MATH 377 - Honours Number Theory : The Agrawal-Kayal-Saxena Primality Test

Azeem Hussain

May 4, 2013

1 Introduction

Whether or not a number is prime is one of its most fundamental properties. Many branches of pure mathematics rely heavily on numbers being prime. Prime numbers are in some sense the purest. Irreducible (over $\mathbb{Z}$), they are the building blocks of all other numbers. And yet, determining if a number is prime is not an easy task, certainly not if the number is very large.

There are various algorithms for primality testing. For example, the Sieve of Eratosthenes, developed in Ancient Greece. Reliable, it works on all inputs. The problem is that it takes very long. The running time is exponential in the size of the input.

Miller devised a test for primality that runs in polynomial time [1]. The problem is that is assumes the extended Riemann hypothesis.

Rabin adapted Miller's test to yield a reliable polynomial time primality test, the Miller-Rabin test [2]. This test works well in practice. The problem is that the test is randomized and not guaranteed to work. However, the probability or error can be made miniscule by successive repetition. Nevertheless, to the mathematician, this leaves something to be desired. Ideally, the test will produce a certificate, a proof that the number is prime. This test can only say that the number is "probably prime."

Most primality tests fall into one of these three categories. Either they scale exponentially in time with the input, they rely on some conjecture, or they are randomized. What we seek is a deterministic, polynomial time algorithm that is guaranteed to work on all inputs and be able to produce a certificate for primality.

In the early 2000s, at the Indian Institute of Technology Kanpur, computer scientists Manindra Agrawal, Neeraj Kayal, and Nitin Saxena accomplished exactly that. Their algorithm is presented here, complete with its proof of correctness and time complexity analysis.

1.1 Notation

Much of the work in this paper involves the ring $(\mathbb{Z}/n\mathbb{Z})][x]/(h(x))$, where $h(x)$ is a monic polynomial with integer coefficients. Polynomials $f(x)$ will be reduced modulo $h(x), n$. This means, given a polynomial with integer coefficients $f(x)$, take the remainder on dividing by $h(x)$, then for each coefficient, take the remainder on dividing by n. This is the residue class $f(x) \pmod{h(x), n}$.

A natural number n is said to be a *perfect power* if there exist naturals a and $b > 1$ such that $n = a^b$.

$\operatorname{ord}_r(n)$ denotes the order of n modulo r ; the least natural t such that $n^t \equiv 1 \pmod{r}$.

log denotes the logarithm to the base 2. Any other base will be explicitly specified.

$\varphi(n)$ denotes Euler's totient function; the quantity of numbers coprime to n.

$O(f(n))$ is big oh notation for time complexity analysis. $f(n) \in O(g(n))$ means that there exist k and c such that if $n > k$, then $f(n) \leq cg(n)$.

We define $\tilde{O}(n) := O(n(\log n)^{O(1)})$.

2 Approach

Theorem 2.1 (The child's binomial theorem)**.** *Let $a \in \mathbb{Z}$, $n \in \mathbb{N}$, $n \geq 2$, $(a, n) = 1$. n is prime if and only if*

$$(x + a)^n = x^n + a \pmod{n}. \tag{1}$$

Proof. Suppose $n = p$ is prime. Then every binomial coefficient other than $\binom{p}{0}$ and $\binom{p}{p}$ has a p in its numerator but not in its denominator. Because it is prime, p has no divisors so it remains in the numerator. As such, all the cross terms are divisible by p. By Fermat's little theorem, $a^n = a^{n-1}a \equiv a$ modulo p.

Conversely, suppose n is composite. Let p be a prime factor such that $p^k || n$. Then $p^{k-1} || \binom{n}{p}$, so n does not divide $\binom{n}{p}$. Hence, the expansion has a cross term $x^{n-p}a^p$. $\square$

At first glance, this theorem seems like it could be a primality test itself. However, n coefficients must be evaluated on the left hand side in the worst case, and this gets unwieldy when n is very large. An alternative is to consider the following equation:

$$(x + a)^n = x^n + a \pmod{x^r - 1, n}, \tag{2}$$

where r is an integer less than n. While this lessens the degree of the polynomials, it also breaks the "only if" direction of the child's binomial theorem. For certain values of a and r, some composite ns can satisfy the congruence. (Such composites are akin to Carmichael numbers, in that they share a property common to all primes despite being composite.) One's first thought might be to find the smallest r such that (2) is guaranteed to hold if and only if n is prime. This turns out to be an unsolved problem in number theory [10]. Nevertheless, the algorithm of Agrawal, Kayal, and Saxena is based on these congruences. Find an r “good enough” and such that the composites that will satisfy 2 can be characterized. Test for these composites first, and then check if (2) is always satisfied.

