課題１：最大公約数の計算（その２）

講義ノート

前回の実習では，gcd1, gcd2 を実際に動かして，最悪の例題に対する計算時間（除余の回数）を測った．今度は 200 個のランダムな入力に対する計算時間を測ってみよう．

その前にまず，前回の gcd1 を次のように改良する．

```
 program gcd1ow (int x,y)
1   N ← x と y の小さな方
2   for (k ← N ∼ 1) {
3      if ((x % k = 0) ∧ (y % k = 0)) {
4         ans ← k
5         break     ⟸ 繰り返しをやめる
6      }
7   }
8   return(ans)
 program end.
```

プログラム K1-3： gcd1 の改良型

前回の gcd1 は，繰り返しをつねに N 回実行したが，今回は最大公約数が求まった時点ですぐにやめるようになっている．したがって $x = 2^n - 1, y = 2^n - 1$ のような入力例では，即座に答えを出せるようになった．（では最悪の入力例とはどんなもの？）この gcd1ow と gcd2 の平均の計算時間を評価してみよう．

プログラムの実行を制御する

実験では 2 つの数をランダムに生成しなければならない．そのためのプログラム genrand は用意されている．けれども多数のデータを，いちいち作って gcd1ow などで実行していたら大変である．そこで我々のシステム（Unix）上で，いろいろなプログラムをまとめて実行する方法を使おう．たとえば，次のプログラム 2 に示したのは $x = 6 \sim 10, y = 2x + 1$ に対して，gcd2 を実行させる方法である．

```
#!/bin/tcsh −f          ⟸ これはおまじない
@ x = 6                 ⟸ 変数 x の値を 6 にセットする
while ( $x <= 10 )      ⟸  変数 x の値が 10 以下の間,
   @ y = 2 * $x + 1           以下の行（end）までを繰り返す
   ./gcd2 $x $y
   @ x = $x + 1
end
```

プログラム J1-1.sh： gcd2 を 5 回実行

このような命令文をタイプしたファイルを作っておけば（それをここでは J1-1.sh とよぶ），あとは `./J1-1.sh` とタイプするだけで実行してくれるのである．このようなファイルをシェルファイルといい，一般に上のような命令群をシェルプログラムという[1]．

```
#!/bin/tcsh −f @ total = 0
@ x = 6
while ( $x <= 10 )
   @ y = 2 * $x + 1
   @ t = `./gcd2T $x $y`
   @ total = $total + $t
   @ x = $x + 1
end
@ ave = $total / 5
echo $total $ave
```

プログラム J1-2.sh： 実行＆集計

```
#!/bin/tcsh −f @ total = 0
@ x = $1
while ( $x <= $2 )
   @ y = 2 * $x + 1
   @ t = `./gcd2T $x $y`
   @ total = $total + $t
   @ x = $x + 1
end
@ave = $total / (($2 - $1) + 1)
echo $total $ave
```

プログラム J1-3.sh： 範囲指定可能

実際には，実行される計算時間を集計しなければならない．gcd1ow や gcd2 は，答えと時間を出力するようになっていたが，我々の興味は時間だけなので，時間だけを出力するようにプログラムを修正しておいた.（そのプログラムを各々 gcd1owT, gcd2T と名付けることにする.）このプログラムを使って，計算時間を集計するには，たとえば上のプログラム 3 のようなシェルプログラムを用意すればよい.（注意：シェルプログラムでは，複数の演算を 1 つの式で行うときは括弧を使うこと.）

プログラム 3 では変数 x のとり得る範囲は 6 ～ 10 だったが，これをシェルプログラムを実行させるときに指定できるようにしたのが，プログラム 4 である．このプログラムを `J1-3.sh` というファイルにしておけば，たとえば `./J1-3.sh 31 44` とタイプすると，31 ～ 44 の範囲で gcd2T を実行し集計をとる．プログラム中の `$1` は，このプログラムを実行させるときに与えられたデータのうちの 1 番目のデータを，`$2` は 2 番目のデータを指し示している．

【宿題】（〆切：10 月 24 日）

問 1 指定された桁数 n に対し，その 2 進桁数の 2 数を 50 組だけランダムに生成し，gcd1ow の計算時間（除余の回数）を測り，集計して平均計算時間を求めるシェルプログラムを記述せよ．なお，指定された 2 進桁数の数をランダムに作るプログラム genrand を使用してもよい.（genrand については「実験ノート」の説明を参照.）

問 2 （オプショナル）

gcd1 をもう少し速いものにする工夫を提案してみよう．ただし，提案はわかりやすく具体的に書くこと．なお，おもしろい工夫の提案はプログラム化してあげます.

実験ノート

[1]注意：シェルプログラムでは，@ の直後には空白を入れなければならない.（もしかすると演算子の前後にも！）その反対に $ 記号と変数名の間には空白を入れてはならない.

１．実験課題

最大公約数を計算する 2 つのプログラム (gcd1ow, gcd2) に対し，入力される 2 数の合計桁数（２進）$n = 10 \sim 50$ の範囲で，ランダムなデータに対して各々 50 回（それ以上は大変！）実行し，平均の計算時間（除余演算の回数）を測定せよ．その結果を gnuplot で表示し，2 つのプログラムで使われているアルゴリズムの平均時間計算量を求めよ．

２．使用ソフト

実験のための次のソフト（コマンド）等は，ディレクトリ `~owatanab/pub/gcd` に用意してある．

実験データ生成用

genrand（プログラムのソースは genrand.c）

[自分のプロンプト] `./gen` n $seed$

- n は欲しいデータの桁数．$seed$ は乱数の初期値．適当な値を使う．
  注：$seed$ は毎回違う値を使わなければならない．(逆にいうと，違いさえすればよい.)
- n 桁の自然数をランダムに作り，それを出力する．

実験対象プログラム

gcd1owT, gcd2T（プログラムのソースは各々 gcd1owT.c, gcd2T.c）

[自分のプロンプト] `./gcd`i x y

- 最大公約数を求めたい自然数の対 x, y.
- x と y の最大公約数を求める．(ただし答えは出さない.)
- 計算中に行われた除余演算の回数を出力．

３．レポートの構成について

レポートは以下のような構成にすること．(前回のレポートでレポート×の人は書き方に注意すること.)

1. 実験手順：前回と同様．(省かないで欲しい.)
2. 実験結果のまとめ：実験で得られた結果を表とグラフにまとめたもの（詳しい個々のデータは付録にする）．また，各アルゴリズムの計算量（これを各々 $t_1(n), t_2(n)$ とする）に対し，実験で gnuplot を使って得られた式を書く．
3. 考察：t_1, t_2 がなぜそのような式になるのかの考察．
4. 付録：今回の実験結果（データが多量にある場合には，その抜粋）．

参考文献

数のことについて少し勉強してみたい人のために

図書館で 整数論，数論 の本を手にとって見てみましょう．難しく書いてある本もあります．めげたら読みやすそうな本を渡り歩きましょう．