효율적 소수성 검정 알고리즘들에 대한 비교 • 분석

이호정, 송정환

한양대학교

A study on effective primality test algorithms

Ho Jung Lee, Jung Hwan Song

Hanyang Univ.

요 약

본 논문에서는 현재 사용되고 있는 소수성 검정 알고리즘의 효율성을 비교하여 효과적인 알고리즘 사용에 관한 방향을 제시하려 한다. 현재 가장 일반적으로 사용하고 있는 Miller-Rabin 소수성 검정법(Miller-Rabin primality test)에 대하여, Miller-Rabin 소수성 검정법 이외에 다른 확률적 소수성 검정법으로 제안된 Frobenius-Grantham 소수성 검정법(Frobenius-Grantham primality test)이 있다. 그러나 합성수 판별에 대한 확률적 우세함에도 불구하고, Miller-Rabin 소수성 검정법을 대체하고 있지 못하는 이유는 시간복잡도(time complexity)가 Randomized polynomial time이기 때문에 같은 확률에 대한 평균 실행 속도가 Miller-Rabin 소수성 검정법보다 크게 효율적이지 못하기 때문이다. 또한, 2002년 Manindra Agrawal이 제시한 AKS 알고리즘(AKS algorithm)은 최초의 다항식 시간내 결정적 소수성 검정법(Polynomial time deterministic primality test)이지만, 시간 복잡도에서 다항식의 차수가 높기 때문에 현재 사용되고 있는 확률적 소수성 검정법(Probabilistic primality test)을 대체하지 못할 것으로 사료된다. 본 논문에서는 최근 발표된 소수성 검정법인 Frobenius-Grantham 소수성 검정법, AKS 알고리즘과 기존의 Miller-Rabin 소수성 검정법의 장단점을 비교 • 분석해 보고자 한다.

I. 서론

공개키 암호시스템의 안전성은 암호 알고리즘의 일방향성과 비밀키의 안전한 보관 및 사용에 의존한다. 암호 알고리즘 완성 이후 공개키 파라메터의 효율적 생성은 공개키 암호시스템에서 없어서는 안 될 요소라 하겠다. 키파라메터를 생성하여 제대로 된 소수인지를 검정하는데 사용되는 것이 소수성 검정 알고리즘(Primality Test Algorithm)이다. 따라서 신뢰할 수 있고 효율적인 소수성 검정 알고리즘의 사용이야말로 공개키 암호를 사용하는 보안제품의 안전성을 보장하기 위한 시작이라고 할 수 있다.

주어진 수에 대한 소수성 검정은 확률적 소수성 검정법(Probabilistic Primality Test)과 결정적 소수판별법(Deterministic Primality Test)으로 나눌 수 있다. 현재 암호학에서 사용되는 소수성 검정법은 그 효율성 때문에 확률적 소수성 검정법을 사용하고 있는 실정이다. 1980년 Rabin. M. O에 의해 정리된 Miller-Rabin 소수성 검정법[1]과 1977년 발표된 Solovay-Strassen 소수성 검정법[4]은 현재 가장 널리 사용되는 확률적 소수 검정법이다. 이러한 확률적 소수성 검정법을 통해 검정된 유사소수(Pseudoprime)는 낮은 오차 정도를 가지기 때문에 암호시스템에서의 사용 시 안전성에는 전혀 문제를 가져오지 않는다. 기존의 Miller-Rabin 소수성 검정 알고리즘 이후, 보다 효율적이고 높은 확률을 가지는 소수성 검정 알고리즘을 찾아 왔으며, 최근 Frobenius-Grantham 확률적 소수 검정 알고리즘[2]과 2002년 최초의 다차시간 복잡도를 가지는 결정적 소수성 검정 알고리즘으로 최대 화제를 불러온 AKS 결정적 소수 판별법[3]이 가장 주목 받는 알고리즘이다.

본 논문에서는 기존의 Miller-Rabin 소수성 검정법, Frobenius-Grantham 소수성 검정법과 AKS 소수판별법을 비교하여 봄으로써 암호학적으로 기존 소수성 검정 알고리즘의 대체 가능성 여부를 확인해 보고 공개키 암호시스템의 키파라메터의

소수성 검정 시 가장 적합한 소수성 검정 알고리즘을 제시하려한다.