3 Algorithm

Algorithm AKS

```
Input: $n \in \mathbb{N}$
  1. If $n$ is a perfect power, output COMPOSITE.
  2. Find the smallest $r$ such that $\text{ord}_r(n) > \log^2 n$.
  3. If $1 < \gcd(a, n) < n$ for some $a \leq r$, output COMPOSITE.
  4. If $n \leq r$, output PRIME.
  5. For $a = 1$ to $\lfloor \sqrt{\varphi(r)} \log n \rfloor$,
         if $(x + a)^n \neq x^n + a \pmod{x^r - 1, n}$, output COMPOSITE.
  6. Output PRIME.
```

The correctness of the algorithm will now be shown through a sequence of lemmas.

Lemma 3.1. *If n is prime, then AKS outputs* PRIME.

Proof. If n is prime, then it is not a perfect power and its gcd with all lesser integers is 1. Thus Steps 1 and 3 will not return COMPOSITE. The for loop verifies the child's binomial theorem, so by Lemma (2.1), step 5 will not return COMPOSITE. The only remaining outputs are PRIME. □

The converse of this lemma will complete the proof of the correctness of the algorithm. Step 4 will never incorrectly output PRIME, because Step 3 will catch composite n. It remains to show that Step 6 will never incorrectly output PRIME. This will be accomplished using a little bit of algebra, and a lot of ingenuity.

Now, we address Step 2 of the algorithm.

Lemma 3.2. *If $n \geq 6$, there exists a prime $r \in [\log^5 n, 2\log^5 n]$ such that $\text{ord}_r(n) > \log^2 n$.*

Proof. This requires the prime number theorem. A weak version [6] states that for any natural N, the product of primes between N and $2N$ is ≥ 2. Now, let $n \geq 6$ be given and define $N := \log^2 n$, $I := [N, 2N]$. Suppose for a contradiction that $\text{ord}_r(n) \leq \log^2 n$ for all prime $r \in I$. Consider the product of all such r.

$$\prod_{\substack{r \in I \\ r \text{ prime}}} r \quad \text{divides} \quad \prod_{i \leq N} (n^i - 1) \qquad \Longrightarrow \qquad \prod_{\substack{r \in I \\ r \text{ prime}}} r \quad \leq \quad \prod_{i \leq N} (n^i - 1)$$

$$\therefore \quad 2^{\log^5 n} = 2^N < \prod_{\substack{r \in I \\ r \text{ prime}}} r \leq \prod_{i \leq N} (n^i - 1) < n^{\sum_{i \leq N} i} < 2^{\log^5 n}$$

This is a contradiction. □

Note that this lemma shows the existence of a *prime* r, which is stronger than what AKS requires. Relaxing this restriction, and using the following lemma, it can be shown that there is an $r \leq \lceil \log^5 n \rceil$ such that $\text{ord}_r(n) > \log^2 n$, using a similar proof to that of (3.2) [3].

Lemma 3.3 (Nair [5]). *For natural number $m \geq 7$, the least common multiple of the first m natural numbers is $\geq 2^m$.*

Having shown that step 2 of the algorithm gets completed, we move on.

The order of n modulo r is $> \log^2 n > 1$, so there must exist a prime factor of n such that $\text{ord}_r(p) > 1$. (The factor will be n itself, if n is prime.)

If Step 5 is reached, it is because Steps 3 and 4 failed to classify n, so $p > r$. Furthermore, $(n, r) = 1 \Rightarrow n \in (\mathbb{Z}/r\mathbb{Z})^\times$. By its primality, $p \in (\mathbb{Z}/r\mathbb{Z})^\times$. For the remainder of the section, n, p, r will be fixed. Also, define $q := \lfloor \sqrt{\varphi(r)} \log n \rfloor$.

