MENENTUKAN PRIMALITAS SEMUA BILANGAN YANG TERDAPAT PADA SELANG TERTENTU SECARA BRUTE FORCE

E.Z. Adnan Kashogi
13505094

Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
Jalan Ganesha 10 Bandung, Jawa Barat
email : ez_adnan@yahoo.co.uk

ABSTRAK

Bilangan prima merupakan suatu bilangan asli yang memiliki sifat unik yaitu bilangan prima hanya habis jika dibagi dengan 1 atau -1 dan dengan dirinya sendiri atau negatif dirinya sendiri. Bilangan prima terdapat tak hingga banyaknya, baik bilangan prima positif maupun bilangan prima negatif, namun bilangan prima terbesar yang berhasil diketahui adalah $2^{32.582.657}-1$ (September 2006). Bilangan ini mempunyai 9.808.358 digit dan merupakan bilangan prima Marsenne yang ke-44 ditemukan oleh Curtis Cooper dan Steven Boone yang merupakan profesor-profesor dari *University of Central Missouri*.
Untuk menentukan semua bilangan prima yang terdapat pada selang tertentu tidaklah mudah, algoritma yang sering digunakan untuk permasalahan ini adalah algoritma *brute force*. *Brute force* merupakan metode pemecahan masalah yang paling sederhana dan paling mudah dilakukan. *Brute force* memiliki pendekatan yang lempang (*straightforward*) untuk memecahkan suatu masalah, biasanya didasarkan langsung pada pernyataan masalah (*problem statement*) dan definisi konsep yang dilibatkan. Algoritma *brute force* menyelesaikan masalah dengan sangat sederhana, langsung dan dengan cara yang jelas (*obvious way*).
Algoritma *brute force* umumnya tidak cerdas dan tidak mangkus, karena membutuhkan jumlah langkah yang besar dalam penyelesaiannya, terutama bila masalah yang dipecahkan berukuran besar (dalam hal ini masukannya). Meskipun demikian *brute force* dapat diterapkan pada sebagian besar masalah. Hampir tidak ada masalah yang tidak dapat dipecahkan dengan metode *brute force*. Bahkan ada masalah yang hanya dapat dipecahkan dengan metode *brute force* seperti menentukan semua bilangan prima yang terdapat pada selang tertentu ini.

Kata kunci: *bilangan prima, brute force, Marsenne*

1. PENDAHULUAN

Sebuah bilangan bulat p dengan p bukan 1 atau -1 dikatakan prima jika dan hanya jika pembagi yang dapat habis membagi bilangan p tersebut hanyalah 1, -1, p, atau –p. Sifat bilangan yang demikian disebut sebagai primalitas.

Terdapat tak hingga bilangan prima, baik bilangan prima positif maupun bilangan prima negatif. Contoh sepuluh bilangan prima positif pertama yaitu 2, 3, 5, 7, 11, 13, 17, 19 , 23 , 29 , sedangkan bilangan prima negatif contohnya yaitu -2, -3, -5, -7 dan lain-lain. Bilangan prima terbesar yang diketahui per September 2006 adalah $2^{32.582.657}-1$. Bilangan ini memiliki 9.808.358 digit dan merupakan bilangan prima Mersenne yang ke-44.

Oleh karena itu, untuk menentukan bilangan prima pada selang tertentu tidaklah mudah, dibutuhkan suatu algoritma yang dapat memecahkan permasalahan tersebut.

Pada makalah ini akan dibahas algoritma yang dapat menentukan semua bilangan prima (dibatasi hanya bilangan prima positif saja) yang terdapat pada selang tertentu diantaranya dengan memanfaatkan sifat unik dari bilangan prima itu sendiri, dengan menggunakan algoritma *sieve of erastothenes*, atau algoritma *sieve of atkin* yang merupakan penyempurnaan dari algoritma *sieve of erastothenes*.

