18.783 Elliptic Curves **Spring 2013**
Problem Set #6 **Due: 04/02/2013**

Description

These problems are related to the material covered in Lectures 11-13. As usual, the first person to spot each non-trivial typo/error will receive one point of extra credit.

Instructions: Solve Problem 1 and any **two** of Problems 2-4. Then do Problem 5, which is a survey. **Late problem sets will lose one point for each hour they are late**.

Problem 1. Subexponential bounds (20 points)

This is an easy problem designed to familiarize you with subexponential complexity bounds. The subexponential complexity bounds most commonly used have the form

$$L_N[\alpha, c] := \exp\Big(\big(c + o(1)\big)(\log N)^\alpha (\log\log N)^{1-\alpha}\Big),$$

where $0 \le \alpha \le 1$ and $c > 0$. The notation $o(1)$ denotes any function $\epsilon(N)$ whose absolute value converges to 0 as $N \to \infty$. Thus $L_N[\alpha, c]$ should really viewed as a set of functions. A function $f(N)$ belongs to the set $L_N[\alpha, c]$ if and only if

$$\lim_{N\to\infty} \frac{\log f(N)}{(\log N)^\alpha (\log\log N)^{1-\alpha}} = c.$$

To get a sense of how these bounds grow with N, and how to compare consider the following table, in which the $o(1)$ term is assumed to be 0 and $n = \log_2 N$.

n	n^5	$L_N[1/4, 1]$	$L_N[1/2, 1]$	$L_N[1/2, \sqrt{2}]$	$n^2 L_N[1/2, \sqrt{2}]$	$N^{1/4}$.
64	1.1×10^{9}	1.1×10^{3}	4.3×10^{5}	9.3×10^{7}	3.8×10^{11}	6.6×10^{4}
128	3.4×10^{10}	1.3×10^{4}	4.6×10^{8}	1.8×10^{12}	2.9×10^{16}	4.3×10^{9}
256	1.1×10^{12}	2.8×10^{5}	1.5×10^{13}	4.2×10^{18}	2.7×10^{23}	1.8×10^{19}
512	3.5×10^{13}	1.3×10^{7}	6.7×10^{19}	1.1×10^{28}	2.9×10^{33}	3.4×10^{38}
1024	1.1×10^{15}	1.6×10^{9}	4.4×10^{29}	8.4×10^{41}	8.8×10^{47}	1.2×10^{77}
2048	3.6×10^{16}	6.1×10^{11}	1.2×10^{44}	2.2×10^{62}	9.2×10^{68}	1.3×10^{154}

1. Simplify the following expressions, in which $0 < \alpha, \beta < 1$ and $c, d > 0$, and $p(x)$ denotes a polynomial of degree k. Interpret sums and products of complexity bounds (sets of functions) in the obvious way, e.g. $S + T$ is the set of all functions $s + t$ with $s \in S$ and $t \in T$.

(a) $L_N[0, c]$ and $L_N[1, c]$
(b) $L_N[\alpha, c] + L_N[\beta, d]$
(c) $L_N[\alpha, c]L_N[\beta, d]$
(d) $L_N[\alpha, c]p(\log N)$
(e) $p(L_N[\alpha, c])$
(f) $L_{p(N)}[\alpha, c]$
(g) $L_{L_N[\alpha,c]}[\beta, d]$

2. For each of the following pairs of complexity bounds $A(N)$ and $B(N)$ representing sets of functions A and B, indicate which of the following holds: (a) $A \subsetneq B$, (b) $B \subsetneq A$, (c) $A = B$, (d) $A \cap B = \emptyset$, or (e) none of the above.

a) $L_N[\alpha, c]$ and $O(L_N[\alpha, c])$.
b) $L_N[\alpha, c]$ and $L_N[\beta, d]$ with $\alpha > \beta$.
c) $L_N[\alpha, c]$ and $L_N[\alpha, d]$ with $c > d$.
d) $L_N[\alpha, c]$ and $O\Big(\exp\big(c(\log N)^\alpha\big)\Big)$.
e) $L_N[\alpha, c]$ and $L_{\exp(\log_2 N)}[\alpha, c]$.

