Subexponential algorithm for discrete logarithm

Mathematical Cryptography, Lectures 22 - 23

Contents

Subexponential complexity

The subexponential algorithm for discrete logarithm (SEDL) bilds on $y-$smooth integers and on linear algebra over the field $\mathbb{Z}_p$. Therefore, the algorithm SEDL works only for subgroups of $\mathbb{Z}_p^*$.

- Exponential complexity: $O(n) = O(2^{\mathrm{len}(n)})$
- Subexponential complexity: $O(2^{f(\mathrm{len}(n))})$, where $f(x) \in o(x)$, i.e. $\lim_{x\to\infty} \frac{f(x)}{x} = 0$.
  The algorithm SEDL has complexity $O(2^{c\sqrt{\mathrm{len}(n)\,\mathrm{len}(\mathrm{len}(n))}})$.
  For example, for $n = 2^{256}$ it gives $O(2^{\sqrt{256\cdot 8}}) \doteq O(2^{47})$.

Smooth numbers

Definition

Let $y \geq 0$ be a real number. An integer $m \geq 1$ is called to be *$y-$smooth* if all prime divisors of m are less than y.

Let $0 \leq y \leq x$ be real numbers. Let us denote the number of all $y-$smooth numbers up to x as $\Psi(y, x)$.

Examples

Numbers 4, 27, 24, $9216 = 3^2 \cdot 2^{10}$ are 3$-$smooth.

$\Psi(2, 10) = 4$ since $1, 2, 4, 8$ are all 2$-$smooth numbers up to 10.
$\Psi(3, 10) = 7$ since $1, 2, 3, 4, 6, 8, 9$ are 3-smooth numbers up to 10.

Obviously $\Psi(n, n) = n$ for any $n \in \mathbb{N}$.

Smooth numbers

Theorem 1

Let $y = y(x)$ satisfy $\lim_{x\to\infty} \frac{\ln(x)}{y} = 0$ and $\lim_{x\to\infty} \frac{\ln(y)}{\ln(x)} = 0$.
Then

$$\Psi(y,x) \geq x\, e^{(-1+o(1))\frac{\ln(x)}{\ln(y)}\, \ln(\ln(x))}$$

Note

Recall that $f \in o(g)$ in case $\lim_{x\to\infty} \frac{f(x)}{g(x)} = 0$.
The symbol $o(1)$ represents a function $f(x)$ for which $\lim_{x\to\infty} f(x) = 0$.

Smooth numbers

Theorem 2

Let $y = y(x)$ satisfy $y \in \Omega(\ln(x)^{1+\epsilon})$ for some $\epsilon > 0$ and $\lim_{x\to\infty} \frac{\ln(y)}{\ln(x)} = 0$. Then

$$\Psi(y,x) = x\, e^{(-1+o(1))\frac{\ln(x)}{\ln(y)}\ \ln(\frac{\ln(x)}{\ln(y)})}$$

Note

Smooth numbers play an important role in the following subexponential algorithms. We will need estimates of how many they are for determining an expected running time of the algorithms.

Linear algebra over a field

A linear algebra over the field $\mathbb{Z}_p$ works just like over $\mathbb{R}$.

Linear space over a field

A *linear space* over the field $(T, +, \cdot)$ is the set L together with addition $\oplus : L \times L \to L$ and numerical multiplication $\boxdot : T \times L \to L$ such that:

- $(L, \oplus)$ is an abelian group with an identity element $\bar{o}$;
- For all $\alpha, \beta \in T$ and all $\bar{u}, \bar{v} \in L$:
  - $\alpha \boxdot (\bar{u} \oplus \bar{v}) = (\alpha \boxdot \bar{u}) \oplus (\alpha \boxdot \bar{v})$
  - $(\alpha + \beta) \boxdot \bar{u} = (\alpha \boxdot \bar{u}) \oplus (\beta \boxdot \bar{u})$
  - $(\alpha \cdot \beta) \boxdot \bar{u} = \alpha \boxdot (\beta \boxdot \bar{u})$
  - $1 \boxdot \bar{u} = \bar{u}$

Elements of L are called vectors, elements of T scalars.

