Probabilistic Primality Testing and Analysis of Probabilistic AKS

Emilie Ma

Summer Research School 2021
Apriltsi, Bulgaria

Introduction

Abstract

This paper aims to analyze the Fermat, the Euler (Solovay-Strassen), and the Miller-Rabin primality tests, three well-known probabilistic algorithms based on Fermat's little theorem. The Agarwal-Kayal-Saxena test, the first polynomial time deterministic primality test developed, is also discussed, as well as a proposal for a new probabilistic adaptation. This probabilistic AKS was found to deliver significant running time decreases, at the expense of eliminating determinism and passing a considerable amount of pseudoprimes.

Introduction

Abstract

This paper aims to analyze the Fermat, the Euler (Solovay-Strassen), and the Miller-Rabin primality tests, three well-known probabilistic algorithms based on Fermat's little theorem. The Agarwal-Kayal-Saxena test, the first polynomial time deterministic primality test developed, is also discussed, as well as a proposal for a new probabilistic adaptation. This probabilistic AKS was found to deliver significant running time decreases, at the expense of eliminating determinism and passing a considerable amount of pseudoprimes.

- In layman's terms: looked at a variety of tests for checking if a number is prime, and analyzed their performance with regards to speed and accuracy

Primality Testing

Primality Testing

- Why is primality testing important?
  - Modern cryptography and cybersecurity
  - Generates a random number, runs through primality test, if prime, then OK for use

Primality Testing

- Why is primality testing important?
  - Modern cryptography and cybersecurity
  - Generates a random number, runs through primality test, if prime, then OK for use
- Well-known primality tests include the Fermat, Euler (Solovay-Strassen), Miller-Rabin, and Agarwal-Kayal-Saxena (AKS) tests

Probabilistic vs Deterministic

- The first three tests mentioned above (Fermat, Euler, and Miller-Rabin) are probabilistic
    - *Probabilistic* $\rightarrow$ element of randomness
    - *Deterministic* $\rightarrow$ given the same input, always returns same output

Probabilistic vs Deterministic

- The first three tests mentioned above (Fermat, Euler, and Miller-Rabin) are probabilistic
  - *Probabilistic* $\rightarrow$ element of randomness
  - *Deterministic* $\rightarrow$ given the same input, always returns same output
- Why wouldn't we always use deterministic tests?
  - Often much slower, especially as we'll see with the AKS test
  - High performance demands where 100% accuracy can be sacrificed

Probabilistic vs Deterministic

- The first three tests mentioned above (Fermat, Euler, and Miller-Rabin) are probabilistic
    - *Probabilistic* $\rightarrow$ element of randomness
    - *Deterministic* $\rightarrow$ given the same input, always returns same output
- Why wouldn't we always use deterministic tests?
    - Often much slower, especially as we'll see with the AKS test
    - High performance demands where 100% accuracy can be sacrificed
- Research goal: how can we take the best of both worlds of probabilistic and deterministic tests?

Background Theory

Preliminary Notes

- Assumes a basic knowledge of modular congruences
- All $\log n$ shown are $\log_2 n$ unless otherwise marked

Agarwal-Kayal-Saxena Primality Test

- Notable for being the first deterministic primality that runs in polynomial time (bounds of $\widetilde{O}(\log^{15/2} n)$)

Agarwal-Kayal-Saxena Primality Test

- Notable for being the first deterministic primality that runs in polynomial time (bounds of $\widetilde{O}(\log^{15/2} n)$)
- Based on the theorem that for any a where a is coprime to n and where $n \geq 2$, the following holds within the polynomial ring $\mathbb{Z}[x]$:

$$(X + a)^n \equiv X^n + a \pmod{n}$$

Agarwal-Kayal-Saxena Primality Test

- The test has 5 key steps:

Agarwal-Kayal-Saxena Primality Test

- The test has 5 key steps:
  - Ensure n is not a perfect power ($n = a^k$); if it is, n is composite

Agarwal-Kayal-Saxena Primality Test

- The test has 5 key steps:
  - Ensure n is not a perfect power ($n = a^k$); if it is, n is composite
  - Find smallest r such that $\mathrm{ord}_r(n) \geq \log n^2$

Agarwal-Kayal-Saxena Primality Test

- The test has 5 key steps:
  - Ensure n is not a perfect power ($n = a^k$); if it is, n is composite
  - Find smallest r such that $\mathrm{ord}_r(n) \geq \log n^2$
  - Check that no number a $2 \leq a \leq r$ divides n; if a does, n is composite

Agarwal-Kayal-Saxena Primality Test

- The test has 5 key steps:
  - Ensure n is not a perfect power ($n = a^k$); if it is, n is composite
  - Find smallest r such that $\mathrm{ord}_r(n) \geq \log n^2$
  - Check that no number a $2 \leq a \leq r$ divides n; if a does, n is composite
  - Check $n \leq r$; if so, n is prime.

Agarwal-Kayal-Saxena Primality Test

- The test has 5 key steps:
    - Ensure n is not a perfect power ($n = a^k$); if it is, n is composite
    - Find smallest r such that $\mathrm{ord}_r(n) \geq \log n^2$
    - Check that no number a $2 \leq a \leq r$ divides n; if a does, n is composite
    - Check $n \leq r$; if so, n is prime.
    - Compute all $(X + a)^n \equiv X^n + a \pmod{n}$ for $1 \leq a \leq \lfloor \sqrt{\phi(r)} \log n \rfloor$; if n passes all these tests, it is prime, otherwise it is composite.

Probabilistic AKS Primality Test

- Step 5 in the AKS algorithm has very high bounds, and is very expensive computationally to check - how can we change these bounds to make AKS faster?

Probabilistic AKS Primality Test

- Step 5 in the AKS algorithm has very high bounds, and is very expensive computationally to check - how can we change these bounds to make AKS faster?
- Probabilistic AKS: this research project's proposal for a new variant of AKS
    - Instead of checking all a $1 \leq a \leq \lfloor \sqrt{\phi(r)} \log n \rfloor$, choose k random a values from that range to check
    - Similar idea as the random choice of a in the Fermat, Euler, and Miller-Rabin tests

Probabilistic AKS Primality Test

- Step 5 dominates the time taken, due to high numbers of computations depending on the value of r, bounded at $O(\log^3 n)$ [1]

Probabilistic AKS Primality Test

- Step 5 dominates the time taken, due to high numbers of computations depending on the value of r, bounded at $O(\log^3 n)$ [1]
  - Each of the $(X + a)^n \equiv X^n + a \pmod{n}$ equations requires $O(r \times \log^2 n)$ computations

Probabilistic AKS Primality Test

- Step 5 dominates the time taken, due to high numbers of computations depending on the value of r, bounded at $O(\log^3 n)$ [1]
  - Each of the $(X + a)^n \equiv X^n + a \pmod{n}$ equations requires $O(r \times \log^2 n)$ computations
  - Instead of having to check $\lfloor \sqrt{\phi(r)} \log n \rfloor$ calculations $(O(\sqrt{(\log^3 n)} \times \log n) = O(\log^{5/2} n))$, k equations can be checked, leading to a worst-case complexity of:

$$\widetilde{O}(k \times r \times \log^2 n) = \widetilde{O}(k \times \log^5 n)$$

Probabilistic AKS Primality Test

- Step 5 dominates the time taken, due to high numbers of computations depending on the value of r, bounded at $O(\log^3 n)$ [1]
  - Each of the $(X + a)^n \equiv X^n + a \pmod{n}$ equations requires $O(r \times \log^2 n)$ computations
  - Instead of having to check $\lfloor \sqrt{\phi(r)} \log n \rfloor$ calculations $(O(\sqrt{(\log^3 n)} \times \log n) = O(\log^{5/2} n))$, k equations can be checked, leading to a worst-case complexity of:

$$\widetilde{O}(k \times r \times \log^2 n) = \widetilde{O}(k \times \log^5 n)$$

Analysis Methods

Analysis Methods

- To consistently analyze the four primality tests (AKS was not tested for pseudoprimes as it is proven correct and would have taken too long), integers in the range $10^5 \leq x \leq 10^6$ were randomly chosen
    - For the Fermat, Euler, and Miller-Rabin tests, 10^4 integers were chosen
    - For probabilistic AKS, 10^2 integers were chosen (as testing 10^4 integers over a large number of trials was infeasible time-wise for this analysis)

Analysis Methods

- To consistently analyze the four primality tests (AKS was not tested for pseudoprimes as it is proven correct and would have taken too long), integers in the range $10^5 \leq x \leq 10^6$ were randomly chosen
  - For the Fermat, Euler, and Miller-Rabin tests, 10^4 integers were chosen
  - For probabilistic AKS, 10^2 integers were chosen (as testing 10^4 integers over a large number of trials was infeasible time-wise for this analysis)
- Sympy's `isprime` function was used as a reference to test for primality

Results

Probabilistic primality tests - accuracy

Figure 1: The effect of increasing the number of base trials k on pseudoprimes passed

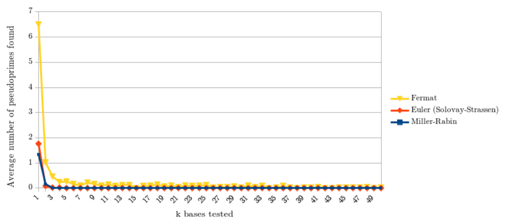
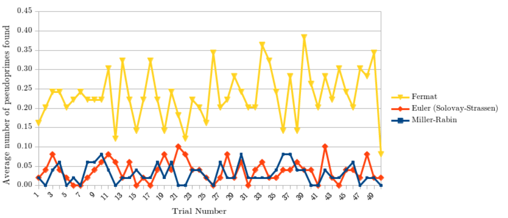


Figure 2: Average pseudoprimes passed across all $1 \leq k \leq 50$ values per trial



- Fermat consistently passed more pseudoprimes

Probabilistic primality tests - accuracy

Figure 1: The effect of increasing the number of base trials k on pseudoprimes passed

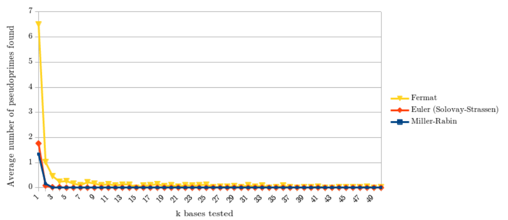
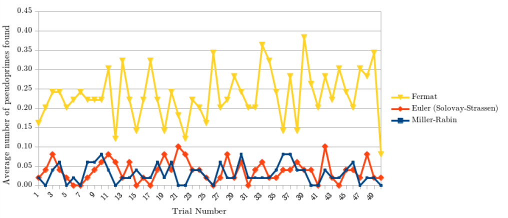


Figure 2: Average pseudoprimes passed across all $1 \leq k \leq 50$ values per trial



- Fermat consistently passed more pseudoprimes
- Euler and Miller-Rabin appear to perform quite similarly but on average Euler will pass more pseudoprimes

Probabilistic primality tests - runtime

Figure 3: The effect of increasing the number of base trials k on pseudoprimes passed

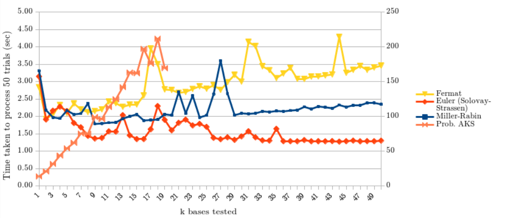
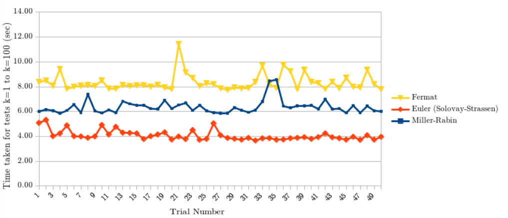


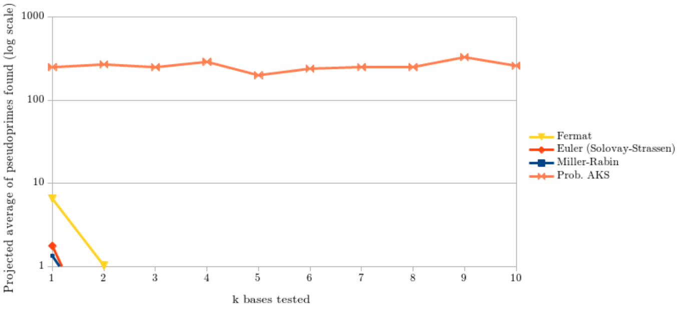
Figure 4: Average time elapsed across all $1 \leq k \leq 50$ values per trial



- A possible explanation for more efficient runtime of Euler and Miller-Rabin is the speed at which numbers are discarded as composite
  - The congruences used in the Euler and Miller-Rabin tests may do so more quickly - though this has not been theoretically verified yet

Projected pseudoprimes vs. k value

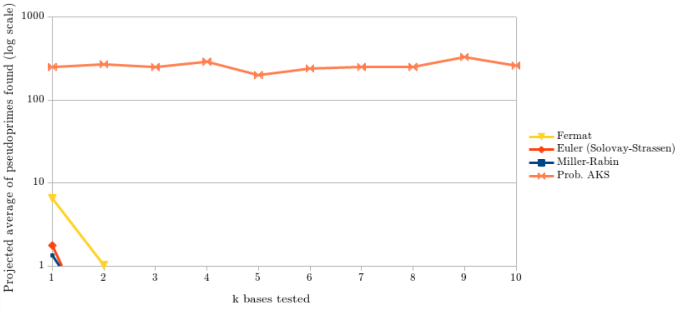
Figure 5: Projected average number of pseudoprimes passed by probabilistic AKS versus other tests



- Significant increase in number of projected pseudoprimes passed by probabilistic AKS compared to the existing tests (200 vs. <10)
    - Projection method: multiply pseudoprimes found by 10^2

Projected pseudoprimes vs. k value

Figure 5: Projected average number of pseudoprimes passed by probabilistic AKS versus other tests



- Significant increase in number of projected pseudoprimes passed by probabilistic AKS compared to the existing tests (200 vs. <10)
  - Projection method: multiply pseudoprimes found by 10^2
- Due to the relatively very low k values tested for probabilistic AKS

Running time of probabilistic AKS

Figure 3: The effect of increasing the number of base trials k on pseudoprimes passed

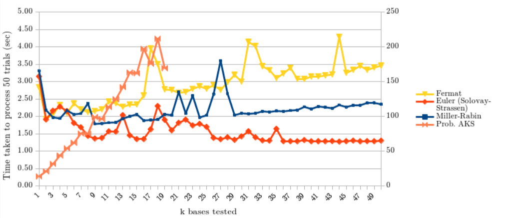
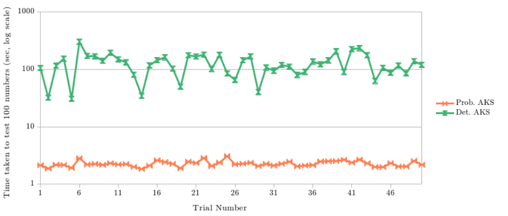


Figure 6: Elapsed running time for probabilistic versus deterministic AKS



- Probabilistic AKS is considerably slower than the existing primality tests, but also significantly faster than deterministic AKS
    - Discrepancies in runtime from probabilistic AKS and existing primality tests can be attributed to time complexity

Sources of Error

- Several possible sources of error were identified in this research project

Sources of Error

- Several possible sources of error were identified in this research project
    - Insufficient bounds of numbers tested; higher amounts of pseudoprimes are found with larger numbers

Sources of Error

- Several possible sources of error were identified in this research project
  - Insufficient bounds of numbers tested; higher amounts of pseudoprimes are found with larger numbers
  - Inconsistent amount of numbers tested for each primality trial: the fifty trials of 10^4 with the Fermat, Euler, and Miller-Rabin tests versus the twenty of 10^2 for probabilistic AKS

Sources of Error

- Several possible sources of error were identified in this research project
  - Insufficient bounds of numbers tested; higher amounts of pseudoprimes are found with larger numbers
  - Inconsistent amount of numbers tested for each primality trial: the fifty trials of 10^4 with the Fermat, Euler, and Miller-Rabin tests versus the twenty of 10^2 for probabilistic AKS
  - Potentially misleading `isprime` Sympy function used; also relies on probabilistic methods

Future Research

- To address the sources of error discussed earlier, future research to be conducted includes:

Future Research

- To address the sources of error discussed earlier, future research to be conducted includes:
    - Experimenting with larger bounds for each primality test; perhaps 10^7 to 10^9

Future Research

- To address the sources of error discussed earlier, future research to be conducted includes:
  - Experimenting with larger bounds for each primality test; perhaps 10^7 to 10^9
  - Running additional trials with the full 10^4 numbers for probabilistic AKS, and increasing the number of trials run for the other primality tests as well

Future Research

- To address the sources of error discussed earlier, future research to be conducted includes:
  - Experimenting with larger bounds for each primality test; perhaps 10^7 to 10^9
  - Running additional trials with the full 10^4 numbers for probabilistic AKS, and increasing the number of trials run for the other primality tests as well
  - Precomputing an array of deterministically verified (with AKS) values to check primality against

Future Research

- To address the sources of error discussed earlier, future research to be conducted includes:
  - Experimenting with larger bounds for each primality test; perhaps 10^7 to 10^9
  - Running additional trials with the full 10^4 numbers for probabilistic AKS, and increasing the number of trials run for the other primality tests as well
  - Precomputing an array of deterministically verified (with AKS) values to check primality against
  - Adjusting the bounds of r in probabilistic AKS to further improve the theoretical runtime complexity

Conclusion

Conclusion

- Fermat: low efficiency, and low accuracy $\rightarrow$ simple to implement, but not practical

Conclusion

- Fermat: low efficiency, and low accuracy $\rightarrow$ simple to implement, but not practical
- Euler: high efficiency, and decent accuracy $\rightarrow$ best in speed-driven scenarios

Conclusion

- Fermat: low efficiency, and low accuracy $\rightarrow$ simple to implement, but not practical
- Euler: high efficiency, and decent accuracy $\rightarrow$ best in speed-driven scenarios
- Miller-Rabin: relatively high efficiency, and higher accuracy than Euler $\rightarrow$ recommended when practical accuracy is key

Conclusion

- Fermat: low efficiency, and low accuracy $\rightarrow$ simple to implement, but not practical
- Euler: high efficiency, and decent accuracy $\rightarrow$ best in speed-driven scenarios
- Miller-Rabin: relatively high efficiency, and higher accuracy than Euler $\rightarrow$ recommended when practical accuracy is key
- Deterministic AKS: no pseudoprimes passed, and very slow to run $\rightarrow$ good for applications where speed is irrelevant

Conclusion

- Fermat: low efficiency, and low accuracy $\rightarrow$ simple to implement, but not practical
- Euler: high efficiency, and decent accuracy $\rightarrow$ best in speed-driven scenarios
- Miller-Rabin: relatively high efficiency, and higher accuracy than Euler $\rightarrow$ recommended when practical accuracy is key
- Deterministic AKS: no pseudoprimes passed, and very slow to run $\rightarrow$ good for applications where speed is irrelevant
- Probabilistic AKS: high numbers of pseudoprimes passed at low k, much faster than deterministic AKS $\rightarrow$ may provide a suitable alternative to deterministic AKS for more practical applications given high enough k values

Acknowledgements

- Many thanks to my mentor, Ms. Pressiana Marinova, for her unfailing guidance and support throughout SRS and the research process, her deep knowledge and clear explanations of new topics, and for always being there to answer all my questions.

Acknowledgements

- Many thanks to my mentor, Ms. Pressiana Marinova, for her unfailing guidance and support throughout SRS and the research process, her deep knowledge and clear explanations of new topics, and for always being there to answer all my questions.
- Much gratitude also to the Summer Research School and High School Student Institute of Mathematics and Informatics for making this research inquiry experience possible, and for hosting such an organized, fun summer program.

Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. **PRIMES is in P.** *Annals of Mathematics*, 160(2):781–793, September 2004.

Richard P Brent. ***Primality Testing.*** Thesis, Australian National University, Mathematical Sciences Institute and College of Engineering and Computer Science, August 2010.

Hendrik Lenstra, Jr. and Carl Pomerance. **Primality testing with Gaussian periods.** July 2005.

Gary L. Miller. **Riemann's Hypothesis and tests for primality.** In *Proceedings of seventh annual ACM symposium on Theory of computing - STOC '75*, pages 234–239, Albuquerque, New Mexico, United States, 1975. ACM Press.

Louis Monier. **Evaluation and comparison of two efficient probabilistic primality testing algorithms.** *Theoretical Computer Science*, 12(1):97–108, September 1980.

Lalitha Kiran Nemana and V. Ch Venkaiah. **An empirical study towards refining the aks primality testing algorithm.** Technical Report 362, 2016.

Carl Pomerance, J. L. Selfridge, and Samuel S Wagstaff. **The Pseudoprimes to 25 • 10^9.** *Mathematics of Computation*, 35(151):1003–1026, July 1980.

Michael O Rabin. **Probabilistic algorithm for testing primality.** *Journal of Number Theory*, 12(1):128–138, February 1980.

Chris Rotella. ***An Efficient Implementation of the AKS Polynomial-Time Primality Proving Algorithm.*** Thesis, Carnegie Mellon University, May 2005.

R. Solovay and V. Strassen. **A Fast Monte-Carlo Test for Primality.** *SIAM Journal on Computing*, 6(1):84–85, March 1977.

Sophoclis Stephanou. **Ssophoclis/AKS-algorithm: Implementation of the AKS primality test algorithm in python., July 2020.**

SymPy Development Team. **Number Theory — SymPy 1.8 documentation, April 2021.**

Tammy Terzian. ***The AKS Primality Test.*** Thesis, The California State University, October 2013.