3. The Canfield-Erdős-Pomerance theorem states that $\psi(x, x^{1/u}) = xu^{-u+o(u)}$ holds uniformly for $u < (1-\epsilon)\log x/\log\log x$. Using this, prove that

$$\frac{1}{x}\psi(x, L_x[1/2, c]) = L_x[1/2, 1/2c]^{-1}.$$

Problem 2. ECM second stage (40 points)

The elliptic curve factorization method (ECM) can be extended to incorporate a *second stage* that substantially improves its practical performance. In this problem you will analyze the benefit of this second stage, and, as a side benefit, derive a generic algorithm to compute the order of a group element using $o(\sqrt{N})$ group operations.

Given an integer N to be factored, a bound M on the largest prime divisor of N one hopes to find, and a smoothness bound $B_1 = L_M[1/2, 1/\sqrt{2}]$, the ECM algorithm selects random elliptic curves $E/\mathbb{Q}$ with a known point P of infinite order and computes the scalar multiple $mP = (x_m : y_m : z_m)$, working with projective coordinates reduced modulo N. The integer $m = \prod \ell_i^{e_i}$ is a product of prime powers that satisfy $\ell_i^{e_i} \le (\sqrt{M}+1)^2 \le \ell_i^{e_i+1}$, ranging over all primes $p_i \le B_1$. The goal is to find a curve for which $\gcd(z_m, N)$ is non-trivial (we actually check $\gcd(z_{m_i}, N)$ for the partial products $m_i = \prod \ell_i^{e_i}$ as we go).

But suppose that, as usually happens, $\gcd(z_m, N) = 1$. Let us assume that N has a prime factor $p \le M$. We know that $\#E(\mathbb{F}_p)$ is not B_1-smooth, meaning that it has a prime factor $q > B_1$, but suppose that q is the *only* prime factor of $\#E(\mathbb{F}_p)$ greater than B_1.[1] Then the point $Q = mP$ must have order q as an element of $E(\mathbb{F}_p)$. Provided q is not too large, say, $q \le B_2$ for some bound $B_2 \approx B_1^2$, then we can try to "compute" the order of mP in $E(\mathbb{F}_p)$ using a baby-steps giant-steps search up to the bound B_2. This is not as simple as it sounds: we don't know p so we must work modulo N while checking for collisions modulo p, but there is an efficient algorithm for detecting collisions [3, §3]. The details of this algorithm do not concern us here, we simply want to consider the potential speedup we might gain from such a *second stage*.

If the prime factors of an integer n are all smaller than y, and all but one of them is smaller than z, then n is said to be *semismooth* with respect to y and z. The function $\psi(x, y, z)$ counts the number of such integers less than or equal to x. We are interested in the quantity $\frac{1}{M}\psi(M, B_2, B_1)$. Under the heuristic assumption that the orders of random elliptic curves over a finite field are about as likely to be semismooth as integers of similar size, this is the probability that our algorithm will be able to find an integer n for which $nP \equiv 0 \bmod q$, either in the first or second stage (we aren't guaranteed to succeed if this happens, we also need $nP \not\equiv 0 \bmod N$, but this is very likely to be true).

Let $B_1 = M^{1/u}$. We saw in class that, under our heuristic assumption, the expected running time of ECM with just a single stage is proportional to

$$M^{1/u}(\psi(M, M^{1/u})/M)^{-1}\mathsf{M}(\log N). \tag{1}$$

[1] As usual, we abuse notation by writing $E(\mathbb{F}_p)$ for the group of $\mathbb{F}_p$-rational points on the reduction of the elliptic curve $E/\mathbb{Q}$ modulo the prime p, where E has good reduction at p.

Using the Canfield-Erdő s-Pomerance bound $\psi(x, x^{1/u})/x = u^{-u+o(u)}$, we found that we should pick $u = \sqrt{2\log M/\log\log M}$ and obtained the bound $L[1/2, \sqrt{2}]\mathsf{M}(\log N)$. But this is a very rough approximation and we ignored several factors logarithmic in M along the way (these are hidden in the $o(1)$ term in the subexponential notation).