Linear algebra over a field

Linear space over a field

- A *subspace* of the linear space L is a nonempty subset $P \subseteq L$ that is closed to addition and numerical multiplication.
- A *basis* of the linear subspace P is its linearly independent subset $B = \{\bar{b}_1, \dots, \bar{b}_n\}$ which generates all the subspace P, so $\bar{u} \in P$ just if $\bar{u} = \sum_{i=1}^{n} a_i \bar{b}_i$, where the n-tuple of coefficients $(a_1 \dots a_n) \in T^{\times n}$ is uniquely determined.
- The n-tuple of coefficients is called the *coordinates* of the vector $\bar{u}$ with respect to the ordered basis B.
- A number of elements of any basis of the subspace P is called the *dimension* of the subspace P, here $\dim P = n$.

Linear algebra over a field

Linear space over a field

- The vectors $\bar{u}_1, \ldots, \bar{u}_m$ are *linearly dependent* if there exist coefficients $c_1, \ldots, c_m \in T$ with at least one $c_i \neq 0$ such that $c_1\bar{u}_1 + \ldots c_m\bar{u}_m = \bar{o}$ (there exists a non-trivial linear combination of the vertors that equals to the zero vector).
- Let L be a linear space of dimension n, then any $m > n$ vectors are linearly dependent.
- In particular, the set $T^{\times n}$ of all $n-$tuples over the field T forms a linear space of the dimension n, so any $n+1$ vectors here form a linearly dependent set.

Linear algebra over the field

Systems of linear equations over a field

- The *Gaussian elimination algorithm* works over any field T, instead of dividing equations by their pivots, it uses multiplication by inverses of their pivots. (In the field T, every non-zero element has an inverse element.)
- Note: Over a ring (over $\mathbb{Z}_n$, where n is not a prime), Gaussian elimination does not work in general because the leading pivots need not to be invertible.
- The system of linear equations can have one solution, or no solution, or $|T|^k$ different solutions, where k is the number of variables we are allowed to choose arbitrarily in T.

Linear algebra over a field

Systems of linear equations over a field

- All solutions of the homogeneous system $\mathbb{A}\bar{x}^T = \bar{o}^T$ form a subspace in $T^{\times n}$ of dimension k, where k is the number of variables we are allowed to choose arbitrarily in T.
- Every solution of the system of equations $\mathbb{A}\bar{x}^T = \bar{b}^T$ is the sum of a partial solution of this system and some solution of the associated homogeneous system.

Linear algebra over a field

Matrix calculus over a field

- The matrix calculus over a field works just like over reals $\mathbb{R}$ - we can define a determinant and a rank of a matrix, or calculate inverse matrices.
- Matrix calculus over a ring can be done with some specialities - e.g. the row rank need not be equal to the column rank (since Gaussian elimination does not work).
- A determinant of a matrix can be defined over a ring, invertible matrices are just those matrices that have an invertible determinant.

SEDL algorithm

Representation of an element

Let G be a cyclic group of order n with a generator a, and let $b \in G$. *Representation of the element* $g \in G$ with respect to the generator a and the element b is any pair of numbers $(s, t) \in \mathbb{Z}_n \times \mathbb{Z}_n$ such that $g = a^s b^t$ in G.
Moreover, if $t \in \mathbb{Z}_n^*$, then the representation is non-trivial.

Proposition

1. For each $t \in \mathbb{Z}_n$ there exists just one $s \in \mathbb{Z}_n$ such that (s, t) is a representation of g with respect to the generator a and the element b.
2. If a non-trivial representation (s, t) of 1 with respect to a and b is known, then discrete logarithm can be computed: $\mathrm{dlog}_a(b) = -st^{-1}$ in $\mathbb{Z}_n$.

SEDL algorithm

Subexponential algorithm for discrete logarithm (SEDL)

Input: p, q, a, b, q,
where $G = \langle a \rangle$ is a subgroup of order q in the group $\mathbb{Z}_p^*$,
p, q are primes,
a is a generator of G, $b \in G$.

Moreover, suppose that $|\mathbb{Z}_p^*| = p - 1 = qm$, where $q \nmid m$.
(We'll discuss later how to proceed without this assumption.)

Output: $x = \mathrm{dlog}_a(b)$, or a report "failure".

The algorithm SEDL looks for a non-trivial representation of 1 with respect to a and b. If it finds any, it computes the discrete logarithm from it.

Algorithm SEDL

Proposition

Let $|Z_p^*| = qm$, where p, q are primes and $q \nmid m$, and let G be a subgroup of order q and H be a subgroup of order m in $\mathbb{Z}_p^*$.
Then $\mathbb{Z}_p^* = G \dot{\times} H$ is an internal direct product of G and H, so

- $G \cap H = \{1\}$,
- $GH = \mathbb{Z}_p^*$.