Masing-masing algoritma tersebut memiliki kelebihan dan kekurangan yang juga akan dibahas pada makalah ini.

2. BILANGAN PRIMA

Bilangan prima merupakan sebuah bilangan bulat p baik positif maupun negatif yang hanya habis jika dibagi dengan 1, -1, p atau –p. Bilangan prima mempunyai beberapa sifat sebagai berikut :

1. Dalam sistem bilangan basis 10, bilangan prima selalu memiliki digit terakhir (angka satuan) 1, 3, 7, atau 9, kecuali untuk bilangan 2 dan 5, karena

angka satuan 0, 2, 4, 6, 8 (untuk bilangan selain 2) berarti kelipatan dari 2 dan angka satuan 5 untuk bilangan selain 5 berati kelipatan lima.

2. Banyaknya bilangan prima yang kurang dari atau sama dengan n mendekati $\frac{n}{\log n}$ ketika n sangat besar. (Teorema bilangan prima).
3. Jika suatu bilangan prima p membagi hasil kali ab, dengan a, b $\in$ N, maka p membagi a; atau p membagi b. (Lemma Euklid).
4. Jika p prima, maka a^p-a habis dibagi p. (*Fermat Little Theorem*).
5. Jika n adalah bilangan bulat dengan $n>1$, maka pasti ada bilangan prima p, di mana $n<p<2n$. (Generalisasi Postulat Bertrand, postulat awal mengatakan bahwa jika n adalah sebuah bilangan bulat positif yang lebih besar dari 3, paling sedikit ada satu bilangan prima di antara n dan $2n$ - 2. Postulat ini pertama kali dibuktikan pada tahun 1850 oleh Pafnuty Chebyshev. Pada tahun 1932, Paul Erdös memberikan bukti menggunakan koefisien binomial).
6. Sebuah bilangan q; (q > 1) adalah bilangan prima jika dan hanya jika (q - 1)! – 1 habis dibagi (*divisible*) oleh q (teorema Wilson/ *Al-Haytham*), sebaliknya bilangan bulat n; (n>4) adalah komposit jika dan hanya jika (n-1)! habis dibagi n.
7. Jika p adalah bilangan prima selain 2, 3, dan 5, maka $\frac{1}{p}$ selalu berbentuk desimal berulang dengan jumlah angka yang muncul pada setiap pengulangan berjumlah 1-p, contohnya:
   $\frac{1}{23}$ = 0.043478260869565217391 3...; 22 angka berulang
   $\frac{1}{29}$ = 0.034482758620689655172413793 1...; 28 angka berulang
8. Jika p adalah bilangan prima yang lebih besar dari 6, maka p mod 6 hasilnya 1 atau 5; dan p mod 30 adalah 1, 7, 11, 13, 17, 19, 23 atau 29.
9. Jika p>1, maka polinom $x^{p-1} + x^{p-2} + ... + 1$ tidak dapat diperkecil (disederhanakan) lagi jika dan hanya jika p adalah bilangan prima.
10. Konstanta Copelan-Erdos, 0.235711131719232931374143, suatu bilangan irasional, didapat dengan mengurutkan bilangan prima.
11. Teorema bilangan prima mengatakan bahwa banyak bilangan prima yang kurang dari x adalah mendekati $\frac{1}{\ln x}$ (dengan kata lain, jika x sangat besar, kemungkinan bahwa suatu bilangan selain x adalah prima sangat sedikit).
12. Karena bilangan prima positif, p hanya habis dibagi oleh 1 dan p, maka p mod i = 0 jika dan hanya jika $i \neq p$ dan $i \neq 1$, sehingga kita dapat merumuskan:
    $\prod_{i=2}^{p-1} p \bmod i \neq 0$.
13. Dalam setiap deret aritmatika divergen, yaitu a, $a + q$, $a + 2q$, $a + 3q$,..., $a + nq$ dengan a dan $q \geq 1$ serta a dan q koprima (prima relatif) serta $n \neq 0$, maka terdapat bilangan prima yang tak hingga banyaknya, atau dengan kata lain ada tak hingga bilangan prima yang kongruen dengan $a \bmod q$ (Teorema Dirichlet, dibuktikan tahun 1935).

