AKS Primality Algorithm

Alexandra Carvalho

MEIC 2003

Plan of the talk

- Brief history
- Notation
- Main idea
- AKS primality algorithm
  - Correctness
  - Complexity
- Conclusion

Brief history

1976 - Miller: deterministic polynomial time
(assuming Extended Riemann Hypothesis)

1980 - Rabin: randomized polynomial time

1983 - Adleman, Pomerance, Rumely:
deterministic $O((\log(n))^{O(\log(\log(\log(n))))})$

1986 - Goldwasser, Kilian: randomized algorithm
with expected polynomial time in almost all inputs

1992 - Adleman, Huang: randomized polynomial time

2002 - Agrawal, Kayal, Saxena:
deterministic polynomial time $\tilde{O}(\log^{12}(n))$

Notation

The congruence $p(x) \equiv q(x) \mod (h(x), n)$:

- $h(x)$ divides $p(x) - q(x)$;
- all coefficients are taken modulo n.

The asymptotic notations:

- An upper bound on a function:

$$O(g(n)) = \{f(n) : \exists_{c,n_0>0} \forall_{n \geq n_0} \ 0 \leq f(n) \leq cg(n)\};$$

- An upper bound on a function with logarithmic factors ignored:

$$\tilde{O}(g(n)) = \cup_{k \in \mathbb{N}} O(g(n) \log^k(g(n)));$$

- Note that $\tilde{O}(\log^k(n)) \subseteq O(\log^{k+1}(n))$.

Main Idea

Proposition [Identity]:

1. n prime $\Rightarrow (x-a)^n \equiv (x^n - a) \mod n$;
2. n composite, $\gcd(a,n) = 1 \Rightarrow (x-a)^n \not\equiv (x^n - a) \mod n$.

Primality test from Identity:

- find a such that $\gcd(a,n) = 1$;
- by 2. if ($(x-a)^n \equiv (x^n - a) \mod n$) then n is prime;
- by 1. if ($(x-a)^n \not\equiv (x^n - a) \mod n$) then n is composite.

We have to compute $n+1$ coefficients :-(

Main Idea

Proposition [AKS prime for n]: there is always a small prime r such that

- $r \in O(\log^6(n))$;
- $r - 1$ largest prime factor $= q \geq 4\sqrt{r}\log(n)$ and $n^{(r-1)/q} \not\equiv 1 \bmod r$.

Definition [Suitable AKS prime for n]: a suitable AKS prime r is such that

- r is an AKS prime for n;
- $gcd(m, n) = 1$ for all $1 \leq m \leq r$.

If there is not a suitable AKS prime r for n then n is composite.

Main ideia

Proposition [AKS Identity]: if there is a suitable AKS prime r for n,

1. n prime $\Rightarrow \forall_{1 \leq a \leq \lfloor 2\sqrt{r}\log(n) \rfloor + 1} (x-a)^n \equiv (x^n - a) \bmod (x^r - 1, n)$;
2. $n \neq p^e \Rightarrow \exists_{1 \leq a \leq \lfloor 2\sqrt{r}\log(n) \rfloor + 1} (x-a)^n \not\equiv (x^n - a) \bmod (x^r - 1, n)$.

A quasi primality test from AKS Identity:

- find a suitable AKS prime r for n;
- if (there is not a suitable r for n) then n is composite;
- by 1. if ($\exists_a (x-a)^n \not\equiv (x^n - a) \bmod (x^r - 1, n)$) then n is composite;
- by 2. if ($\forall_a (x-a)^n \equiv (x^n - a) \bmod (x^r - 1, n)$) then $n = p^e$.

Main ideia

Primality test from AKS Identity:

- if (n is of the form a^b, $b > 1$) then n is composite;
- find a suitable small prime r for n;
- if (there is not a suitable r for n) then n is composite;
- by 1. if ($\exists_a (x-a)^n \not\equiv (x^n - a) \bmod (x^r - 1, n)$) then n is composite;
- by 2. if ($\forall_a (x-a)^n \equiv (x^n - a) \bmod (x^r - 1, n)$) then n is prime.