Or, $G \times H \simeq \mathbb{Z}_p^*$ and each element $z \in \mathbb{Z}_p^*$ can be written uniquely in the form $z = gh$, where $g \in G$ and $h \in H$.

Proposition

Let $|\mathbb{Z}_p^*| = qm$, where p is a prime, and let H be a subgroup of order m in the group $\mathbb{Z}_p^*$. Then for any element $z \in \mathbb{Z}_p^*$ is $z^q \in H$.

Algorithm SEDL

First stage of the algorithm SEDL

We use $y-$smoothness, we will discuss an appropriate choice of the parameter $y < p$ later on.
Let $p_1, \ldots, p_k$ be all primes up to y, so there are k many of them.

We find $(k+1)$ $y-$smooth numbers from $\mathbb{Z}_p^*$ by random, each of the form $a^{s_i} b^{t_i} h_i$, where $a^{s_i} b^{t_i} = g_i \in G$, $h_i \in H$.

We do this for every $1 \leq i \leq k+1$ as follows:

- choose randomly $s_i, t_i \in \mathbb{Z}_q$ and $\tilde{h}_i \in \mathbb{Z}_p^*$, count $h_i = \tilde{h}_i^q \in H$
- verify by trial division if $z_i = a^{s_i} b^{t_i} h_i$ in $\mathbb{Z}_p^*$ is $y-$smooth, i.e. whether $z_i = p_1^{e_{i_1}} \cdot \ldots \cdot p_k^{e_{i_k}}$ in $\mathbb{Z}$ where $0 < z_i < p$, then $a^{s_i} b^{t_i} h_i = p_1^{e_{i_1}} \cdot \ldots \cdot p_k^{e_{i_k}}$ in $\mathbb{Z}_p^*$
- if not, then repeat the random choice

Algorithm SEDL

First stage of the algorithm SEDL

Remark:

It would be sufficient to find randomly $(k+1)$ $y-$smooth numbers from the subgroup G, each of the form $a^{s_i}b^{t_i}$, but we would not be able to estimate the expected time for searching because we don't know how many $y-$smooth numbers are in the subgroup G.

We can only estimate how many $y-$smooth numbers are up to p, so in $\mathbb{Z}_p^*$, and because of this we choose numbers of the form $a^{s_i}b^{t_i}h_i = g_ih_i \in \mathbb{Z}_p^*$.

Algorithm SEDL

Second stage of the algorithm SEDL

We use linear algebra over the field $\mathbb{Z}_q$, where $q = |G|$.
We know that q is prime, therefore $\mathbb{Z}_q$ is a field.

In the first stage, we have found $(k+1)$ equalities of shape:

$$a^{s_i} b^{t_i} h_i = p_1^{e_{i_1}} \cdot \cdot p_k^{e_{i_k}} \text{ in } \mathbb{Z}_p^*$$

For each $1 \le i \le k+1$ we consider the $k-$tuple of exponents $\bar{v}_i = (e_{i_1}, \ldots, e_{i_k})$ as a vector over the field $\mathbb{Z}_q$ for now.
The set $\mathbb{Z}_q^{\times k}$ of all $k-$tuples over $\mathbb{Z}_q$ forms a linear space of thedimension k. So our $(k+1)$ vectors must be linearly dependent, or there exists a non-trivial linear combination of them which equals to the zero vector.

Algorithm SEDL

Second stage of the algorithm SEDL

There exist coefficients $c_1, \ldots, c_{k+1} \in \mathbb{Z}_q$, not all zero, such that

$$c_1 \bar{v}_1 + \ldots + c_{k+1} \bar{v}_{k+1} = \bar{o} = (0, \ldots, 0) \text{ in } \mathbb{Z}_q^{\times k}.$$

If we look at this combination over $\mathbb{Z}$, then all the components of the result vector are divisible by q.

$$c_1 \bar{v}_1 + \ldots + c_{k+1} \bar{v}_{k+1} = (e_1, \ldots, e_k) \text{ in } \mathbb{Z}^{\times k}, \ q \mid e_i \text{ for each } i.$$

We find the coefficients $c_1, \ldots, c_{k+1}$ using Gaussian elimination, which works over the field $\mathbb{Z}_q$.
(We will solve a homogeneous system of k equations for $(k+1)$ variables over $\mathbb{Z}_q$. We just need to find one non-trivial solution.)

