きである．大きい整数の素数判定に対して上の定理は R が素数かどうかわからない不完全な分解のときにも w の素数判定に使えるという利点を持っている．ここでは逆に，R の方に小さい素因数を含めて，テスト FT 3，FT 4 の対象となる素因数 p_i の個数を少なくし，かつ，巾指数 $\frac{w-1}{p_i}$ を小さくした．さらに，w が素数の場合，小さな b でテストをパスしやすくなっている．

類似のものに

定理． $w - 1 = \prod_{i=1}^{n} p_i^{e_i}$ を $w-1$ の素因数分解とするとき，各 i に対して

$$a_i^{w-1} \equiv 1 \pmod{w}, \quad a_i^{\frac{w-1}{p_i}} \not\equiv 1 \pmod{w}$$

なる自然数 a_i が見つかれば w は素数である．

という判定法がある［6］，［7］．しかし，すべての p_i がテストの対象となり，w が素数でも，小さい p_i ではパスするまでにたくさんの b を必要とすることがある．

いずれにしても引数値 w が 1 ワードの因数を持たないことがわかった時点で素数判定を取り入れた効果は歴然としたものがあり(注1)，これを用いた REALQF5 では，判別式が 2 倍となっても実行時間は 1.6〜1.7 倍という結果が得られた．ただ，この計算法では判別式の性格によって結果が大きく異なるので，ここに述べたことは判別式 d について $d \equiv 1, 5 \pmod 8$ であるか $\equiv 0 \pmod 2$ であるかのほかに，実二次体 $Q(\sqrt{d})$ に対応する L 関数の値 $L(1, \chi)$ の大小によって分類した上でのことである（詳しくは REALQF3−5 のドキュメントファイルを参照のこと）．

最後に，pd31s は REALQF5 以外の UBASIC プログラムでも使用することができる．しかし，qf55 は EMS に書き込みを行うし，REALQF5 のプログラムに大きく依存しているので他の用途には利用できない．

このプログラムを用いて A. Hédi［1］の結果などをたくさん計算してみていろいろ面白いことが観察できた．REALQF4 までとくらべると計算可能な範囲が判別式で64倍であるからまだまだ利用価値は広がると思われる．ワークステーションなどで動く別の単一の言語に書き直すことも可能であろう．ここで取り入れた考え方がなるべく多くの実りをもたらすことを望む．

（注 1）本稿提出後，w が 2^{10} 以下の因数を持たないことがわかった時点で w の素数判定を行うようにした．

参考文献

［1］A. Hédi, Cycles canoniques d'ideaux reduits et nombre des classes de certains corps quadratique reels, Nagoya Math. J., **103** (1986), 127−132.
［2］木田祐司，UBASIC86（多倍長計算用 BASIC）第8.3版ユーザーズマニュアル，日本評論社，1992.
［3］森本光生，フェルマー型の素数について，数学，**38** No. 4 (1986), 350−354.
［4］高木貞治，初等整数論講義，第12版，共立出版株式会社，昭和32年.
［5］和田秀男，実二次体の類数表，上智大学数学講究録 No.10, 1981.
［6］和田秀男，高速乗算法と素数判定法，上智大学数学講究録 No.15, 1983.
［7］和田秀男，素因子分解とコンピュータ，数学，**38** No. 4 (1986), 345−350.