We assume Step 5 of AKS does not output COMPOSITE. The objective now is to show that n cannot be composite. This will be done by constructing a certain group G and, under the choice of q and r, exhibiting a contradiction on $\#G$ on any composite not already ruled out.

For $a = 0, 1, \ldots, q$,

$$\begin{aligned}
 & & (x+a)^n &\equiv x^n + a \pmod{x^r - 1, n} \\
p|n & \;\Rightarrow & (x+a)^n &\equiv x^n + a \pmod{x^r - 1, p} && (3) \\
(1) & \;\Rightarrow & (x+a)^p &\equiv x^p + a \pmod{x^r - 1, p} && (4) \\
(3) \text{ and } (4) & \;\Rightarrow & (x+a)^{\frac{n}{p}} &\equiv x^{\frac{n}{p}} + a \pmod{x^r - 1, p}. && (5)
\end{aligned}$$

For a polynomial $f(x)$, define the set $S := \{k \in \mathbb{N} : (f(x))^k \equiv f(x^k) \pmod{x^r - 1, p}\}$ The elements of this set are said to be *introspective for* $f(x)$. We see that n and $\frac{n}{p}$ are both introspective for $f(x) = x + a$. We will extend this family of polynomials, but first, some lemmas about introspective numbers.

Lemma 3.4. *The introspective numbers for $f(x)$ are closed under multiplication.*

Proof. Let $j, k \in S$. Take the introspection equation with k, replace x by x^j. As $x^r - 1 | x^{jr} - 1$, the reduction can be peformed modulo $(x^r - 1, p)$.

$$\begin{aligned} f(x^k) = & \quad (f(x))^k \quad \pmod{x^r - 1, p} \\ f((x^j)^k) = & \quad (f(x^j))^k \quad \big(\text{mod } (x^j)^r - 1, p\big) \\ f((x^j)^k) = & \quad (f(x^j))^k \quad \pmod{x^r - 1, p} \end{aligned}$$

$$\therefore (f(x))^{jk} \equiv (f(x)^j)^k \equiv (f(x^j))^k \equiv f((x^j)^k)) \equiv f(x^{jk}) \pmod{x^r - 1, p} \tag{6}$$

□

Lemma 3.5. *If k is introspective for $f(x)$ and $g(x)$, then it is introspective for $f(x)g(x)$.*

Proof.

$$((f(x)g(x))^k \equiv (f(x))^k (g(x))^k \equiv f(x^k)g(x^k) \pmod{x^r - 1, p} \tag{7}$$

□

Define the set of integers $I := \{(\frac{n}{p})^i p^j \ : \ i, j \geq 0\}$ and the set of polynomials $P := \{\prod_{a=0}^{q} (x + a)^{e_a} \ : \ e_a \geq 0\}$. By Lemmas 3.4 and 3.5, every number in I is introspective for every polynomial in P. Now, define group H to be the set of residue classes of I modulo r. Note that from step 3 of AKS, $(n, r) = (p, r) = 1$, so H is a subgroup of $(\mathbb{Z}/r\mathbb{Z})^\times$ and H is generated by n and p. For notational convenience, let $t := \#H$. Also, $\text{ord}_r(n) > \log^2 n$ implies that $t > \log^2 n$.

To create the group G which will ultimately elicit a contradiction, we call upon the r^{th} cyclotomic polynomial, $\Phi_r(x) = \prod_{1 \leq k \leq r} (x - \xi_k)$, where ξ_k are the distinct r^{th} primitive roots of unity, for $k = 1, 2, \ldots, r$. $\Phi_r(x)$ divides $x^r - 1$ and factors into irreducibles over $\mathbb{F}_p$, the field of p elements. Because of the isomorphism of finite fields, we can view it as $\mathbb{F}_p = \mathbb{Z}/p\mathbb{Z}$. Select an irreducible factor $h(x)$. Its degree will be greater than one, because $\text{ord}_r(p) > 1$. Let G be the set of residues of polynomials in P, $\pmod{h(x), p}$. The set of all residues is $\mathbb{F} := (\mathbb{Z}/p\mathbb{Z})/(h(x))$, generated by $x, x + 1, x + \ldots, [q]_p$ and is the finite field of $p^{\deg((h(x)))}$ elements, and G is a multiplicative subgroup thereof.

We now find bounds on $\#G$.

Lemma 3.6. $\#G \geq \binom{t+q}{t-1}$.