Algorithm SEDL

Second stage of the algorithm SEDL

Consider again $(k+1)$ equalities $a^{s_i} b^{t_i} h_i = p_1^{e_{i_1}} \cdot \ldots \cdot p_k^{e_{i_k}}$ in $\mathbb{Z}_p^*$ from the first stage. If we power each $i-$th equality to the corresponding c_i and multiply all the equalities by each other, we get the equality:

$$a^s b^t h = p_1^{e_1} \cdot \ldots \cdot p_k^{e_k} \text{ in } \mathbb{Z}_p^*,$$

where $s = \sum_{i=1}^{k+1} c_i s_i, \; t = \sum_{i=1}^{k+1} c_i t_i$ in $\mathbb{Z}_q, \; h = \prod_{i=1}^{k+1} h_i^{c_i}$ in $\mathbb{Z}_p^*$.

Not all c_i are zero in $\mathbb{Z}_q$, so there could be $s \neq 0$ and $t \neq 0$.

Moreover, we know that $q \mid e_i$, thus $p_i^{e_i} \in H$ for each i.

Algorithm SEDL

Second stage of the algorithm SEDL

Finaly, we have the equality

$$a^s b^t = h^{-1} p_1^{e_1} \cdot \ldots \cdot p_k^{e_k} \text{ in } \mathbb{Z}_p^*,$$

where the element on the left is from the subgroup G and the element on the right is from the subgroup H.

But since $G \cap H = \{1\}$ (see the assumption), this element must be equal to 1. We have found a representation of 1 with respect to the generator a and the element b,

$$a^s b^t = 1 \text{ in } G \subseteq \mathbb{Z}_p^*.$$

If $t \neq 0$, we compute $\mathrm{dlog}_a(b) = -st^{-1}$ in $\mathbb{Z}_q$.
If $t = 0$, we report a failure.

Algorithm SEDL

- for $i \leftarrow 1$ to $k+1$ do
  - repeat
    - choose $s_i, t_i \overset{\not{q}}{\leftarrow} \mathbb{Z}_q$, $\tilde{h}_i \overset{\not{q}}{\leftarrow} \mathbb{Z}_p^*$ at random
    - $h_i \leftarrow \tilde{h}_i^q$, $z_i \leftarrow a^{s_i} b^{t_i} h_i$ in $\mathbb{Z}_p$
    - test if z_i is $y-$smooth (trial division)
  - until $z_i = p_1^{e_{i_1}} \cdot \cdot p_k^{e_{i_k}}$ for some $e_{i_1}, \ldots, e_{i_k} \in \mathbb{Z}$
  - $\bar{v}_i \leftarrow (e_{i_1}, \ldots, e_{i_k})$ in $\mathbb{Z}^{\times k}$ enddo
- apply Gaussian elimination over $\mathbb{Z}_q$ to find $c_1, \ldots, c_{k+1} \in \mathbb{Z}_q$, not all zero, such that $c_1\bar{v}_1 + \ldots + c_{k+1}\bar{v}_{k+1} = (0, \ldots, 0)$ in $\mathbb{Z}_q^{\times k}$
- $s \leftarrow \sum_{i=1}^{k+1} c_i s_i$, $t \leftarrow \sum_{i=1}^{k+1} c_i t_i$ in $\mathbb{Z}_q$
- if $t = 0$ in $\mathbb{Z}_q$
  - then output "failure"
  - else $x \leftarrow (-st^{-1})$ in $\mathbb{Z}_q$ and output x endif

Algorithm SEDL

Example

$G = \langle 4 \rangle$ is a subgroup of order 11 in the group $\mathbb{Z}_{23}^*$, $|\mathbb{Z}_{23}^*| = 2 \cdot 11$, so $H = \{\pm 1\}$. Count $\mathrm{dlog}_4(12)$ in $\mathbb{Z}_{23}^*$ by SEDL and choose the parameter of smoothness $y = 4$.
(Note: $12^{11} = 1$ in $\mathbb{Z}_{23}^*$, so $12 \in G$ and $\mathrm{dlog}_4(12)$ is defined.)

- Stage 1 - we calculate in $\mathbb{Z}_{23}^*$, randomly we get the equations:
  R_1: $4^5 \cdot 12^7 \cdot 1 = 8 = 2^3$, hence $\bar{v}_1 = (3, 0)$.
  R_2: $4^4 \cdot 12^9 \cdot 1 = 12 = 2^2 \cdot 3^1$, hence $\bar{v}_2 = (2, 1)$.
  R_3: $4^3 \cdot 12^5 \cdot 1 = 2 = 2^1$, hence $\bar{v}_3 = (1, 0)$.