II. 소수성 검정의 개념

주어진 크기의 수 n이 소수임을 확인하는 가장 일반적인 방법은 $\sqrt{n}$보다 작은 모든 소수로 나누어, 나눌 수 있다면 합성수이고 나눌 수 없다면 소수임을 확인하는 것이다. 이것을 Trial Division이라 하는데 일반적으로 n이 1000일 경우까지는 Trial Division으로 소수성을 판별하고, 그 이상일 경우 알려져 있는 다른 소수판별법을 사용하는 것이 가장 효과적인 방법으로 알려져 있다.

위의 Trial Division의 높은 시간복잡도 때문에 보다 효율적인 다른 알고리즘의 연구가 필요하며, 소수성 검정 알고리즘에 주어진 수 n을 입력했을 때, 다음의 두 가지 결과를 출력할 수 있다.

- 소수임을 증명 가능한 알고리즘 사용
  → 증명 가능한 소수(Provable prime)
- 소수성 증명 불가능한 알고리즘 사용
  → 확률적 소수(Probable prime)

소수성 증명 불가능한 알고리즘을 사용한 소수성 검정은 확률적 소수성 검정법(Probabilistic primality test)라 하며, 주어진 입력 n이 합성수임을 명확히 판별할 수 있다. 그러나 확률적 소수판별법은 "확률적 소수"라는 출력 시 그 소수성에 대한 수학적인 증명은 가능하지 않다. 이러한 이유로 간혹 확률적 소수성 검정은 합성 검정(Compositeness test)이라는 이름으로 사용되기도 한다.

소수성 증명 가능한 알고리즘을 사용한 소수성 판별은 주어진 입력 n이 소수이거나 합성수임을 명확히 출력할 수 있으며, 이러한 출력에 대한 이유가 수학적으로 증명 가능해야 한다. 이러한 소수판별법은 결정적 소수성 검정법(Deterministic primality test)라 하며, 이에 대한 여러 알고리즘이 존재하지만 시간 복잡도가 매우 높고 계산에 소요되는 자원의 크기가 너무 크기 때문에 암호학에 대한 응용은 원활히 이루어지지 못하고 있는 실정이다.

III. 확률적 소수성 검정 알고리즘

확률적 소수성 검정은 한 번 실행함으로도 합성수 판별을 명확히 판단할 수 있다. 또한, 독립적으로 실행 시 마다 "소수" 임을 출력한다면 그에 따라 소수성에 대한 신뢰도가 높아지게 된다. 따라서 확률적 소수성 검정은 필요한 신뢰도까지의 반복적인 실행으로 원하는 신뢰성을 가지는 소수를 선택할 수 있는 것이다. 이러한 반복 실행 시 누적되는 오차의 확률은 각각의 실행마다 독립적이고, 또한 한 번 실행 시 오차 확률의 배가 된다. 만약 합성수에 대해 독립적으로 t번 확률적 소수성 검정을 시행한다면, 주어진 수 n이 소수임을 증명할 확률은 최대 $\left(\frac{1}{2}\right)^t$가 된다.[5]

확률적 소수성 검정을 이용하는 가장 일반적인 알고리즘은 Miller-Rabin 소수성 검정법과 Solovay-Strassen[5] 소수성 검정법이다. 그러나, Miller-Rabin 소수성 검정이 Solovay-Strassen 소수성 검정보다 실행시간도 효율적이고 오차확률도 작기 때문에 Miller-Rabin 소수성 검정법이 대중적으로 사용되고 있다.

1. Miller-Rabin 소수성 검정

확률적 소수성 검정 알고리즘 중 가장 일반적으로 사용하고 있는 강한 유사소수 검정법으로 알려져 있다. 1976년 Miller. G[4]에 의해 제안되었으며, 1980년 Rabin. M. O.[16]에 의해 정리된 확률적 소수성 검정 알고리즘이다. 현재 RSA를 비롯한 수많은 암호시스템에서 사용하고 있으며, 높은 신뢰도와 무엇보다도 적은 계산량으로 확률적 소수성 검정의 대표로 꼽히는 알고리즘이다. Miller-Rabin 소수성 검정은 다음의 정리 1.[1]을 기반으로 한다.