2.1 Menentukan Primalitas Semua Bilangan Pada Selang Tertentu Secara Brute Force

Bilangan prima memiliki sifat yang unik yaitu :

1. Semua bilangan yang lebih kecil atau sama dengan 100 merupakan bilangan prima jika bilangan tersebut tidak habis dibagi dengan bilangan 2, 3, 5 dan 7.
2. Semua bilangan yang lebih kecil atau sama dengan 400 merupakan bilangan prima jika bilangan tersebut tidak habis dibagi dengan bilangan 2, 3, 5, 7, 11, 13, 17 dan 19.
3. Semua bilangan yang lebih kecil atau sama dengan 10.000 merupakan bilangan prima jika bilangan tersebut tidak habis dibagi dengan bilangan 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, dan 97.
4. Semua bilangan yang lebih kecil atau sama dengan 1.000.000 merupakan bilangan prima jika bilangan tersebut tidak habis dibagi dengan semua bilangan prima yang terdapat pada selang 1 sampai 1000.

Dari sifat billangan prima tersebut kita dapat menentukan primalitas suatu bilangan pada selang tertentu secara *brute force* sebagai berikut :

1. Untuk n suatu bilangan yang lebih kecil atau sama dengan 100, maka banyak nya bilangan prima pada selang 1 sampai n adalah :

```
//Algoritma berikut ditulis dalam bahasa java
/*Program akan mengoutputkan semua bilangan
prima yang terdapat pada selang 1 sampai n
dengan n <= 100*/
```

```
public class TestPrime {
  public static void main (String args []) {
   int i;
   for (i = 2; i<=n; i++) {
    if ((i==2)||(i==3)||(i==5)||(i==7)) {
      System.out.print(i +", ");
    } else {
     if ((i%2!=0) && (i%3!=0) && (i%5!=0) &&
(i%7!=0)) {
      System.out.print(i +", ");
     }
    }
   }
  }
}
```

2. Untuk n suatu bilangan yang lebih kecil atau sama dengan 400, maka banyaknya bilangan prima pada selang 1 sampai n adalah :

```
//Algoritma berikut ditulis dalam bahasa java
/*Program akan mengoutputkan semua bilangan
prima yang terdapat pada selang 1 sampai n
dengan n <= 400*/

public class TestPrime {
  public static void main (String args []) {
    int i;
    for (i = 2; i<=n; i++) {
      if ((i==2) || (i==3) || (i==5) ||
(i==7)||(i==11) || (i==13) || (i==17) ||
(i==19)) {
       System.out.print(i +", ");
     } else {
       if ((i%2!=0) && (i%3!=0) && (i%5!=0)
&& (i%7!=0) && (i%11!=0) && (i%13!=0) &&
(i%17!=0) && (i%19!=0)) {
          System.out.print(i +", ");
       }
     }
   }
  }
}
```

3. Untuk n suatu bilangan yang lebih kecil atau sama dengan 10.000, maka banyaknya bilangan prima pada selang 1 sampai n adalah :

```
//Algoritma berikut ditulis dalam bahasa java
/*Program akan mengoutputkan semua bilangan
prima yang terdapat pada selang 1 sampai n
dengan n <= 10.000*/

public class TestPrime {
  public static void main (String args []) {
    int i;
    for (i = 2; i<=n; i++) {
      if ((i==2) || (i==3) || (i==5) ||
(i==7)||(i==11) || (i==13) || (i==17) ||
(i==19)|| (i==23) || (i==29) || (i==31)||
(i==37)||(i==41) || (i==43) || (i==47)||
(i==53)|| (i==59) || (i==61) || (i==67)||
(i==71)|| (i==73)||(i==79) || (i==83) ||
(i==89) || (i==97)) {
       System.out.print(i +", ");
     } else {
       if ((i%2!=0) && (i%3!=0) && (i%5!=0)
&& (i%7!=0) && (i%11!=0) && (i%13!=0) &&
(i%17!=0) && (i%19!=0)&& (i%23!=0) &&
(i%29!=0) && (i%31!=0) && (i%37!=0) &&
(i%41!=0) && (i%43!=0) && (i%47!=0) &&
(i%53!=0)&&(i%59!=0) && (i%61!=0) &&
(i%67!=0) && (i%71!=0) && (i%73!=0) &&
(i%79!=0) && (i%83!=0) && (i%89!=0)&&
(i%97!=0)) {
          System.out.print(i +", ");
       }
     }
   }
  }
}
```