  Note: The choice $4^3 \cdot 12^5 \cdot (-1) = 21 = 3 \cdot 7$ was unsuccessful.

Algorithm SEDL

Example - continued

- Stage 2 - we count over $\mathbb{Z}_{11}$, by Gaussian elimination we find a non-trivial solution for $c_1(3,0) + c_2(2,1) + c_3(1,0) = (0,0)$ which is $c_1 = 1$, $c_2 = 0$, $c_3 = -3 = 8$.
- Completing of calculations - $R_1^1 \cdot R_2^0 \cdot R_3^8$ gives equality:
  $4^{29} \cdot 12^{47} \cdot 1 = 2^{11} = 1$ in $\mathbb{Z}_{23}^*$,
  while $4, 12 \in G$, so we count modulo 11 in the exponent:
  $4^7 \cdot 12^3 = 1$ in $\mathbb{Z}_{23}^*$ is a non-trivial representation of 1.
- Hence $3x + 7 = 0$ in $\mathbb{Z}_{11}$, $x = -7 \cdot 3^{-1} = 5$.
  The discrete logarithm $\mathrm{dlog}_4(12) = 5$.

Algorithm SEDL

Generalization of the algorithm SEDL

The algorithm SEDL can be modified to count discrete logarithm in a subgroup G of order q^e in $\mathbb{Z}_p^*$, where p, q are primes, $|\mathbb{Z}_p^*| = q^e m$, $q \nmid m$. Let H be a subgroup of order m in $\mathbb{Z}_p^*$.
The algorithm SEDL still works because $\mathbb{Z}_p^* = G \dot{\times} H$.
The first stage proceeds in the same way, in the second stage we should solve a homogeneous system of equations over the ring $\mathbb{Z}_{q^e}$.

Gaussian elimination over a ring does not work in general, but in this case it can modified so that it will find a non-trivial solution, which are coefficients $c_1, \dots, c_{k+1} \in \mathbb{Z}_{q^e}$, not all zero and even not all divisible by q. Then the counted t has a chance to be invertible in $\mathbb{Z}_{q^e}$, which happens if $q \nmid t$. So a non-trivial representation of 1 could be found, and the discrete logarithm can be computed.

Algorithm SEDL

Exercise

Suppose we are able to use the algorithm SEDL to compute the discrete logarithm in a subgroup G' of order q^e of the group $\mathbb{Z}_p^*$, where $|\mathbb{Z}_p^*| = p - 1 = q^e m$, $q \nmid m$. Designe an algorithm that computes the discrete logarithm in the subgroup G of order q of $\mathbb{Z}_p^*$, where $q \mid p - 1$ (without any further assumption on q).

Input: the generator a of the group G, $b \in G$, p, q primes
Output: $x = \mathrm{dlog}_a(b)$ in G
Hint: Note that $G \subseteq G'$. Find the generator c of the group G', compute $\mathrm{dlog}_c(a)$, $\mathrm{dlog}_c(b)$ in G' and count x from them.

Analysis of the algorithm SEDL

Let's go back to the basic version of the algorithm SEDL which computes the discrete logarithm of the element b in the subgroup $G = \langle a \rangle$ of order q of the group $\mathbb{Z}_p^*$, where p, q are primes, $|\mathbb{Z}_p^*| = p - 1 = qm$ and $q \nmid m$.
We want to analyze the output and the expected running time of the algorithm.

Proposition

The probability that the algorithm SEDL reports a failure is $\frac{1}{q}$.

It can be shown that every $t \in \mathbb{Z}_q$ can be found by the algorithm SEDL with the same probability. Then $P[t = 0] = \frac{1}{q}$.

Analysis of the algorithm SEDL

Expected time of the algorithm SEDL

- First stage: Let's denote by σ the probability that a random element from $\mathbb{Z}_p^*$ is $y-$smooth. Then the expected number of loops for finding one $y-$smooth integer of the form $a^{s_i} b^{t_i} h_i \in \mathbb{Z}_p^*$ equals to $\frac{1}{\sigma}$. We divide each integer by all k primes up to y ($y < p$), which takes the time $k \operatorname{len}(p)^2$. We need to find $(k+1)$ such $y-$smooth integers. $E(TIME1) = O(\frac{k^2}{\sigma} \operatorname{len}(p)^2)$
- Second stage: Gaussian elimination on a matrix of type $(k, k+1)$ requires roughly k^3 operations in $\mathbb{Z}_q$ and its time dominates in the second stage. $TIME2 = O(k^3 \operatorname{len}(p)^2)$
- Expected time for SEDL: $E(TIME) = O((\frac{k^2}{\sigma} + k^3) \operatorname{len}(p)^2)$

Analysis of the algorithm SEDL

Expected time of the algorithm SEDL

We shall estimate k and σ using y.

