3 Prime Tests, 3 Factorization Algorithms & BOINC

Mathematics → Number Theory → Computational Number Theory → Primality Testing & Factorization

Primality Testing:
Is a given integer prime?
ie exactly divisible only by itself and 1, eg 7

Integer Factorization:
Find a (unique) list of all the (prime) divisors of a given integer
eg 15=3*5

Naïve prime test:
Trial divide by all integers up to square root

Naïve factorization algorithm:
Trial divide by all integers up to square root

Running time of naïve prime test: exponential = kn^(1/2)
(Storage space required negligible)

Running time of naïve factorization algorithm: exponential = kn^(1/2)
(Storage space required negligible)

Fermat's Little Theorem:

$a^p = a \pmod p$

ie for p prime, raising any integer, a, to power p and subtracting a will give a result divisible by p

Converse:

$a^n = a \pmod n \Rightarrow n$ prime

Often, but NOT always true

Russian Peasant exponentiation:
Exponentiation by squaring ie binary ladder
eg consider a^8 = ((a^2)^2)^2
Running time: polynomial = kln(n)

Fermat probable (pseudo-)prime test (NOT deterministic)
Implementations in NT packages common

Wanless' factorization algorithm (unproven)
we2tr34.cpp
& BOINC: WEP-M+2 (+ others?!)

Running time of Fermat pseudoprime test:
polynomial = kln(n)
(Storage space required negligible)

Running time of Wanless' factorization algorithm:
polynomial = kln(n)
(Storage space required negligible)

Pocklington primality test (deterministic)
Relies on (partial) factorization of p-1 to generate recursive list of successive p
(uses Fermat's little theorem)
(isn't always able to work)

Pollard factorization algorithm
Relies on (complete) factorization of p-1, with p being factor to find
(uses Fermat's little theorem)
(isn't always able to work easily)

Elliptic Curves

Way of introducing controlled variation into the system, essentially mapping a geometric transform wrt a cubic equation (eg $y^2=x^3+ax+b$)

ECPP primality test (from Pocklington)

ECM factorization algorithm (from Pollard)

Running time of ECPP test: polynomial = kln(n)^4
(Storage space required negligible)

Running time of ECM factorization algorithm: near polynomial
(Storage space required negligible)

ECPP Implementations
Windows: Primo
Linux: ECPP (Morain)
Mac (+W/L): GMP-ECPP (open source)

ECM Implementations
ecm15.cpp (tutorial)
(M/W/L): GMP-ECM (open source)

ECPP & BOINC (general test)
PrimeGrid?

ECM & BOINC
yoyo@home
(record 68-digit factor)

3 Prime Tests, 3 Factorization Algorithms & BOINC

- Naïve prime test
- Pocklington prime test (FLT)
- ECPP (elliptic curves)
- Fermat pseudoprime test

- Naïve factorization algorithm
- Pollard factorization algorithm (FLT)
- ECM (elliptic curves) (yoyo@home)
- Wanless factorization algorithm (WEP-M+2)