Dari algoritma di atas kita dapat melihat bahwa untuk menentukan primalitas suatu bilangan pada selang tertentu sangat tidak efektif, karena untuk menentukan bilangan prima pada selang 1 sampai n dimana n lebih kecil atau sama dengan 100 kita harus mengetahui semua bilangan prima yang terletak pada selang 1 sampai 10. Jika n lebih kecil atau sama dengan 400 maka kita harus mengetahui semua bilangan prima yang terletak pada selang 1 sampai 20. Bahkan untuk n yang lebih besar lagi yaitu lebih kecil atau sama dengan 10.000 kita harus mengetahui semua bilangan prima yang terletak pada selang 1 sampai dengan 100.

2.2 ALGORITMA *SIEVE OF ERASTOTHENES*

Algoritma *Sieve Of Erastothenes* (saringan *Erastothenes*) merupakan algoritma untuk menentukan primalitas suatu bilangan asli yang telah ada sejak zaman Yunani Kuno. Berikut penjelasan tentang algoritma *Sieve of Erastothenes* :

1. Tulis daftar bilangan - bilangan yang akan diuji primalitasnya, dari 2 hingga bilangan terbesar yang diinginkan. Sebut saja daftar ini sebagai daftar A.
2. Tandai angka dua dari daftar A dan pindahkan ke dalam daftar yang lain yang menampung bilangan – bilangan prima yang kita cari .
3. Coret semua angka yang merupakan kelipatan dari 2 dalam daftar A.
4. Angka yang ditemukan berikutnya dalam daftar A adalah bilangan prima. Tandai dan pindahkan ke daftar B.
5. Coret semua angka yang merupakan kelipatan bilangan prima dalam daftar A tadi, angka yang telah dicoret sebelumnya tidak perlu dicoret lagi.
6. Ulangi proses 4 dan 5 sampai tidak ada angka yang tersisa pada daftar A.

Berikut ini *pseudocode* dari algoritma *Sieve of erastothenes* :

```
// arbitrary search limit
limit ← 1.000.000

// assume all numbers are prime at first
is_prime(i) ← true, i ∈ [2, limit]

for n in [2, √limit]:
    if is_prime(n):
        // eliminate multiples of each prime,
        // starting with its square
        is_prime(i) ← false, i ∈ {n², n²+n,
n²+2n, ..., limit}

for n in [2, limit]:
    if is_prime(n): print n
```

Meskipun algoritma ini cukup mangkus dalam menguji primalitas semua bilangan yang terdapat pada selang tertentu, karena kita tidak perlu mengetahui bilangan prima lainnya, namun algoritma ini memerlukan waktu yang lama untuk sebuah proses penentuan primalitas semua bilangan, selain itu algoritma ini harus dimulai dari bilangan 2 agar tidak ada bilangan yang terlewatkan.

2.3 ALGORITMA *SIEVE OF ATKIN*

Algoritma ini ditemukan oleh A. O. L. Atkin and Daniel J. Bernstein, merupakan algoritma modern yang digunakan dalam penentuan primalitas suatu bilangan. Algoritma *Sieve of Atkin* merupakan pengembangan dari algoritma *Sieve of erastothenes* yang telah dioptimasi. Algoritmanya adalah sebagai berikut:

1. Semua sisa pembagian adalah sisa modulo 60 (*all remainders are modulo-sixty remainders*)
2. *x* dan *y* bilangan positif real

Algoritma:

1. Buat daftar hasil (*result list*), diisi dengan 2, 3, dan 5.
2. Buat daftar saringan (*sieve list*) dengan input semua bilangan asli positif, seluruh input pada daftar ini bukan merupakan bilangan prima.
3. Untuk bilangan dengan sisa modulo 60 nya adalah kelipatan 2, atau 3, atau 5 dapat diabaikan, karena bilangan tersebut sudah pasti tidak prima.
4. Untuk setiap input (n) di *sieve list*:
    - Jika n mod 60 = 1, 13, 17, 29, 37, 41, 49, atau 53, cari solusi dari $4x^2 + y^2 = n$.
    - Jika n mod 60 = 7, 19, 31, atau 43, cari solusi dari $3x^2 + y^2 = n$.
    - Jika n mod 60 = 11, 23, 47, atau 59, cari solusi dari $3x^2 - y^2 = n$ dengan x>y.
    - Jika n mod 60 = nilai lainnya, tinggalkan saja.
5. Mulai dengan nilai terkecil pada *sieve list*.
6. Ambil bilangan selanjutnya pada *sieve list* yang masih merupakan bilangan prima.
7. Masukkan bilangan yang ada di *result list.*
8. Kuadratkan angka tersebut dan tandai setiap nilai kuadrat sebagai "bukan bilangan prima".
   Ulangi langkah 5 sampai langkah 8.

Berikut ini *pseudocode* dari algoritma *Sieve of erastothenes* :

```
// arbitrary search limit
limit ← 1.000.000

// initialize the sieve
is_prime(i) ← false, i ∈ [5, limit]

// put in candidate primes:
// integers which have an odd number of
// representations by certain quadratic forms
for (x, y) in [1, √limit] × [1, √limit]:
    n ← 4x²+y²
    if (n ≤ limit) ∧ (n mod 12 = 1 ∨ n mod 12 =
5):
        is_prime(n) ← ¬is_prime(n)
    n ← 3x²+y²
    if (n ≤ limit) ∧ (n mod 12 = 7):
        is_prime(n) ← ¬is_prime(n)
    n ← 3x²-y²
    if (x > y) ∧ (n ≤ limit) ∧ (n mod 12 = 11):
        is_prime(n) ← ¬is_prime(n)

// eliminate composites by sieving
for n in [5, √limit]:
    if is_prime(n):
        // n is prime, omit multiples of its
square; this is
        // sufficient because composites which
managed to get
        // on the list cannot be square-free
        is_prime(k) ← false, k ∈ {n², 2n², 3n²,
..., limit}

print 2, 3
for n in [5, limit]:
    if is_prime(n): print n
```

3. KESIMPULAN

Beberapa kesimpulan yang dapat diambil dari makalah ini adalah :

1. Algoritma *brute force* merupakan algoritma yang menyelesaikan suatu masalah secara sederhana sehingga mudah dimengerti.
2. Algoritma *brute force* dapat menyelesaikan hampir semua masalah yang ada meskipun algoritma ini tidak mangkus dan umumnya tidak "cerdas".
3. Untuk beberapa masalah seperti masalah menentukan semua bilangan prima pada selang tertentu yang telah dijelaskan di atas hanya algoritma *brute force* lah yang dapat digunakan.

REFERENSI

[1] http://search.mit.edu
Tanggal akses : 16 Mei 2007
Waktu akses : 15:30

[2] http://en.wikipedia.org
Tanggal akses : 16 Mei 2007
Waktu akses : 15:45

[3] Munir, Rinaldi. Strategi Algoritmik. Departemen Teknik Informatika (ITB). 2007