We only have to compute at most r coefficients :-)

AKS Primality Algorithm

```
Input:  integer $n \geq 2$
Output:  if $n$ is prime returns YES otherwise returns NO

if ($n$ is of the form $a^b$, $b > 1$) return NO;
$r = 2$;
while ($r < n$) {
   if ($\gcd(n, r) \neq 1$) then return NO;
   if ($r$ is prime) then {
      $q =$ largest prime factor of $r - 1$;
      if (($q \geq 4\sqrt{r}\log(n)$) and ($n^{(r-1)/q} \not\equiv 1 \mod r$)) then break;
   }
   $r = r + 1$;
}
for ($a = 1$) to ($\lfloor 2\sqrt{r}\log(n) \rfloor + 1$) {
   if (($(x - a)^n \not\equiv (x^n - a) \mod (x^r - 1, n)$) then return NO;
}
return YES;
```

Correctness

Halting: follows from the existence of an AKS prime r for n.

Correctness: follows from AKS Identity.

Correctness

Outline of the proof of AKS Identity:

- n prime $\Rightarrow (x-a)^n \equiv (x^n - a) \mod n$ (by Identity)
  $\Rightarrow (x-a)^n \equiv (x^n - a) \mod (x^r - 1, n)$
- n composite: (by contradiction)
  - assume that n is not a power of a prime
    $\Rightarrow |\{n^i p^j : 0 \le i, j \le \lfloor \sqrt{r} \rfloor\}| > r$, for some p (Lemma)
    $\Rightarrow \exists_{(i,j) \neq (i',j')} : n^i p^j \equiv n^{i'} p^{j'} \mod r$ (by the pigeon hole principle)
  - assume that $(x-a)^n \equiv (x^n - a) \mod (x^r - 1, n)$
    $\Rightarrow (x-a)^{p^u n^v} \equiv (x^{p^u n^v} - a) \mod (x^r - 1, n) \; \forall_{u,v \ge 0}$ (Lemma)
    $\Rightarrow (x-a)^{p^i n^j} \equiv (x^{p^i n^j} - a) \mod (x^r - 1, n)$ (with $(u,v) = (i,j)$)
    $(x-a)^{p^{i'} n^{j'}} \equiv (x^{p^{i'} n^{j'}} - a) \mod (x^r - 1, n)$ (with $(u,v) = (i',j')$)
    $\Rightarrow (x-a)^{p^i n^j} \equiv (x-a)^{p^{i'} n^{j'}} \mod (x^r - 1, n)$ (by $n^i p^j \equiv n^{i'} p^{j'} \mod r$)
    $\Rightarrow p^i n^j = p^{i'} n^{j'}$ (Lemma)
    $\Rightarrow (i,j) = (i',j')$ (Lemma)

Complexity

Testing if n is a perfect power: $\tilde{O}(\log^4(n))$

Finding r (while loop) with $O(\log^6(n))$ iterations:

- Computing $\gcd(n, r)$ (Euclid): $O(\log^3(n))$
- Testing if r is prime (trial division): $O(\sqrt{r}\log^2(r))$
- Computing largest prime factor of $r - 1$: $O(\sqrt{r}\log^2(r))$
- Computing $n^{(r-1)/q} \mod r$: $O(\log^2(n) + \log^3(r))$
- Total: $\tilde{O}(\log^9(n))$

Complexity

AKS condition (for loop) with $O(\sqrt{r}\log(n))$ iterations:

- Computing $(x-a)^n \mod (x^r-1)$ (FFT): $\tilde{O}(r\log^2(n))$
- Computing $(x^n-a) \mod (x^r-1)$: $O(\log^2(n))$
- Total: $\tilde{O}(\log^{12}(n))$

Overall complexity of AKS algorithm: $\tilde{O}(\log^{12}(n))$

Complexity

Computing $\gcd(n, r)$ (Euclid): $O(\log^3(n))$

```
Input:  integers n,r
Output:  gcd(n,r)
 1. if r == 0
 2.    then return n;
 3.    else return gcd(r, n mod r);
```

Lamé's Theorem: The number of recursive calls of Euclid's algorithm is $O(\log(n))$.