We can get a much more precise estimate by using the Dickman function $\rho(u)$ to approximate $\psi(x, x^{1/u})/x$. The Dickman function $\rho(u)$ is defined via the differential delay equation

$$\rho'(u) = -\rho(u-1)/u,$$

with $\rho(u) = 1$ for $0 \le u \le 1$. Asymptotically $\rho(u) = \psi(x, x^{1/u})/x + o(1)$, and in practice $\rho(u)$ is very close to $\frac{1}{x}\psi(x, x^{1/u})$ for x and u in the range we are interested in. Sage has a built-in function `dickman_rho(u)` that computes a good numerical approximation to $\rho(u)$. See [2, §1] if you want to know more about $\rho(u)$ and its relation to $\psi(x, y)$.

To minimize (1) it suffices to thus suffices to minimize

$$M^{1/u}/\rho(u). \tag{2}$$

1. Using Newton's method, write a simple function in Sage that approximates the value of u that minimizes (2) for a given value of M (accurate to at least 3 decimal places).

For the sake of simplicity, let us suppose that $B_2 = B_1^2 = M^{2/u}$ and that the second stage has a running time approximately equal to that of the first. Then the expected running time of ECM with a BSGS second stage is heuristically proportional to

$$2M^{1/u}(M/\psi(M, M^{2/u}, M^{1/u})) \cdot \mathsf{M}(\log N), \tag{3}$$

with the same constant of proportionality as in our single stage analysis. In fact, we should optimally spend asymptotically slightly *less* time on the second stage than the first; this would allow us to save the factor of 2 in (3). You will prove below that this can actually be achieved using $B_2 = B_1^2$ if we modify the baby-steps giant-steps search appropriately.

Analogous to $\rho(u)$, Bach and Peralta [1] define the semismooth probability function

$$G(a, b) = \lim_{x\to\infty} \frac{1}{x}\psi(x, x^b, x^a)$$

(note the reverse order of a and b). The function $G(a, b)$ can be numerically approximated using the Dickman function in terms of the function $F(\alpha) = \rho(1/\alpha)$ as

$$G(\alpha, \beta) = F(\alpha) + \int_\alpha^\beta F\left(\frac{\alpha}{1-t}\right)\frac{dt}{t}.$$

By numerically approximating $G(a, b)$ we can determine a suitable choice of u to minimize the quantity

$$M^{1/u}/G(1/u, 2/u). \tag{4}$$

This calculation is a bit time consuming, so a table of optimal u values for $M = 2^k$ with $k = 10, 20, 30, \ldots, 200$ has been prepared for you and can be found in the Sage worksheet 18.783 Problem Set 4 Problem 2.sws, which also implements a function `G(a,b)`that approximates $G(\alpha, \beta)$ using $\rho(u)$.

2. Use the algorithm you implemented in part 1 to generate a similar table of optimal u values that minimize (2). Then, for $k = 20, 40, 60, \ldots, 200$ compute $M^{1/u_1}/\rho(u_1)$ and $M^{1/u_2}/G(1/u_2, 2/u_2)$, with $M = 2^k$ and u_1 chosen to minimize the first quantity and u_2 chosen to minimize the second. List these values and their ratio in a table.

The ratios express the speedup we might hope to gain by using a second stage. You should find that the speedup is clearly increasing with k, implying that it is asymptotically better than a constant factor. Nevertheless, the second stage does not improve the subexponential complexity bound, which ignores even polynomial factors of $\log M$.

3. Prove that the heuristic expected running time of ECM with a second stage is still $L_M[1/2, \sqrt{2}]\mathsf{M}(\log N)$, the same as with just one stage. Based on the data in your table from part 2, estimate what the asymptotic speedup is as a function of $\log M$.