정리 1. n을 홀수라 하고 홀수인 r에 대하여 $n-1 = 2^s r$이라 한다. a는 $gcd(a, n) = 1$을 만족하는 정수라 한다. 그러면 $0 \le j \le s-1$인 어떤 정수 j에 대하여 다음 중 하나를 만족한다.

$$a^r \equiv 1 \ (mod n)$$
$$a^{2^j r} \equiv -1 \ (mod n)$$

위의 정리를 바탕으로 다음의 Miller-Rabin 소수성 검정 알고리즘을 생각할 수 있다.

표 1 : Miller-Rabin 소수성 검정 알고리즘

알고리즘 : $Miller-Rabin(n,t)$

입력 : 홀수 $n \geq 3$과 반복회수 $t \geq 1$

출력 : "소수", "합성수"

홀수인 r에 대하여 $n-1 = 2^s r$이라 하고, 다음을 t회 반복적으로 수행한다.

① $2 \leq a \leq n-2$인 난수 a를 생성.
② $y = a^r (mod n)$ 계산.
③ 만약 $y \neq 1$, $y \neq n-1$이면 다음을 수행한다.
  a. $i = 1$
  b. $i < t$ 이고 $y \neq n-1$이면 다음을 수행한다.
    I. $y = y^2 (mod n)$
    II. $y = 1$이면 "합성수" 출력 후 중지
    III. $i = i+1$
  c. $y \neq n-1$이면 "합성수" 출력 후 중지
④ "소수" 출력

위의 알고리즘 $Miller-Rabin(n,t)$는 각각의 a가 정의 3. 2.의 ①을 만족하는지를 확인하는 알고리즘이다. 위의 알고리즘 중 ③-b.-II.의 경우 만약 $y = 1$이면 $a^{2^i r} \equiv 1 (mod n)$이다. 이러한 경우 $a^{2^{i-1} r} \neq \pm 1 (mod n)$ 이고, $gcd(a^{2^{i-1} r} - 1, n)$은 n의 자명한 인수가 될 수 없기 때문에, n은 합성수가 된다. 만약 n이 소수라면 정리 3. 3.에 위배되지 않기 때문에, n이 합성수라면 알고리즘 $Miller-Rabin(n,t)$은 확실하게 "합성수"를 출력하게 된다. 마찬가지로 만약 n이 소수라면 알고리즘 $Miller-Rabin(n,t)$은 언제나 "소수"를 출력한다.

결국 반복적으로 수행되는 $Miller-Rabin(n,t)$은 t회까지의 모든 반복에서 "소수"라는 출력을 얻어야만 소수성을 검정한 것으로 한다. 또한 실험적으로 합성수인 입력 n이 t회 $Miller-Rabin(n,t)$ 수행할 경우 소수로 판별될 확률은 $\left(\frac{1}{4}\right)^t$보다 작다.

2. Frobenius-Grantham 소수성검정

확률적 소수성 검정 알고리즘 중 비교적 높은 신뢰도를 갖는 알고리즘이다. 1998년 Jon Grantham에 의하여 완성된 알고리즘으로써, 유한체 내에서 2차 다항식을 이용하여 대부분의 pseudoprimes, Euler pseudoprimes, strong pseudoprimes 등의 특수한 형태의 소수를 일반화하여 검정할 수 있는 알고리즘이다.[2]

Frobenius-Grantham 소수성 검정 알고리즘은 다음 표 2.와 같이 구현할 수 있다.

표 2 : Frobenius-Grantham 소수성 검정 알고리즘

알고리즘 : $Frobenius-Grantham(n,t)$

입력 : 홀수 $n \geq 1$과 반복회수 $t \geq 1$

출력 : "소수", "합성수"

① 입력받은 정수 n이 제곱수인지 검정하고 제곱수이면 "합성수" 출력 후 검정 중단한다.

$2's = n^2 - 1$을 만족하는 각각 다른 난수인 정수 r, s를 생성하여 다음을 t회 반복적으로 시행한다. 반복 시행하는 경우 매번 새로운 r, s를 생성하여 사용한다.

② 다음 중 하나를 만족하는 $b, c \in Z_n$인 b, c를 선택한다.
  a. $gcd(b^2 + 4c, n) \neq 1, n$ 혹은 $gcd(c, n) \neq 1, n$이면 "합성수" 출력 후 검정 중단.
  b. $Jacobisymbol\left(\frac{b^2 + 4c}{n}\right) = -1$, $\left(\frac{-c}{n}\right) = 1$
  이면 ③으로 진행.

③ 다항식 $x^{\frac{n+1}{2}} \bmod (n, x^2 - bx - c)$의 차수가 0이면 ④로 진행. 그렇지 않으면 "합성수" 출력 후 검정 중단.

④ $x^s \neq 1 \bmod (n, x^2 - bx - c)$이면 ⑤로 진행. 그렇지 않으면 "합성수" 출력 후 검정중단.

⑤ $x^s = 1 \bmod (n, x^2 - bx - c)$와 $x^{2^j s} = -1 \bmod (n, x^2 - bx - c)$을 만족하는 $0 \leq j \leq r-2$인 j가 존재하면 "소수" 출력. 그렇지 않으면 "합성수" 출력 후 검정 중단.

*Jacobisymbol*의 연산은 주어진 정수의 소인수분해 이상의 시간복잡도를 가지기 때문에, 실제 구현 검정할 경우 가장 많은 시간이 소요되는 부분이다. Miller-Rabin 소수성 검정과 같이 대중적으로 사용되고 있는 Solovay-Strassen 소수성 검정[16]의 경우도 *Jacobisymbol*의 연산이 사용되기 때문에 그 효율성과 구현 용이성이 Miller-Rabin 소수성 검정보다 현저히 떨어지기 때문에 Miller-Rabin 소수성 검정을 대체하고 있지 못하고 있다.

Frobenius-Grantham 소수성 검정의 경우 입력된 합성수 n이 검정을 통과하여 소수로 판별될 확률은 $\frac{1}{7710}$으로써, 1회 검정시 Miller-Rabin 소수성 검정보다 높은 신뢰도를 갖게 된다. 실험적으로 1회 검정시 Frobenius-Grantham 소수성 검정은 Miller-Rabin 소수성 검정의 3배의 시간이 걸리기 때문에 이론적으로 Miller-Rabin 소수성 검정보다 효율적인 알고리즘으로 볼 수 있으며, 구현에 대한 검정은 뒤에서 다루기로 한다.

IV. 결정적 소수 검정

확률적 소수성 검정 알고리즘과는 달리 수학적 증명일 이용하여 주어진 정수가 소수임을 검정하는 방법이다. 알고리즘의 출력 결과는 수학적으로 완전히 신뢰할 수 있으며, 일반적으로 결정적 소수 검정 알고리즘은 확률적 소수성 검정 알고리즘보다 월등히 높은 시간 복잡도를 갖게 된다. 결정적 소수 검정 알고리즘은 보통 다음과 같은 것들이 있으며, 일반적으로 NP-class[6]의 복잡도를 갖는다.

Elliptic Curve Primality Test[7]

APR Primality Test[8]

Maurer's Algorithm[9]

Sieve of Eratosthenes

Brute force incremental search

위의 예 말고도 다른 결정적 소수 검정이 있으며, 에라토스테네스의 체(Sieve of Eratosthenes)와 brute forceforce incremental search의 경우는 너무도 높은 계산 복잡도 때문에 암호시스템에서의 사용 가능성은 없는 검정이다. 타원곡선을 이용한 소수 검정(Elliptic Curve Primality Test)은 $O(n^6)$의 시간복잡도를 가지며, 결정적 소수 검정 알고리즘으로 알려져 있다. 그러나 타원곡선 소수성 검정은 일반적으로 맞는 알고리즘으로 알려져 있으나, 알고리즘의 일부분은 증명되지 않은 채로 사용하고 있는 부분이 존재한다. 또한 APR 소수 검정(APR Primality Test)도 $O((log_2 n)^{log_2 log_2 log_2 n})$의 시간 복잡도를 가짐으로써 실제 암호시스템에서는 용이하게 사용할 수 없는 것으로 사료된다.

AKS는 최초의 결정적 다항식 시간 내(deterministic polynomial time) 소수성 검정 알고리즘으로써 암호시스템에서의 사용가능성을 타진해볼 가치가 충분할 것으로 보인다. 따라서 본 논문에서는 AKS 소수 검정 알고리즘의 구현 검정을 통하여 암호시스템에서의 사용가능성 여부를 확인해 보고자 한다.

1. AKS 소수 검정

AKS 소수 검정 알고리즘은 페르마 정리(Fermat's little theorem)를 기본으로 하는 알고리즘으로써 2002년 8월 인도 Kanpur에 있는 Indian Institute of Technology의 Manindra Agrawal과 Neeraj Kayal, Nitin Saxena가 제안한 알고리즘이다. 이 알고리즘은 최초의 다항식 시간 복잡도를 가지는 결정적 소수 검정 알고리즘으로, 응용시스템에서 사용의 귀추가 주목되고 있다.

정리 2. 만약 p가 소수이고, a와 p가 서로소이라면 소수 r에 대하여 다음을 만족한다.

$$(x-a)^p \equiv (x^p - a)\,(\mathrm{mod}\, x^r - 1, p)$$

따라서 주어진 정수 n이 제곱수 형태인지를 확인한 후 위의 정리 2.에 따라 소수임을 검정하게 되고, 그것이 바로 AKS 소수 검정 알고리즘이 된다. 위의 정리에서 주어진 식을 만족하는 소수 r에 대한 존재성은 R.C.Baker와 E.Foubry에 의하여 증명되었다[10][11]. 이러한 소수 r을 유용한 소수(Useful prime)라 하며 존재성의 증명에 따르는 정리 3.에서 AKS 소수 검정의 구체적인 알고리즘을 생각할 수 있다.

정리 3.

만약 $(x-a)^p \equiv (x^p-a)\,(\mathrm{mod}\,p, x^r-1)$ 에서 소수 r이 존재하고 q가 $r-1$의 최대 소인수이면 다음을 생각할 수 있다.

① $q \geq 4\sqrt{r}\log_2 n\ (n>1)$

② $n^{\frac{r-1}{q}} \neq 1 \bmod r$

③ $q \mid o_r(n)$

④ 위의 ①과 ②를 만족 ↔ ①과 ③을 만족

위의 정리들을 이용하여 아래의 AKS 소수 검정 알고리즘을 구성할 수 있다.

표 3 : AKS 소수 검정 알고리즘

알고리즘 이름 : $AKS(n)$

입력 : 홀수 $n \geq 1$

출력 : "소수", "합성수"

① 만약 n이 $a^b\,(b>1)$의 형태라면, "합성수" 출력 후 검정 중단.

② $r=2$로 초기화

③ $r<n$인 동안 아래를 실행한다.

a. 만약 $gcd(n,r) \neq 1$이면, "합성수" 출력 후 검정 중단.

b. 만약 r이 소수이면, 아래를 실행한다.

1) q는 $r-1$의 최대 소수인 인수로 정의

2) 만약 $q \geq 4\sqrt{r}\log n$이고 $n^{\frac{r-1}{q}} \neq 1 \bmod r$이면, ③을 빠져 나와 ④를 실행한다.

3) $r=r+1$

④ a를 1에서 $2\sqrt{r}\log n$까지 다음을 실행한다.

만약 $(x-a)^n \neq (x^n-a)\,(\mathrm{mod}\,x^r-1, n)$ 이면, "합성수" 출력 후 검정 중단.

⑤ "소수" 출력

위의 AKS 소수 검정 알고리즘 $AKS(n)$은 ①에서 $(x-a)^n \equiv (x^n-a)\,(\mathrm{mod}\,x^r-1, n)$을 통과할 수 있는 n의 형태인 제곱수를 검정하게 된다. ③에서 유용한 소수 r을 생성 후, ④에서 정리 2.를 검정하면 결정적 소수 검정 알고리즘이 되는 것이다.

①의 제곱수 검정은 1998년 발표된 Daniel J. Bernstein의 "Perfect power detecting"[12] 알고리즘을 이용한다. "Perfect power detecting" 알고리즘은 $O((\log_2 n)^3)$의 복잡도를 가지는 알고리즘이다. 또한 ③-b.-1)에서 소인수를 찾기 위하여 "brute force incremental search"를 이용하는데, "brute force incremental search"가 비록 NP-class의 알고리즘이지만 r의 크기가 작기 때문에 충분히 주어진 시간에 실행될 수 있다. 전체 AKS 소수 검정 알고리즘의 복잡도는 $O((\log_2 n)^{12})$으로 나타나고 있으며, 다항식 시간 복잡도를 가짐을 알 수 있다.

V. 소수성 검정 알고리즘의 효율성 구현 비교

1. 확률적 소수성 검정의 오류

확률적 소수성 검정 알고리즘은 여러 번 반복 실행하여 합성수가 소수로 판별될 확률을 낮추는 검정법이다. 때문에 확률적 소수성 검정을 할 경우 반복회수 t를 적당하게 결정해야만 신뢰성 있고 효율적인 소수성 검정을 할 수가 있다.

현재 확률적 소수성 검정의 경우 검정할 수의 크기에 따라 아래 표 4와 같이 정해진 오류의 한계를 권고하고 있다.

표 4 : 확률적 소수성 검정의 오류 한계 권고안

수의 크기 (bit)	오류 한계
128	2^{-100}
256	2^{-100}
512	2^{-101}
1024	2^{-143}
2048	2^{-202}

Miller-Rabin 소수성 검정의 최대 오류 확률은 $\frac{1}{4}$이고, Frobenius-Grantham 소수성 검정의 최대 오류 확률은 $\frac{1}{7710}$이기 때문에 다음과 같이 오류 한계에 따르는 각 확률적 소수성 검정 알고리즘의 실행 회수를 제시할 수 있다.

표 5 : 오류 한계에 따르는 확률적 소수성 검정의 실행 회수

MR : Miller-Rabin 소수성 검정 실행회수

FG : Frobenius-Grantham 소수성 검정 실행회수

수의 크가(bit)	MR	FG
128	50	8
256	50	8
512	51	8
1024	72	11
2048	101	16

위의 표 5는 오류 한계를 만족하는 Miller-Rabin 소수성 검정과 Frobenius-Grantham 소수성 검정의 실행 회수 t를 제시한 것이다. 실행 회수 t는 각 알고리즘의 1회 실행시 최대 오류에 따라 결정된 것으로 알고리즘의 최대 실행 회수를 의미한다. 그러나 알고리즘의 오류 확률이 언제나 최대 오류를 가지지는 않을 것이며 실제 평균 오류 한계는 표 5의 최대 오류 한계보다 작아질 것을 알 수 있다. 다만 평균 오류 한계에 의한 실행 회수나 최대 오류 한계에 의한 실행 회수가 크게 차이가 나지 않기 때문에 최대 오류 한계에 따르는 실행 회수만을 생각하기로 한다.

2. 알고리즘의 구현

본 논문에서 효율적 소수성 검정 알고리즘을 비교하기 위하여 각 알고리즘을 구현하였다. 구현된 알고리즘은

Miller-Rabin 소수성 검정

Frobenius-Grantham 소수성 검정

AKS 알고리즘

이고, 아래의 스펙에서 구현되었다.

표 6 : 알고리즘 구현 스펙

CPU	Pentium4 2.4Ghz Xeon
RAM	4 Gbyte
OS	Window XP
Language	Microsoft Visual C++ 6.0
HDD	7200 rpm

현재 32 bit 시스템의 컴퓨터에서는 한 번에 계산할 수 있는 최대 정수의 크기가 부호 bit을 무시한다 하더라도 2^{32} 밖에 되지 않는다. 따라서 최대 2048 bit 이상의 수에 대한 연산을 하기 위하여 Multi Precision Library를 사용하였다. 최근 많이 사용되고 있는 Multi Precision Library로는 GNU 프로젝트의 GMP, Victor Shoup의 NTL과 Shamus software의 M.I.R.A.C.L이 있다. 본 논문에서의 구현은 M.I.R.A.C.L을 이용하여 했다. 또한 *Jacobisymbol*의 계산 등에서 자체적인 연산을 수행하였으나 Library 자체의 연산과 크게 다르지 않은 것으로 사료된다. M.I.R.A.C.L은 곱셈의 Fast Fourier Transform의 사용과 Fast Polynomial Factorizing 등의 방법을 사용함으로써 좋은 효율성을 보여주는 Library이다.

3. 각 알고리즘의 효율성 구현 비교

Frobenius-Grantham 소수성 검정은 Miller-Rabin 소수성 검정보다 평균 3배의 시간복잡도를 가지고 있다. 따라서 표 5에 따르면 Frobenius-Grantham 소수성 검정이 Miller-Rabin 소수성 검정에 비하여 아래의 표 7과 같은 이론적 효율성을 가지게 된다.

표 7 : MR에 대한 FR의 효율성

MR : Miller-Rabin 소수성 검정 실행회수

FG : Frobenius-Grantham 소수성 검정 실행회수

수의 크기 (bit)	MR	FG	FG의 효율성
128	50	8	108 %
256	50	8	108 %
512	51	8	108 %
1024	72	11	118 %
2048	101	16	110 %

그러나 Solovay-Strassen 소수성 검정[4]과 마찬가지로 Frobenius-Grantham 소수성 검정[2]도 *Jacobisymbol*의 연산을 포함하고 있기 때문에 구현 검정에서는 위와 다른 결과를 보이고 있었다. 또한 구현에는 Multi Precision Library의 사용에 따르는 연산의 과부하가 있기 때문에 수행시간에서는 위와는 전혀 다른 차이를 보이고 있다.

AKS 소수 검정 알고리즘은 다항식 시간 알고리즘이기는 하지만 5자리 정수(16 bit)에 대한 검정만으로도 많은 실행 시간을 요구하고 있다.

구현 후 검정에 따르는 수행 시간은 아래의 표 8에서 확인할 수 있다. 표 8의 확률적 소수성 검정 알고리즘의 수행 시간은 오류 한계에 따르는 회수만큼 실행한 시간이다. 또한 각각의 실행 시간은 수 회 실험 후 평균 실행 시간을 택한 것이다.

표 8 : 소수성 검정 알고리즘의 실행 시간 비교
MR : Miller-Rabin 소수성 검정 실행시간
FG : Frobenius-Grantham 소수성 검정 실행시간
AKS : AKS 소수 검정 실행시간
* MR과 FG의 실행 시간의 단위는 초이다.
* AKS의 실행 시간의 단위는 년이다.

수의 크기 (bit)	MR	FR	AKS
16	-	-	(12 일) 0.032 년
128	253	1022	3610 년
256	792	3252	1850938563 년
512	1639	8593	-
1024	3048	24301	-
2048	7125	51524	-

표 8의 결과에서 AKS 소수 검정 알고리즘의 실행 시간은 5자리 정수(16 bit)에 대한 검정 시간이 12일인 것으로부터 추정한 수치이다. 아래의 그림 1은 위의 표 8을 그래프화한 것으로 AKS 소수 검정 알고리즘은 비교의 대상이 될 수 없어 포함시키지 않았다.

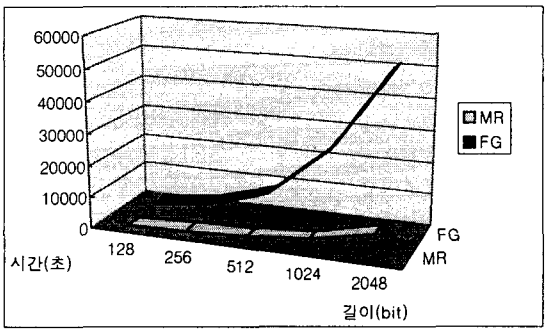

그림 1 : 확률적 소수성 검정 알고리즘의 속도 비교

VI. 결론

본 논문에서는 대표적으로 사용되고 있는 "Miller-Rabin 소수성 검정"과 "Frobenius-Grantham 소수성 검정", 그리고 최초의 결정적 다항식 시간 소수 검정 알고리즘인 "AKS 소수 검정"에 대하여 효율성을 비교해 보았다. 표 8에서 알 수 있듯이 같은 오류 한계에 대한 확률적 소수성 검정의 실행 시간은 이론적인 수치와는 상이하게 다른 것으로 확인되었다. 이유는 구현에서 사용된 Multi Precision Library의 연산들의 작용에 의하여 실제 복잡도와 다른 계산 복잡도를 가지기 때문인 것으로 사료된다. 또한 AKS 소수 검정 알고리즘은 다항식 시간 알고리즘이기는 하지만 큰 지수의 복잡도를 가지기 때문에 암호시스템에서의 사용에는 무리가 있을 것으로 보인다.

AKS 소수 검정의 논문에서 주장하듯이[3] 알고리즘의 복잡도를 $O((\log_2 n)^6)$이 될 수 있고 혹은 Daniel J, Bernstein이 밝혔듯이[17] $O((\log_2 n)^4)$까지 복잡도를 낮출 수 있다 하더라도 일반적으로 사용되는 크기의 수에 대한 사용은 어려울 것으로 사료된다. 결국 현재 일반적으로 사용되는 키 길이에 대한 소수성 검정에는 Miller-Rabin 소수성 검정을 대체할 정도의 알고리즘은 없는 것으로 생각할 수 있다.

그러나 항상 64 bit 이상의 크기를 가지는 소수만을 암호시스템에서 사용되는 것은 아니므로, 작은 크기의 수에 대한 소수성 검정은 결정적 소수 검정 알고리즘을 적용할 수 있을 것이다.

현재는 확률적 소수성 검정을 통한 검정이 그 높은 효율성 때문에 일반적인 소수성 검정 알고리

즘을 대표하고 있다. 그러나 AKS 소수 검정 알고리즘이나 Certificate을 이용한 결정적 소수 검정법 등 기존의 결정적 소수 검정 알고리즘보다 현실적인 방법이 계속 대두되고 있다. 아무리 낮은 오류 확률을 갖는다 하더라도 확률적으로 소수를 검정하는 것과 결정적으로 소수임을 확인하는 것은 확연히 다른 것이기 때문에, 32bit 전후의 작은 크기의 수에 대한 소수성 검정에 대한 결정적 소수 검정은 필요다. 만약 데이터베이스에 저장되어 있는 소수들을 사용한다 하더라도 작은 소수에 대한 신뢰성을 확보하기 위하여 결정적 소수 검정을 행해야 한다.

현재 확률적 소수성 검정에 대한 연구는 정체되어 있고, 결정적 소수 검정에 대한 연구가 주를 이루고 있는 추세이다. 이는 검정을 위한 시스템의 고사양화와 알고리즘 내 연산의 효율성 증대로부터 오는 자연스런 현상이다. 따라서 큰 수에 대한 확률적 소수성 검정과 작은 수에 대한 결정적 소수 검정의 사용을 적절하게 조화시키는 것이 효과적일 것으로 사료된다.

참고문헌

[1] M.O.Rabin, "Probabilistic algorithm for testing primality", J. Number Theory(12), 1980

[2] Jon Grantham, "A probable prime test with high confidence. J. Number Theory(72), 1998

[3] Manindra Agrawal, Neeraj Kayal and Nitin Saxena, "PRIMES is in P", Preprint, 2002

[4] R. Solovay and V. Strassen, "A fast Monte-Carlo test for primality", SIAM J. Comput. 6(1), 1977

[5] Alfred J. Menezes, Paul C. van Oorschot, Scott A. Vanstone, "Handbook of applied cryptography", CRC, 1997

[6] Michael sipser, "Introduction to the theory of computation", PWS publishing company, 1997

[7] A.O.L. Atkin, F. Morain, "Elliptic curves and primality proving", Mathematics of Computation 61, 1993

[8] Adleman. L. M, Pomerance. C, Rumely. R. S, "On distinguishing prime numbers from composit number", Ann. Math(117), 1983

[9] Ueli Maurer. "Fast generation of prime numbers and secure public key cryptographic parameters", J. Cryptology(8), 1995

[10] R.C.Baker, G.Harman, "The Brun-Titchmarsh theorem on average", In proceedings of a conference in honor of heini halberstam(1), 1996

[11] E.Fouvry, "Theorem de Brun-Titchmarsh: application au theoreme de fermet", 1996

[12] Daniel. J. Bernstein, "Detecting perfect powers in essentially linear time", Mathematics of computation(67), 1998

[13] GNU project, http://www.swox.com/gmp

[14] Victor shoup, http://www.shoup.net

[15] Shamus software, http://indigo.ie/~mscott

[16] Miller. G, "Riemann's hypothesis and tests for primality", J. Comp. Syst. Sci(13), 1976

[17] Daniel. J. Bernstein, "Probeing prmality after Agrawal-Kayal-Saxena", Draft, 2003