Assume that $y = e^{\ln(p)^{\lambda+o(1)}}$, $0 < \lambda < 1$, so that we can use the Theorem 1 estimating the number of $y-$smooth integers up to p.

- $\sigma = \frac{\Psi(y,p-1)}{p-1} \geq \frac{\Psi(y,p)}{p} \geq e^{(-1+o(1))\frac{\ln(p)}{\ln(y)}\ln(\ln(p))}$
- By Chebyshev's theorem, $k = \pi(y) = \Theta(\frac{y}{\ln(y)})$.
  So it can be deduced (for any y) that $k = e^{(1+o(1))\ln(y)}$.
- $\mathrm{len}(p)^2 = e^{o(1)\ln(y)}$ due to our assumption for y.

Analysis of the algorithm SEDL

Expected time of the algorithm SEDL

We plug in $E(TIME) = O((\frac{k^2}{\sigma} + k^3)\,\mathrm{len}(p)^2)$ to get an estimate:

$$E(TIME) \leq e^{(1+o(1))\max\{\frac{\ln(p)}{\ln(y)}\ \ln(\ln(p))+2\ln(y);\,3\ln(y)\}}$$

Now we want to choose the parameter y so that the estimate of the expected time is minimal.

Let's denote $\mu = \ln(y)$, $A = \ln(p)\ln(\ln(p))$.
We want to find a minimum of function $f(\mu) = \max\{\frac{A}{\mu} + 2\mu; 3\mu\}$,
we use the basic calculus (zero first derivation).

Analysis of the algorithm SEDL

Expected time of the algorithm SEDL

For $f_1(\mu) = \frac{A}{\mu} + 2\mu$ is $f_1'(\mu) = -\frac{A}{\mu^2} + 2 = 0$ for $\mu = \pm\sqrt{\frac{A}{2}}$.

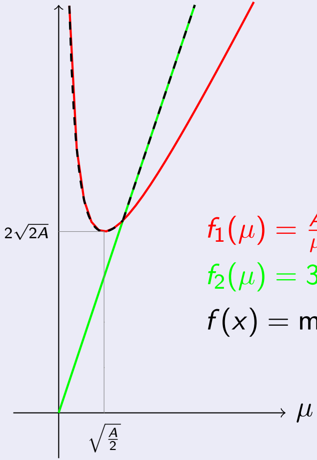
A local minimum is at $\mu = \sqrt{\frac{A}{2}}$, the value of the minimum is $4\sqrt{\frac{A}{2}}$.

The function $f_2(\mu) = 3\mu$ takes the value $3\sqrt{\frac{A}{2}}$ in this point.

Thus $\mu = \sqrt{\frac{A}{2}}$ is the minimum point for $f(\mu) = \max\{f_1(\mu); f_2(\mu)\}$ and the value of the minimum is $4\sqrt{\frac{A}{2}} = 2\sqrt{2A}$.

Analysis of the algorithm SEDL

Expected time of the algorithm SEDL



$$f_1(\mu) = \frac{A}{\mu} + 2\mu,$$

$$f_2(\mu) = 3\mu$$

$$f(x) = \max\{f_1(\mu), f_2(\mu)\}$$

Analysis of the algorithm SEDL

Expected time of the algorithm SEDL

We choose the parameter $y = e^{\sqrt{\frac{A}{2}}} = e^{\frac{1}{\sqrt{2}}\sqrt{\ln(p)\ln(\ln(p))}}$
(note that it satisfies the assumption of our calculation).

For this y, the expected time of algorithm SEDL will be

$$E(TIME) \leq e^{(2\sqrt{2}+o(1))\sqrt{\ln(p)\ln(\ln(p))}},$$

thus subexponential with constant $2\sqrt{2} \doteq 2.828$ in the exponent.

Note

The constant in the exponent can be reduced to 2.0 if we use a better estimate of number of $y-$smooth integers (Theorem 2).

Algorithm SEDL

Literature

- