IEICE TRANS. INF. & SYST., VOL.E94–D, NO.6 JUNE 2011

PAPER

Probabilistic Analysis on the Optimal Combination of Trial Division and Probabilistic Primality Tests for Safe Prime Generation*

Heejin PARK†, *Member and* **Dong Kyue KIM**††a), *Nonmember*

SUMMARY A safe prime p is a prime such that $(p-1)/2$ is also a prime. A primality test or a safe primality test is normally a combination of trial division and a probabilistic primality test. Since the number of small odd primes used in the trial division affects the performance of the combination, researchers have studied how to obtain the optimal number of small odd primes to be used in the trial division and the expected running time of the combination for primality tests. However, in the case of safe primality tests, the analysis of the combination is more difficult, and thus no such results have been given. In this paper, we present the first probabilistic analysis on the expected running time and the optimal number of small odd primes to be used in the trial division for optimizing the tests. Experimental results show that our probabilistic analysis estimates the behavior of the safe primality tests very well.

key words: *safe prime, safe prime generation, trial division, miller-rabin test, cryptography, information security*

1. Introduction

A safe prime p is a prime such that $(p-1)/2$ is also a prime. Generating large primes is important in cryptography: Cryptographic algorithms such as RSA [1] cryptosystem and ElGamal [2] cryptosystem and signature schemes such as DSS [3] require generating large (512-bit or 1024-bit) primes. Safe primes are also important in cryptography and have been studied extensively. The traditional DH key agreement protocol [4] uses safe primes to protect itself from the subgroup attacks. Recently, many cryptosystems and signature schemes require safe primes to guarantee their securities: The secure coalition-resistant group signature suggested by Ateniese et al.'s [5] and Cramer and Shoup's signature scheme [6] require safe primes. Gennaro et al.'s hash-and-sign signature [7] and Gennaro et al.'s undeniable signature [8] also require safe primes. Besides, Shoup's threshold signature [9], Camenisch and Lysyanskaya's credential system [10], and Fujisaki and Okamoto's zero-knowledge protocol [11] require safe primes. Sharing and verifying safe primes also have been studied: Algesheimer et al. [12] and One and Kubiatowicz [13] studied generation of shared safe prime (products). Camenisch and Michels [14] presented a zero-knowledge proof that a number is the product of two safe primes. Ibrahim [15] suggested a verifiable threshold sharing of a safe prime. In addition, distributed computation of the RSA function [16]–[18] relies heavily on distributed primality tests of safe primes. Thus, safe primes have been used extensively and are important in cryptography.

Generating an n-bit prime is an iterative application of odd random number generation and primality test. Odd random numbers are mostly generated by *random search* or *incremental search* [19]. The random search is the traditional and basic method for prime generation. In each iteration, it generates an odd random number r and test its primality. Thus, it generates a prime with a uniform probability over the entire given interval. The incremental search sacrifices uniformity slightly for speedup. It generates an odd random number r only once, and test whether $r+2(i-1)$ is prime or not in the ith, $(i \geq 1)$ iteration. In addition, the incremental search has many variants which are optimized for various situations. Furthermore, both random search and any incremental search are much faster than the primality test.

Since the random number generation is much faster than the primality test, most of the running time of the prime generation is consumed by the primality test. There are several deterministic primality tests such as trial division [20], Pocklington's test [21], Maurer's algorithm [22] and AKS test [23] and probabilistic primality tests such as Solovay-Strassen test [24], Fermat test [20], and Miller-Rabin test [25], [26]. Generally, several primality tests are combined to speed up the primality test and the most widely used combination is the combination of the trial division and a probabilistic primality test such as Fermat test and Miller-Rabin test. The speed of the combination is affected by k, i.e., the number of primes used in the trial division. Maurer [22] analyzed the combination and presented formulas to estimate the expected running time of the combination and k_{opt}, the optimal number of small odd primes to be used in the trial division for optimizing the combination when odd random numbers are generated by random search. Later, for a variant of incremental search, Brandt et al. [27] studied optimization and expected running time of the combination. In the case of safe prime generation, the analysis is far more difficult so no analysis has been given even for

Manuscript received April 7, 2010.
Manuscript revised January 27, 2011.

†The author is with the Faculty of Department of Computer Science and Engineering, Hanyang University, Seoul 133–791, South Korea.
††The author is with the Faculty of Department of Electronic Engineering, Hanyang University, Seoul 133–791, South Korea.

*This work was supported by the research fund of Hanyang University (HY-2005-S), the National Research Foundation of Korea (NRF) grant funded by the Korea government (MEST) (2010-0025668), and Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2009-0090441) and (2010-0013441).

a) E-mail: dqkim@hanyang.ac.kr (corresponding author)

DOI: 10.1587/transinf.E94.D.1210

the random search, not to mention the incremental search.

In this paper, we present the first probabilistic analysis on the expected running time and k_{opt} for the safe primality test based on random search. The results are convenient to use in practice because the expected running time is given as a function of $\mathtt{div}_n$, $\mathtt{ppt}_n$, and k where $\mathtt{div}_n$ is the time required for dividing an n-bit number by a word-sized prime, $\mathtt{ppt}_n$ is the time required for performing a probabilistic primality test on an n-bit integer, and k is the number of small primes used in the trial division. Thus, once $\mathtt{div}_n$ and $\mathtt{ppt}_n$ are measured, one can easily estimate the expected running time and k_{opt}.

This paper is organized as follows. We show how to compute the expected running time in Sect. 2 and k_{opt} in Sect. 3. In Sect. 4, we present some experimental results and compare them with the expectation by our analysis. In Sect. 5, we conclude.

2. Expected Running Time

We first introduce the safe primality test based on random search [28], [29]. Let r denote a random odd integer and p_i's ($1 \le i \le k$) are small odd primes.

1. Trial division on r and $(r-1)/2$: Check if $r \not\equiv 0, 1 \pmod{p_i}$ for each prime $p_1 < p_2 < \cdots < p_k$. Note that checking if $r \not\equiv 1 \pmod{p_i}$ is the same as checking if $(r-1)/2 \not\equiv 0 \pmod{p_i}$.
2. Probabilistic primality test on r.
3. Probabilistic primality test on $(r-1)/2$.

For the safe primality test above, we present a probabilistic analysis on the expected running time (denoted by $S_n(k)$) on an odd n-bit integer r when k smallest odd primes are used in the trial division.

Theorem 1.
$$S_n(k) \approx \sum_{i=1}^{k}\prod_{j=1}^{i-1}\left(1-\frac{2}{p_j}\right)\cdot \mathtt{div}_n + \left(\prod_{i=1}^{k}\left(1-\frac{2}{p_i}\right)+\frac{2}{n\ln 2}\cdot\prod_{i=1}^{k}\left(1-\frac{1}{p_i-1}\right)\right)\cdot \mathtt{ppt}_n.$$

where $\mathtt{div}_n$ is the time required to divide an n-bit integer by a word-sized integer and $\mathtt{ppt}_n$ is the time required to perform the probabilistic primality test on an n-bit integer.

Proof. We denote by X_i the event of dividing an n-bit integer by p_i for $1 \le i \le k$ in the trial division, by Y the event of performing the probabilistic primality test on r, and by Y' the event of performing the probabilistic primality test on $(r-1)/2$. Let $\Pr\{X_i\}$, $\Pr\{Y\}$, and $\Pr\{Y'\}$ denote the probabilities of X_i, Y, and Y', respectively. We denote by $\mathtt{div}_n(i)$ the time required to divide an n-bit integer by p_i, by $\mathtt{ppt}_n$ ($\mathtt{ppt}_{n-1}$) the time required to perform the probabilistic primality test on an n-bit ($(n-1)$-bit) integer. Then, $S_n(k)$ can be represented as follows.

$$S_n(k) = \sum_{i=1}^{k}(\Pr\{X_i\}\cdot \mathtt{div}_n(i)) + \Pr\{Y\}\cdot \mathtt{ppt}_n + \Pr\{Y'\}\cdot \mathtt{ppt}_{n-1}. \quad (1)$$

We show how to compute $\Pr\{X_i\}$, $\Pr\{Y\}$, and $\Pr\{Y'\}$. We first consider the probability $\Pr\{X_i\}$ that we divide an n-bit integer r by p_i in the trial division. We divide r by p_i if and only if $r \not\equiv 0, 1 \pmod{p_j}$ for all $1 \le j \le i-1$. Hence, the probability $\Pr\{X_i\}$ is

$$\Pr\{X_i\} = \prod_{j=1}^{i-1}\left(1-\frac{2}{p_j}\right). \quad (2)$$

The probability $\Pr\{Y\}$ that we perform the probabilistic primality test on r is the probability that $r \not\equiv 0, 1 \pmod{p_j}$ for all $1 \le j \le k$. Hence, the probability $\Pr\{Y\}$ is

$$\Pr\{Y\} = \prod_{i=1}^{k}\left(1-\frac{2}{p_i}\right). \quad (3)$$

Consider the probability $\Pr\{Y'\}$ that we perform the probabilistic primality test on $(r-1)/2$. Let q_n denote the probability that an odd n-bit integer r is a prime. According to the prime number theorem [20],

$$q_n \approx \frac{1}{2^{n-2}}\left(\frac{2^n}{n\ln 2}-\frac{2^{n-1}}{(n-1)\ln 2}\right) \approx \frac{2}{n\ln 2}. \quad (4)$$

We perform the probabilistic primality test on $(r-1)/2$ if and only if r is a prime and $(r-1)/2$ is not divisible by any primes up to p_k. (We assume that the error probability of the primality test is equal to 0.) We consider the probability that $(r-1)/2$ is not divisible by any primes up to p_k when r is a prime. It should be noted that r is a prime already implies $(r-1)/2 \not\equiv -1\cdot 2^{-1} \bmod p_i$ because $r \not\equiv 0 \bmod p_i$. Hence the probability that $(r-1)/2 \not\equiv 0 \bmod p_i$ when r is a prime is $1-1/(p_i-1)$ and the probability $\Pr\{Y'\}$ is

$$\Pr\{Y'\} \approx \frac{2}{n\ln 2}\cdot\prod_{i=1}^{k}\left(1-\frac{1}{p_i-1}\right). \quad (5)$$

Hence, by replacing $\Pr\{X_i\}$, $\Pr\{Y\}$, and $\Pr\{Y'\}$ in Eq. (1) by Eqs. (2) - (5), we get the following approximation.

$$S_n(k) \approx \sum_{i=1}^{k}\left(\prod_{j=1}^{i-1}\left(1-\frac{2}{p_j}\right)\cdot \mathtt{div}_n(i)\right) + \prod_{i=1}^{k}\left(1-\frac{2}{p_i}\right)\cdot \mathtt{ppt}_n + \frac{2}{n\ln 2}\cdot\prod_{i=1}^{k}\left(1-\frac{1}{p_i-1}\right)\cdot \mathtt{ppt}_{n-1}$$

We can simplify $S_n(k)$ further in real situation. Since n is quite large (512, 1024 or 2048), $\mathtt{ppt}_n \approx \mathtt{ppt}_{n-1}$ and thus $\mathtt{ppt}_{n-1}$ can be replaced by $\mathtt{ppt}_n$. In addition, $\mathtt{div}_n(i) \approx \mathtt{div}_n(j)$ for $1 \le i, j \le k$ because p_i and p_j are normally stored in a word of a machine and thus both $\mathtt{div}_n(i)$ and

$\mathtt{div}_n(j)$ can be replaced by $\mathtt{div}_n$, the time required to divide an n-bit integer by a word-sized integer. Hence, $S_n(k)$ can be rewritten as in the claim of this theorem, which completes the proof. □

3. Finding the Optimal Number of Primes Used in the Trial Division

We compute the optimal value k_{opt} minimizing $S_n(k)$, i.e., the optimal number of primes used in the trial division to make the safe primality test fastest.

Theorem 2. *The expected running time $S_n(k)$ is minimized at k_{opt} that satisfies*

$$\frac{2}{p_{k_{opt}}} \cdot \left(1 + \frac{1}{n \ln 2} \cdot \prod_{i=1}^{k_{opt}} \frac{p_i}{p_i - 1}\right) = \frac{\mathtt{div}_n}{\mathtt{ppt}_n}.$$

Proof. We first compute $\Delta S_n(k) = S_n(k) - S_n(k-1)$, then show $\Delta S_n(k)$ is negative when $k < k_{opt}$, zero at k_{opt} and positive when $k > k_{opt}$, which means $S_n(k)$ is minimized at k_{opt}.

$$\begin{aligned}
\Delta S_n(k) &\approx \sum_{i=1}^{k} \prod_{j=1}^{i-1} \left(1 - \frac{2}{p_j}\right) \mathtt{div}_n \\
&+ \left(\prod_{i=1}^{k} \left(1 - \frac{2}{p_i}\right) + \frac{2}{n \ln 2} \prod_{i=1}^{k} \left(1 - \frac{1}{p_i - 1}\right)\right) \mathtt{ppt}_n \\
&- \left(\sum_{i=1}^{k-1} \prod_{j=1}^{i-1} \left(1 - \frac{2}{p_j}\right) \mathtt{div}_n + \left(\prod_{i=1}^{k-1} \left(1 - \frac{2}{p_i}\right)\right.\right. \\
&\left.\left. + \frac{2}{n \ln 2} \prod_{i=1}^{k-1} \left(1 - \frac{1}{p_i - 1}\right)\right) \mathtt{ppt}_n\right) \\
&= \prod_{j=1}^{k-1} \left(1 - \frac{2}{p_j}\right) \mathtt{div}_n - \left(\prod_{i=1}^{k-1} \left(1 - \frac{2}{p_i}\right) \frac{2}{p_k}\right. \\
&\left. + \frac{2}{n \ln 2} \prod_{i=1}^{k-1} \left(1 - \frac{1}{p_i - 1}\right)\left(\frac{1}{p_k - 1}\right)\right) \mathtt{ppt}_n \\
&= \prod_{i=1}^{k-1} \left(1 - \frac{2}{p_i}\right) \left(\mathtt{div}_n - \left(\frac{2}{p_k}\right.\right. \\
&\left.\left. + \frac{2}{n \ln 2} \frac{\prod_{i=1}^{k-1}(1 - \frac{1}{p_i - 1})}{\prod_{i=1}^{k-1}(1 - \frac{2}{p_i})} \cdot \frac{1}{p_k - 1}\right) \mathtt{ppt}_n\right) \\
&= \prod_{i=1}^{k-1} \left(1 - \frac{2}{p_i}\right) \left(\mathtt{div}_n \right. \\
&\left. - \frac{2}{p_k} \left(1 + \frac{1}{n \ln 2} \prod_{i=1}^{k} \frac{p_i}{p_i - 1}\right) \mathtt{ppt}_n\right) \qquad (6)
\end{aligned}$$

By definition, $\Delta S_n(k) = 0$ at k_{opt}. It remains to show $\Delta S_n(k)$ is negative when $k < k_{opt}$ and positive when $k > k_{opt}$.

Let $\alpha(k)$ denote $\frac{1}{p_k}\left(1 + \frac{1}{n \ln 2} \cdot \prod_{i=1}^{k} \frac{p_i}{p_i - 1}\right)$. Then, $\Delta S_n(k)$ in Eq. (6) can be represented as follows.

$$\Delta S_n(k) = \prod_{i=1}^{k-1} \left(1 - \frac{2}{p_i}\right) \cdot (\mathtt{div}_n - 2\alpha(k)\mathtt{ppt}_n) \qquad (7)$$

Since $\mathtt{ppt}_n >> \mathtt{div}_n$, $\mathtt{div}_n - 2\alpha(k)\mathtt{ppt}_n$ is negative when k is sufficiently small. We show $\alpha(k)$ decreases as k grows bigger, which implies $\mathtt{div}_n - 2\alpha(k)\mathtt{ppt}_n$ increases and thus $\Delta S_n(k)$ is negative when $k < k_{opt}$ and positive when $k > k_{opt}$. We can show $\alpha(k)$ decreases by showing $\alpha(k)/\alpha(k-1) < 1$ because it is positive.

$$\begin{aligned}
\frac{\alpha(k)}{\alpha(k-1)} &= \frac{\frac{1}{p_k}\left(1 + \frac{1}{n \ln 2} \cdot \prod_{i=1}^{k} \frac{p_i}{p_i - 1}\right)}{\frac{1}{p_{k-1}}\left(1 + \frac{1}{n \ln 2} \cdot \prod_{i=1}^{k-1} \frac{p_i}{p_i - 1}\right)} \\
&= \frac{p_{k-1}}{p_k} \cdot \frac{1 + A \cdot \frac{p_k}{p_k - 1}}{1 + A} \qquad (8) \\
&\text{where } A = \frac{1}{n \ln 2} \cdot \prod_{i=1}^{k-1} \frac{p_i}{p_i - 1}.
\end{aligned}$$

Since $p_k/(p_k - 1) > 1$ and $A > 0$,

$$\frac{1 + A \cdot \frac{p_k}{p_k - 1}}{1 + A} < \frac{p_k}{p_k - 1}$$

This inequality can be shown easily by subtracting the left-hand side from the right-hand side. Using this inequality, we get the following inequality.

$$\frac{\alpha(k)}{\alpha(k-1)} < \frac{p_{k-1}}{p_k} \cdot \frac{p_k}{p_k - 1} = \frac{p_{k-1}}{p_k - 1}.$$

Since both p_k and p_{k-1} are odd primes, $p_k > p_{k-1} + 1$ and $\frac{p_{k-1}}{p_k - 1} < 1$ and $\alpha(k)/\alpha(k-1) < 1$. Now, we showed $\alpha(k)$ decreases, which means $\Delta S_n(k)$ is negative when $k < k_{opt}$ and positive when $k > k_{opt}$.

The last one we would show is there exists k_{opt} satisfying $\Delta S_n(k_{opt}) = 0$ in every case. We already showed that $\alpha(k)$ decreases, however, this does not imply k_{opt} always exists. Consider that $\lim_{k \to \infty} \alpha(k) = C$ for some $C > 0$. If $C > \mathtt{div}_n/(2\ \mathtt{ppt}_n)$, $\mathtt{div}_n - 2\alpha(k)\mathtt{ppt}_n$ (thus $\Delta S_n(k)$) is negative for all k's and thus k_{opt} satisfying $\Delta S_n(k_{opt}) = 0$ does not exist. Therefore, we will show $\lim_{k \to \infty} \alpha(k) = 0$ which means $\lim_{k \to \infty} \Delta S_n(k) > 0$ and there always exists k_{opt} satisfying $\Delta S_n(k_{opt}) = 0$.

Recall that

$$\alpha(k) = \frac{1}{p_k}\left(1 + \frac{1}{n \ln 2} \cdot \prod_{i=1}^{k} \frac{p_i}{p_i - 1}\right).$$

We first show that $\prod_{i=1}^{k} \frac{p_i}{p_i - 1} = O(\ln p_{k-1})$ as follows.

$$\prod_{i=1}^{k} \frac{p_i}{p_i - 1} = \prod_{i=1}^{k} \left(1 + \frac{1}{p_i - 1}\right)$$
$$= \left(1 + \frac{1}{3-1}\right) \prod_{i=2}^{k} \left(1 + \frac{1}{p_i - 1}\right)$$

Since $p_i - 1 > p_{i-1}$, $\frac{1}{p_i - 1} < \frac{1}{p_{i-1}}$ and $1 + \frac{1}{p_i - 1} < 1 + \frac{1}{p_{i-1}}$ and thus

$$\left(1 + \frac{1}{3-1}\right) \prod_{i=2}^{k} \left(1 + \frac{1}{p_i - 1}\right) < 2 \prod_{i=1}^{k-1} \left(1 + \frac{1}{p_i}\right)$$

Since $1 + 1/p_i \le \exp(1/p_i)$,

$$2 \prod_{i=1}^{k-1} \left(1 + \frac{1}{p_i}\right) \le 2 \prod_{i=1}^{k-1} \exp(1/p_i) = 2 \exp\left(\sum_{i=1}^{k-1} 1/p_i\right).$$

Since $\sum_{i=1}^{k-1} 1/p_i < \ln\ln p_{k-1} + B + 1/\ln^2 p_{k-1}$ for prime-reciprocal constant B ($\approx$ 0.261) by the Theorem 8.8.5 in [30], $\sum_{i=1}^{k-1} 1/p_i = O(\ln\ln p_{k-1})$ and thus,

$$2 \exp\left(\sum_{i=1}^{k-1} 1/p_i\right) = 2 \exp(O(\ln\ln p_{k-1})) = O(\ln p_{k-1}).$$

Hence, we showed $\prod_{i=1}^{k-1} \frac{p_i}{p_i - 1} = O(\ln p_{k-1})$. Using this, we can show $\lim_{k\to\infty} \alpha(k) = 0$ in the following way.

$$\lim_{k\to\infty} \alpha(k) = \lim_{k\to\infty} \frac{1}{p_k} \left(1 + \frac{1}{n \ln 2} \prod_{i=1}^{k} \frac{p_i}{p_i - 1}\right)$$
$$= \lim_{k\to\infty} \frac{1}{p_k} \left(1 + \frac{1}{n \ln 2} O(\ln p_{k-1})\right) = 0$$

□

We showed how to compute the optimal value k_{opt} minimizing $S_n(k)$ in Theorem 2. Now, we introduce a simple and useful approximation rule to compute k_{opt} in practical situation. Most cryptographic algorithms require very large safe primes e.g., 1024 or 2048-bit safe primes are required to guarantee the securities of the cryptographic algorithms. Thus we can assume that $n \ge 1{,}024$ in practice. In addition, it is not common to store more than 100,000 smallest odd primes for trial division. Note that the 100,000th smallest odd prime $p_{100,000}$ is 1,299,721, which is even bigger than a million. Our experimental results in the next section show k_{opt}'s for a PC are far less than 100,000. Thus, we focus on finding k_{opt} when $n \ge 1{,}024$ and $k_{opt} \le 100{,}000$. The following theorem presents a simple approximation rule to compute k_{opt} in this situation. Although this rule is much simpler than the accurate formula in Theorem 2, it still estimates $p_{k_{opt}}$ quite well.

Theorem 3. $p_{k_{opt}} \approx 2 \cdot \mathtt{ppt}_n/\mathtt{div}_n$ *when* $n \ge 1{,}024$ *and* $k_{opt} \le 100{,}000$ (*i.e.*, $p_{k_{opt}} \le 1{,}299{,}721$).

Proof. We show that the expression '$1 + \frac{1}{n \ln 2} \cdot \prod_{i=1}^{k_{opt}} \frac{p_i}{p_i - 1}$', in Theorem 2 is close to 1 on condition that $n \ge 1{,}024$ and $k_{opt} \le 100{,}000$. Since each $\frac{p_i}{p_i - 1}$ is bigger than 1, $\prod_{i=1}^{k_{opt}} \frac{p_i}{p_i - 1}$ increases as k_{opt} grows bigger and since $\prod_{i=1}^{100{,}000} \frac{p_i}{p_i - 1} < 12.54$, $\prod_{i=1}^{k_{opt}} \frac{p_i}{p_i - 1}$ is at most 12.54. From the fact $\prod_{i=1}^{k_{opt}} \frac{p_i}{p_i - 1}$ is at most 12.54 and that $n \ge 1{,}024$, one can easily derive that $0 < \frac{1}{n \ln 2} \cdot \prod_{i=1}^{k_{opt}} \frac{p_i}{p_i - 1} < 0.018$ and thus $1 + \frac{1}{n \ln 2} \cdot \prod_{i=1}^{k_{opt}} \frac{p_i}{p_i - 1} \approx 1$. (Note that $\frac{1}{n \ln 2} \cdot \prod_{i=1}^{k_{opt}} \frac{p_i}{p_i - 1}$ decreases, as n increases: When $n = 2{,}048$, $\frac{1}{n \ln 2} \cdot \prod_{i=1}^{k_{opt}} \frac{p_i}{p_i - 1} < 0.009$.) Hence, we can replace the expression '$1 + \frac{1}{n \ln 2} \cdot \prod_{i=1}^{k_{opt}} \frac{p_i}{p_i - 1}$' in Theorem 2 by 1, and we get $p_{k_{opt}} \approx 2 \cdot \mathtt{ppt}_n/\mathtt{div}_n$. □

We can practically compute an approximate $p_{k_{opt}}$ when $n \ge 1024$ in the following way. We first compute $p_{k_{opt}}$ by Theorem 3. If it is bigger than 1,299,791, it is abandoned. Otherwise (if the computed $p_{k_{opt}}$ is 1,299,791 or less), we check if it is a real approximation. (Note that Theorem 3 does not preclude the rare possibility that the computed $p_{k_{opt}}$ is 1,299,791 or less even though the real $p_{k_{opt}}$ is bigger than 1,299,791.) Thus, we check if the computed $p_{k_{opt}}$ satisfies the equation in Theorem 2 approximately. If so, it is a real approximation. Otherwise, it it abandoned. Note that calculating both sides of the equation in Theorem 2 is not difficult, once $p_{k_{opt}}$ is given.

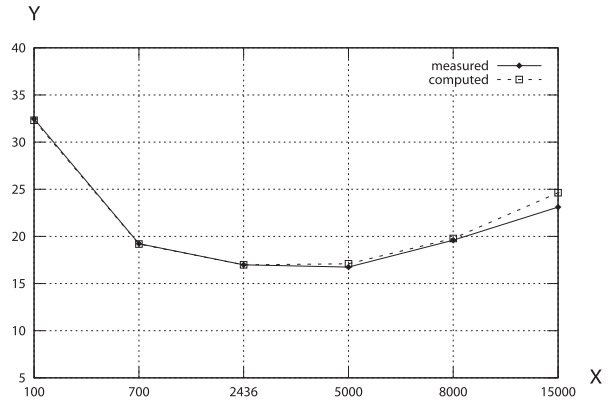
It should be noted Theorem 3 gracefully degrades when $n < 1024$. For example, when $n = 512$, $0 < \frac{1}{n \ln 2} \cdot \prod_{i=1}^{k_{opt}} \frac{p_i}{p_i - 1} < 0.036$, which implies the approximation is still useful even though it is a little less accurate.

4. Experimental Results

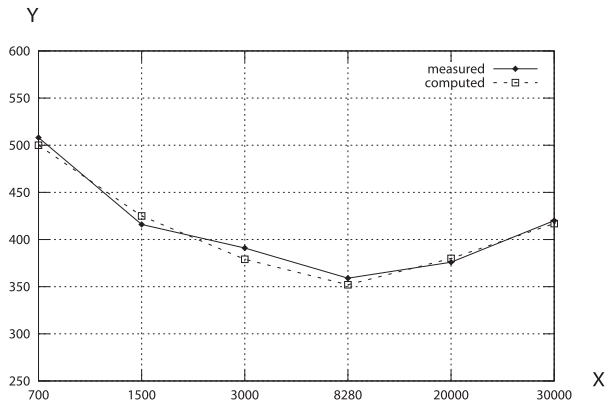
We measured the running time of the safe primality test, in which Miller-Rabin test is used for probabilistic primality test, with various number of primes being used in the trial division and compared them with the expected running time for each case by Theorem 1 (Fig. 1). We did experiments on generating 1024-bit and 2048-bit primes by implementing and running a C program on a PC with an intel 2.4 Ghz Core2Duo 6600 CPU and 2 GB main memory.

In the graphs, X-axis means the number of smallest odd primes used in the trial division and Y-axis means the safe prime generation time in seconds. Solid lines are experimental results, which are average measured running times of 1,000 safe primality tests for each case. Dashed lines are expected running times computed by Theorem 1 from $\mathtt{div}_{1,024}$ (= 734 ns) and $\mathtt{ppt}_{1,024}$ (= 7,875 μs) for 1024 bits and $\mathtt{div}_{2,048}$ (= 1,328 ns) and $\mathtt{ppt}_{2,048}$ (= 56,062 μs) for 2048 bits. In this figure, the expected running times are very close to the measured running times.

We also computed k_{opt} and compared it with the measured data. For 1,024 bits, the k_{opt} computed from $\mathtt{div}_{1,024}$ and $\mathtt{ppt}_{1,024}$ by Theorem 2 is 2,436 where $p_{2,436} = 21{,}727$. This computed optimal value corresponds to the experimental results because the measured running time is fastest around 2,436 primes in Fig. 1 (a). For 2,048 bits, the computed k_{opt} is 8,280 and the measured running time is also fastest around 8,280 primes where $p_{8,280} = 85{,}037$



(a) 1024 bits



(b) 2048 bits

Fig. 1 Experimental Results on safe primality tests.

Table 1 Comparison of accurate and approximate $p_{k_{opt}}$'s.

bits	accurate $p_{k_{opt}}$	approximate $p_{k_{opt}}$	diff
1024	$p_{2,436} = 21{,}727$	$p_{2,406} = 21{,}467$	1.20%
2048	$p_{8,280} = 85{,}037$	$p_{8,228} = 84{,}431$	0.71%

(Fig. 1 (b)). Hence, the experimental results show Theorem 1 and 2 estimate the running times and k_{opt}'s very well.

We also show the approximate rule in Theorem 3 estimates $p_{k_{opt}}$ quite well enough to use in practice.

In Table 1, the accurate $p_{k_{opt}}$ is computed by Theorem 2 and the approximate $p_{k_{opt}}$ is computed by Theorem 3. The difference between them, which is computed by '(accurate $p_{k_{opt}}$ − approximate $p_{k_{opt}}$) × 100/accurate $p_{k_{opt}}$', is very small (about 1%) and thus one can use the approximation rule in most cases except the case that the accurate k_{opt} is really necessary, which is rare.

5. Concluding Remark

We presented probabilistic analysis to compute the expected running time and k_{opt}, the optimal number of primes used in the trial division for the safe primality test based on random search. In addition, we suggested a simple and useful approximation rule for computing k_{opt}. The experimental results showed that our analysis estimates the behavior of the safe primality test very well.

In the preliminary version of this paper [31], we also presented an analysis of another safe primality test used in OpenSSL [32] which is not included in this paper. That safe primality test is similar to the one analyzed in this paper except that it does two trial divisions, each of which is on r and $(r-1)/2$, respectively. Obviously, it is inferior to the one analyzed in this paper and, in addition, its analysis is very similar to the analysis given in this paper. Thus, we did not include the analysis of the inferior test in this paper.

Our future work is to analyze the expected running time and k_{opt} of incremental search for safe prime generation based on the analysis on random search, which is harder to analyze.

References

[1] R.L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," Commun. ACM, vol.21, no.2, pp.120–126, 1978.
[2] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," IEEE Trans. Inf. Theory, vol.31, no.4, pp.469–472, 1985.
[3] National Institute for Standards and Technology, "Digital Signature Standard (DSS)," Federal Register, vol.56, p.169, 1991.
[4] W. Diffie and M. Hellman, "New directions in cryptography," IEEE Trans. Inf. Theory, vol.22, no.6, pp.644–654, 1976.
[5] G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik, "A practical and provably secure coalition-resistant group signature scheme," CRYPTO'00, LNCS 1880, pp.255–270, 2000.
[6] R. Cramer and V. Shoup, "Signature schemes based on the strong RSA assumption," Proc. 6th ACM CCS, pp.46–52, 1999.
[7] R. Gennaro, S. Halevi, and T. Rabin, "Secure hash-and-sign signature without the random oracle," EUROCRYPT'99, LNCS 1592, pp.123–139, 1999.
[8] R. Gennaro, H. Krawczyk, and T. Rabin, "RSA-based Undeniable signature," CRYPTO'97, LNCS 1294, pp.132–149, 1997.
[9] V. Shoup, "Practical threshold signatures," EUROCRYPT'00, LNCS 1087, pp.207–220, 2000.
[10] J. Camenisch and A. Lysyanskaya, "Efficient non-transferable anonymous multi-show credential system with optional anonymity revocation," EUROCRYPT'01, LNCS 2045, pp.93–118, 2001.
[11] E. Fujisaki and T. Okamoto, "Statistical zero knowledge protocols to prove modular polynomial relations," CRYPTO'97, LNCS 1294, pp.16–30, 1997.
[12] J. Algesheimer, J. Camenisch, and V. Shoup, "Efficient computation modulo a shared secret with application to the generation of shared safe-prime products," CRYPTO'02, LNCS 2442, pp.417–432, 2002.
[13] E. Ong and J. Kubiatowicz, "Optimizing robustness while generating shared secret safe primes," PKC'05, LNCS 3386, pp.120–137, 2005.
[14] J. Camenisch and M. Michels, "Proving in zero-knowledge that a number is the product of two safe primes," CRYPTO'99, LNCS 1592, pp.107–122, 1999.
[15] M.H. Ibrahim, "Verifiable threshold sharing of a large secret safe-prime," ITCC'05, pp.608–613, 2005.
[16] M.H. Ibrahim, "Eliminating quadratic slowdown in two-prime RSA function sharing," Int. J. Netw. Secur., vol.7, no.1, pp.107–114, 2008.
[17] M.H. Ibrahim, "Efficient dealer-less threshold sharing of standard RSA," Int. J. Netw. Secur., vol.8, no.1, pp.134–145, 2009.
[18] M.H. Ibrahim, I.A. Ali, I.I. Ibrahim, and A.H. El-Sawy, "Fast fully distributed and threshold RSA function sharing," Proc. Information Systems: New Generation Conference (ISNG 2004), pp.11–15, 2004.

[19] A.J. Menezes, P.C. Oorschot, and S.A. Vanstone, Handbook of Applied Cryptography, CRC Press, 1997.
[20] T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein, Introduction to Algorithms, 2nd ed., MIT Press, 2001.
[21] H.C. Pocklington, "The determination of the prime or composite nature of large numbers by Fermat's theorem," Proc. Cambridge Philosophical Society, vol.18, pp.29–30, 1914.
[22] U.M. Maurer, "Fast generation of prime numbers and secure public-key cryptographic parameters," J. Cryptology, vol.8, no.3, pp.123–155, 1995.
[23] M. Agrawal, N. Kayal, and N. Saxena, "PRIMES is in P," Annals of Mathematics, vol.160, no.2, pp.781–793, 2004.
[24] R. Solovay and V. Strassen, "A fast Monte-Carlo test for primality," SIAM J. Comput., vol.6, pp.84–85, 1977.
[25] G.L. Miller, "Riemann's hypothesis and tests for primality," J. Comput. Syst. Sci., vol.13, no.3f, pp.300–317, 1976.
[26] M.O. Rabin, "Probabilistic algorithm for primality testing," J. Number Theory, vol.12, pp.128–138, 1980.
[27] J. Brandt, I. Damgard, and P. Landrock, "Speeding up prime number generation," ASIACRIPT'91, LNCS 739, pp.440–449, 1991.
[28] H. Park, "An efficient implementation of safe prime generation," International Conference on Ubiquitous Computing, pp.241–243, Oct. 2003.
[29] M.J. Wiener, "Safe prime generation with a combined sieve," http://eprint.iacr.org/2003/186.ps.gz
[30] E. Bach and J. Shallit, Algorithmic Number Theory, vol.1, 2nd printing, MIT Press, 1997.
[31] H. Park, S.K. Park, K. Kwon, and D.K. Kim, "Probabilistic analyses on finding optimal combinations of primality tests in real applications," ISPEC'05, LNCS 3439, pp.74–84, 2005.
[32] OpenSSL, http://www.openssl.org

Heejin Park received the B.S., M.S. and Ph.D. degrees in Computer Engineering from Seoul National University in 1994, 1996, and 2001, respectively. From 2001 to 2002, he worked as a post-doctoral researcher for the Department of Computer Engineering at Seoul National University. From 2003 to 2003, he was a research professor at Ewha Womens University. He is currently an associate professor in the Department of Computer Science and Engineering at Hanyang University, Korea. His research interests are in the areas of cryptography, information security, and computer algorithm.

Dong Kyue Kim received the B.S., M.S. and Ph.D. degrees in Computer Engineering from Seoul National University in 1992, 1994, and 1999, respectively. From 1999 to 2005, he was an assistant professor in the Division of Computer Science and Engineering at Pusan National University. He is currently an associate professor in the Division of Electronics and Computer Engineering at Hanyang University, Korea. His research interests are in the areas of embedded security systems, crypto-coprocessors, and information security.