Let $Q = mP$ be the point obtained after an unsuccessful first stage. When using baby-steps giant-steps to implement the second stage we can take advantage of the fact that, for any prime divisor $p \le M$ of N, in the group $E(\mathbb{F}_p)$ the reduction of the point Q cannot have order divisible by any prime $p_i \le B_1$. Indeed, the second stage will succeed only in the case where Q has prime order $q \in (B_1, B_2]$ in $E(\mathbb{F}_p)$.

This means that our baby-steps giant-steps search only needs to check $O(B_2/\log B_2)$ distinct multiples of Q, those corresponding to prime values. In principle, this could potentially be achieved with just $\sqrt{B_2/\log B_2}$ group operations, but it is not obvious how to do this. At a minimum, we can certainly avoid checking multiples of small primes $2, 3, 5, \ldots, \ell$ whose product t is substantially less than $\sqrt{B_2}$, for the sake of concreteness, let's say $t \approx B_2^{1/4}$. We should then compute baby steps of the form iQ with $\gcd(i, t) = 1$ for all $1 \le i \le r$ for some multiple r of t, followed by giant steps of the form jrQ for $1 \le j \le s$, where $rs \ge B_2$.

4. Explain how to choose r and s so that the number of baby steps and giant steps are approximately equal, and give a tight asymptotic bound on the total number of steps in terms of B_2. You may use the Prime Number Theorem and standard facts it implies, such as $\sum_{p\le x} \log p \sim x$ and $\sum_{p\le x} \frac{1}{p} = \log\log x + O(1)$.[2]

5. Now forget about ECM. Using your answer to part 4, describe a generic algorithm to compute the order of an element $\alpha \in G$ given an integer $N > |\alpha|$ that uses $o(\sqrt{N})$ group operations (the order of α may be prime or composite).

6. Modify the algorithm in part 5 to not require N, so that it computes $|\alpha|$ using $o(\sqrt{|\alpha|})$ group operations.

7. Computing $|\alpha|$ is equivalent to computing the discrete logarithm of the identity with respect to α. Explain why your algorithm does not contradict Shoup's $\Omega(\sqrt{p})$ generic lower bound for the discrete logarithm problem in the case that $|\alpha| = p$ is prime.

Problem 3. ECPP (40 points)

Let us define an *elliptic curve primality proof* (ECPP) for p as a sequence of *certificates* $C_1, C_2, \ldots, C_k$, where each certificate C_i is of the form $(p_i, A_i, B_i, x_i, y_i, p_{i+1})$ with $p_1 = p$ and $p_{k+1} < (\log p)^4$. In each certificate C_i, the primes p_i and p_{i+1} satisfy

$$(\sqrt[4]{p_i} + 1)^2 < p_{i+1} < (\sqrt{p_i} + 1)^2/2,$$

and $P_i = (x_i, y_i)$ is a point of order p_{i+1} on the elliptic curve $y^2 = x^3 + A_i x + B_i$ over $\mathbb{F}_{p_i}$.

[2] The second fact doesn't require the Prime Number Theorem, it was proven earlier by Mertens.

1. Generate a random 100-bit prime using the Sage function `random_prime` and construct an elliptic curve primality proof for it. Your proof should not require more than half a dozen certificates.

2. Analyze the complexity of verifying an elliptic curve primality proof. Express your answer solely in terms of $n = \log p$ (so assume a worst-case certificate).

3. Analyze the asymptotic complexity of constructing an elliptic curve primality proof using the Goldwasser-Kilian algorithm given in class, under the heuristic assumption that the orders of random elliptic curves over $\mathbb{F}_p$ have factorizations comparable to random integers in the interval $[p, 2p]$. Assume that trial division and the Miller-Rabin test are used for attempted factorizations. Use an $O(n^5 \log\log n)$ complexity bound for point-counting via Schoof's algorithm.

4. Now suppose that you want to construct an elliptic curve primality proofs that can always be verified in $O(n\mathsf{M}(n))$ time, where $n = \log p$. Under the heuristic assumption above, give a probabilistic algorithm for constructing such a proof whose expected running time is bounded by $L_p[\alpha, c]$, using the tightest values of α and c that you can.