Proof. This proof takes place in the field $\mathbb{F}$. Let $f(x), g(x) \in P$, $k \in I$. Can two polynomials of degree less than t map to the same element of G? Suppose they can. $h(x)$ divides $x^r - 1$ and k is introspective for $f(x)$ and $g(x)$ $\pmod{x^r - 1, p}$, so it follows that

$$f(x^k) \equiv g(x^k) \tag{8}$$

in the field $\mathbb{F}$. Consider their difference $\Delta(y) = f(y) - g(y)$. x^k will be a root of $\Delta(y)$, and because $k \in I$ was arbitrary, (8) holds for all $m \in H$. Because $h(x)$ is a factor of $\Phi_r(x)$ and

$(k, r) = 1$, each x^m is a primitive r^{th} root of unity. Thus every element of H corresponds to a different root of unity, so $\Delta(y)$ has $\#H = t$ distinct roots. Now,

$$\deg(\Delta(y)) = \begin{cases} \max\{\deg(f(y)), \deg(g(y))\} & \text{if } f(y) \neq g(y) \text{ in } (\mathbb{Z}/p\mathbb{Z})[y], \\ 0 & \text{if } f(y) = g(y) \text{ in } (\mathbb{Z}/p\mathbb{Z})[y]. \end{cases}$$

However, $f(x)$ and $g(x)$ are both of degree less than t by hypothesis. So $\Delta(y) \equiv 0$ which implies $f(x)$ and $g(x)$ are identical. Each distinct polynomial in P therefore maps to a different element of H.

We now turn our attention to $x + a$ for $a = 0, 1, \ldots, q$.

$q = \lfloor \sqrt{\varphi(r)} \log n \rfloor < \sqrt{r} \log n < r < p$, so $x + a$ is a different element of $\mathbb{F}$ for $0 \leq a \leq q$. Furthermore, none of these are congruent to zero in $\mathbb{F}$ because $\deg(h(x)) > 1$. Therefore, there are at least $\binom{(t-1)+(q+1)}{t-1} = \binom{t+q}{t-1}$ polynomials of degree less than t in G. □

$\#G$ can also be bounded above, provided n is not a power of p. This condition, as we will see, is critical.

Lemma 3.7. *If n is not a power of p, then $\#G \leq n^{\sqrt{t}}$.*

Proof. Define $\hat{I} \subset I$ as follows:

$$\hat{I} := \left\{ \left(\frac{n}{p}\right)^i p^j \ ; \ 0 \leq i, j \leq \lfloor \sqrt{t} \rfloor \right\}.$$

If n is not a power of p, then $\#\hat{I} = (\lfloor \sqrt{t} \rfloor + 1)^2 > t = \#G$. The pigeonhole principle implies that two elements of $\hat{I}$ are equivalent modulo r. Let these be k_1, k_2. Let $f(x) \in P$. Then,

$$[f(x)]^{k_1} \equiv f(x^{k_1}) \equiv f(x^{k_2}) \equiv [f(x)]^{k_2} \pmod{x^r - 1, p},$$

which implies $[f(x)]^{k_1} \equiv [f(x)]^{k_2}$ in $\mathbb{F}$. Hence $f(x) \in G$ is a root of $Q(y) = y^{k_1} - y^{k_2}$ in $\mathbb{F}$. By a similar argument to that used in Lemma 3.6, $Q(y)$ has at least $\#G$ distinct roots in $\mathbb{F}$.

$$\deg(Q(y)) = \max\{k_1, k_2\} \leq \left(\left(\frac{n}{p}\right) p \right)^{\lfloor \sqrt{t} \rfloor} \leq n^{\sqrt{t}},$$

therefore $\#G \leq n^{\sqrt{t}}$. □

These lemmas all culminate in the proof of the correctness of the algorithm.

Theorem 3.8. *AKS outputs* PRIME *if and only if n is prime.*

Proof. If n is prime, then AKS outputs PRIME, as proven in Lemma 3.1.