Complexity

Testing if r is prime (trial division): $O(\sqrt{r}\log^2(r))$

```
Input:  integer r with r ≥ 2
Output:  YES if r is prime and NO otherwise
 1. t = 2; s = 4;
 2. while (s ≤ r) {
 3.    if (r mod t == 0)
 4.       then return NO;
 5.       else t = t + 1; s = s + 2t − 1;
 6. }
 7. return YES;
```

Complexity

Computing largest prime factor of $r-1$: $O(\sqrt{r}\log^2(r))$

```
Input:  integer r with r ≥ 2
Output:  the largest prime factor of r
 1. p = 1; y = 2; x = r;
 2. while ((x ≠ 1) and (y^2 ≤ r)) {
 3.    while (x mod y == 0) {
 4.       x = x/y; p = y;
 5.    }
 6.    y = y + 1;
 7. }
 8. if (x == 1) then return p else return x;
```

Complexity

Computing $n^{(r-1)/q} \mod r$: $O(\log^2(n) + \log^3(r))$

Computing $a^b \mod r$ (repeated squaring): $O(\log^3(n))$ with $a, b, r \leq n$

```
Input:  integers a,b,r
Output:  a^b mod r
 1. x = a mod r; y = b; z = 1;
 2. while (y ≠ 0) {
 3.    if (y is even)
 4.       then y = y/2; x = x^2 mod r;
 5.       else y = y - 1; z = zx mod r;
 6. }
 7. return z;
```

Complexity

Computing $(x-a)^n \mod (x^r-1)$ (FFT): $\tilde{O}(r\log^2(n))$

```
Input:  integers n,r,a with 2 ≤ r < n and 1 ≤ a < n
Output:  all coefficients of the polynomial (x − a)^n mod (x^r − 1, n)
 1. f(x) = 1; g(x) = x − a; y = n;
 2. while (y ≠ 0) {
 3.    if (y is even)
 4.       then y = y/2; h(x) = g(x)g(x); g(x) = h(x) mod (x^r − 1, n);
 5.       else y = y − 1; h(x) = f(x)g(x); f(x) = h(x) mod (x^r − 1, n);
 6. }
 7. return f(x);
```

Complexity

Computing $(x^n - a) \mod (x^r - 1)$: $O(\log^2(n))$

Computing $(x^n - a) \bmod (x^r - 1)$ is equivalent to computing $n \bmod r$:

$$
\begin{aligned}
(x^n - a) &\equiv (x^{cr+n \mod r} - a) \mod (x^r - 1) \\
&\equiv (x^{cr}x^{n \mod r} - a) \mod (x^r - 1) \\
&\equiv (x^{n \mod r} - a) \mod (x^r - 1) \qquad (x^r \equiv 1 \mod (x^r - 1))
\end{aligned}
$$

Conclusion

- Still inefficient for practical uses;
- Improvements on the complexity made by Bernstein;
- Possible conjecture holding implies improvement to $\tilde{O}(\log^3(n))$:

  Conjecture. *If* $n \not\equiv 0 \mod r$ *and if*

  $$(x-1)^n \equiv (x^n - 1) \mod (x^r - 1, n),$$

  either n *is prime or* $n^2 \equiv 1 \mod r$*.*

  Primality test with Conjecture:

  - find $r \in O(log(n))$ such that $n \not\equiv 0 \mod r$ and $n^2 \not\equiv 1 \mod r$;
  - if $((x-1)^n \equiv (x^n - 1) \mod (x^r - 1, n))$
    then n is prime else n is composite.