Problem 4. Pomerance proofs (40 points)

A *Pomerance proof* is a special form of an elliptic curve primality proof that involves just a single certificate (p, A, x_0, k) and uses a Montgomery curve $By^2 = x^3 + Ax^2 + x$ over $\mathbb{F}_p$ on which there is a point (x_0, y_0) of point of order $2^k > (\sqrt[4]{p} + 1)^2 \geq 2^{k-1}$. Note that neither the y-coordinate nor B is needed to verify the certificate (no matter what $x_0^3 + Ax_0^2 + x_0$ is, there exists a nonzero B and a y_0 that will work and the verifier does not need to know what they are), but the verifier should check that $A^2 \not\equiv 4 \bmod q$ for all primes q dividing p, to ensure that the curve is not singular.

Every prime p has a Pomerance proof, but for a general prime p no efficient algorithm is known for finding one. In this problem you will develop a very efficient algorithm to construct a Pomerance proof for primes of a special form. Let E be the elliptic curve defined by $y^2 = x^3 + 8$.

1. Using the formula $\#E(\mathbb{F}_p) = p + 1 + \sum_{x \in \mathbb{F}_p} \left(\frac{x^3+8}{p}\right)$, prove that for every odd prime $p \equiv 2 \bmod 3$ we have $\#E(\mathbb{F}_p) = p + 1$.

2. Prove that for any prime $p \equiv 11 \bmod 12$ the curve $E/\mathbb{F}_p$ can be put in Montgomery form $By^2 = x^3 + Ax^2 + x$. Give a deterministic algorithm that computes A and B in time $O(n\mathsf{M}(n))$, where $n = \log p$.

3. Give a probabilistic algorithm to construct a Pomerance proof for primes of the form $p = 3 \cdot 2^m c - 1$, where c is odd and $2^m > (\sqrt[4]{p} + 1)^2$, and analyze its complexity. Be sure to address the fact that the algorithm you gave in part 2 assumes that p is prime, but now it must also handle composite values of p.

4. Implement your algorithm and use it to construct a Pomerance proof for a prime of the form $p = 2^k \cdot 3^m - 1$ that is greater than 2^{1000}. Be sure to format you answer so that all of the digits in the certificate you construct fit on the page. You may wish to use trial division by small primes to eliminate obviously composite values of p before attempting to construct a primality proof, but it is not worth using a Miller-Rabin test to detect composites; explain why this is so.

Problem 5. Survey

Complete the following survey by rating each of the problems you attempted on a scale of 1 to 10 according to how interesting you found the problem (1 = "mind-numbing," 10 = "mind-blowing"), and how difficult you found the problem (1 = "trivial," 10 = "brutal"). Also estimate the amount of time you spent on each problem to the nearest half hour.

	Interest	Difficulty	Time Spent
Problem 1			
Problem 2			
Problem 3			
Problem 4			

Also, please rate each of the following lectures that you attended, according to the quality of the material (1="useless", 10="fascinating"), the quality of the presentation (1="epic fail", 10="perfection"), the pace (1="way too slow", 10="way too fast", 5="just right") and the novelty of the material (1="old hat", 10="all new").

Date	Lecture Topic	Material	Presentation	Pace	Novelty
3/14	Discrete Logarithm Problem (part 2)				
3/19	Elliptic curve factorization method				
3/21	Elliptic curve primality proving				

Please feel free to record any additional comments you have on the problem sets or lectures, in particular, ways in which they might be improved.

References

[1] E. Bach and R. Peralta, *Asymptotic semismoothness probabilities*, Mathematics of Computation **65** (1998) 1701–1715.

[2] A. Granville, *Smooth numbers, computational number theory and beyond*, in Algorithmic Number Theory: Lattices, Number Fields, Curves and Cryptography (MSRI Workshop), MSRI Publications **44** (2008), 267–324.

[3] P. Zimmermann and B. Dodson, *20 years of ECM*, Algorithmic Number Theory 7th International Symposium (ANTS VII), LNCS 4076 (2006), 525–542.

18.783 Elliptic Curves
Spring 2013