If n is composite, then from Lemma 3.6,

$$\begin{aligned}
\#G &\geq \binom{t+q}{t-1} \\
&\geq \binom{q+1+\lfloor\sqrt{t}\log n\rfloor}{\lfloor\sqrt{t}\log n\rfloor} && \because t > \sqrt{t}\log n \\
&\geq \binom{2\lfloor\sqrt{t}\log n\rfloor+1}{\lfloor\sqrt{t}\log n\rfloor} && \because q = \lfloor\sqrt{\varphi(r)}\log n\rfloor > \lfloor\sqrt{t}\log n\rfloor \\
&\geq \left(\frac{2\lfloor\sqrt{t}\log n\rfloor+1}{\lfloor\sqrt{t}\log n\rfloor}\right)^{\lfloor\sqrt{t}\log n\rfloor} && \because \binom{n}{k} > \left(\frac{n}{k}\right)^k \\
&= \left(2+\frac{1}{\lfloor\sqrt{t}\log n\rfloor}\right)^{\lfloor\sqrt{t}\log n\rfloor} \\
&= 2^{\log\left(\left(2+\frac{1}{\lfloor\sqrt{t}\log n\rfloor}\right)^{\lfloor\sqrt{t}\log n\rfloor}\right)} \\
&= 2^{\lfloor\sqrt{t}\log n\rfloor\log\left(\left(2+\frac{1}{\lfloor\sqrt{t}\log n\rfloor}\right)\right)} \\
&> 2^{\sqrt{t}\log n} && \because \lfloor\sqrt{t}\log n\rfloor > \lfloor\log^2 n\rfloor \geq 1 \\
&= n^{\sqrt{t}}.
\end{aligned}$$

So $\#G > n^{\sqrt{t}}$ which contradicts Lemma 3.7, unless n is a power of a p. In this case, Step 1 of AKS will output COMPOSITE. Therefore, AKS will output PRIME only if n is indeed prime. □

4 Time Complexity Analysis

The algorithm runs in time $\tilde{O}(\log^{10.5} n)$. After a few facts about computer arithmetic, we will be ready to prove this.

Let m be the number of bits (binary digits) of an integer. Addition and subtraction can be performed in $O(m)$. Multiplication can be performed in $O(m\log m\log\log m) = \tilde{O}(m)$ using the fastest known multiplication algorithm, the Fast Fourier Transform[9][1]. Modular reduction of integers can also be performed in $\tilde{O}(m)$. Given two polynomials of degree at most d with coefficients with at most m bits, their product can be computed in $\tilde{O}(md)$. Hence, reducing a polynomial $f(x) \pmod{x^r-1, n}$ can also be done in $\tilde{O}(md)$. Using exponentiation by squaring, raising to the power n takes at most $\log n$ steps.

With this knowledge, we analyze the time complexity of AKS.

[1]The Fast Fourier Transform relies on primitive roots of unity, and may be of interest to those with some knowledge of number theory.

Step 1. Perfect powers can be detected in $\tilde{O}(\log n)$ [7]. The idea behind this is calculating higher and higher roots using Newton's method, but to achieve this time bound, one must be precarious. This is not the bottleneck step though. A natural way permits this step in $\tilde{O}(\log^3 n)$.

Step 2. To find the least r such that $\operatorname{ord}_r(n) > \log^2 n$, check if $n^k \not\equiv 1 \pmod{r}$ for every $k \leq \log^2 n$. Increment r and try again until this is so. There are at most $O(\log^2 n)$ multiplications modulo r for each value of r, so $\tilde{O}(r \log^2 n)$ for these nested loops. Lemma 3.2 guarantees that $r \leq \lceil \log^5 n \rceil$, so Step 2 runs in $\tilde{O}(\log^7 n)$.

Step 3. At most r gcd computations. Using the Euclidean algorithm, each $\gcd(a, n)$ can be found in at most $5 \log_{10} a$ steps, where $a < n$ [8][2]. Again, given the size of r, this step runs in $\tilde{O}(\log^7 n)$.

Step 4. $O(\log n)$.

Step 5. Note that the totient of r is difficult to compute. If it is known, the for loop will have fewer iterations, but it does not affect the asymptotic running time. Each congruence involves multiplication of degree r polynomials with coefficients in $O(\log n)$. The for loop has $\lfloor \sqrt{\varphi(r)} \log n \rfloor$ iterations, so this step takes

$$\tilde{O}(\lfloor \sqrt{\varphi(r)} \log n \rfloor (r \log n)) = \tilde{O}(r \sqrt{\varphi(r)} \log^3 n) \in \tilde{O}(r \sqrt{r} \log^3 n) = \tilde{O}(\log^{10.5} n).$$

Step 5 is the bottleneck, so AKS runs in $\tilde{O}(\log^{10.5} n)$. As claimed, this is a completely deterministic primality test that runs in polynomial time.

5 Evolution

The algorithm presented above is not the original algorithm that Agrawal, Kayal, and Saxena devised. Their landmark paper PRIMES is in P was in preprint for a while as many improvements were made from other contributors. Below is a version of the algorithm dated back to August 2002 (publication of the algorithm above was in 2004) [4].

[2]This was shown by Gabriel Lamé in 1844 and is considered the first publication on computational complexity.

Algorithm AKS-2002

```
Input: $n \in \mathbb{N}$
  1. If $n$ is a perfect power, output COMPOSITE.
  2. Find prime $r < n$ such that for q the largest prime factor of $r - 1$,
     $q \geq 4\sqrt{r}\log n)$ and $(n^{\frac{r-1}{q}} \neq 1 \pmod{r})$.
  3. If $n \leq r$, output PRIME.
  4. For $a = 1$ to $2\sqrt{r}\log n$,
         if $(x-a)^n \neq x^n - a \pmod{x^r - 1, n}$, output COMPOSITE.
  5. Output PRIME.
```

This version of the algorithm might look a bit peculiar. In particular, Step 2 requires that r be prime, which seems demanding for a primality testing algorithm. Furthermore, the largest prime factor of $r - 1$ is needed. Factoring is also a hard problem (harder than identifying primes), so this part of the algorithm is a bit unsettling. Such r exist in the interval $[c_1 \log^6 n, c_2 \log^6 n]$, where c_1, c_2 can be found using results from analytic number theory on prime counting functions. Note that the for loop in Step 4 requires more than twice as many iterations to ensure the group G is large enough. The algorithm runs in $\tilde{O}(\log^{12} n)$.

The lower bound on $\#G$ given in 3.6 was discovered by Hendrik Lenstra Jr. It greatly simplified the proof of the correctness, and it improved the running time [4]. Other results arose to improve the running time to $\tilde{O}(\log^{9\frac{9}{11}} n)$ (Goldfeld), $\tilde{O}(\log^{7.5} n)$ (Fouvry), $\tilde{O}(\log^{7.49} n)$ (Baker and Harman) [6].

The running time was improved, once again, due to work by Hendrik Lenstra Jr. He and Carl Pomerance were able to improve the running time to $\tilde{O}(\log^6 n)$ by generating rings using Gaussian periods, instead of roots of unity. Their proof relies on recently developed results in both analytic and additive number theory [10]. Though a great acheivement, one cannot help but be impressed by the algorithm of Agrawal, Kayal, and Saxena presented in Section 3. Using scantly more than the tools acquired in a first course in abstract algebra, they solved an age-old problem in six steps.

References

[1] G.L. Miller, Riemann's hypothesis and test for primality, *J. Comput. Sys. Sci.* **13** (1976), 300-317.

[2] M.O. Rabin, Probabilistic algorithm for testing primality, *J. Number Theory* **12**, (1980) 128-138.

[3] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena, PRIMES is in P, *Ann. Math.* **160** (2004), no. 2, 781-793.

[4] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena, Primes is in P, Preprint (http://www.cse.iitk.ac.in/users/manindra/algebra/primality_original.pdf), August 2002.

[5] M. Nair, On Chebyshev-type inequalities for primes, *Amer. Math. Month.*, **80** (1982), no. 2, 126-129.

[6] Andrew Granville, It is easy to determine whether a given integer is prime, *Bull. Amer. Math. Soc.* **42** (2004), no. 1, 3-38.

[7] Daniel J. Bernstein, Detecting perfect powers in essentially linear time, *Math. of Computation* **67** (1998), no. 223, 1253-1283.

[8] William J. LeVeque, *Fundamentals of Number Theory*, Dover Publications, New York 1977.

[9] Jon Kleinberg and Éva Tardos, *Algorithm Design*, Pearson, New York 2006.

[10] H.W. Lenstra jr. and Carl Pomerance, Primality testing with Gaussian periods, Preprint (http://www.math.dartmouth.edu/∼carlp/aks041411.pdf), April 2011.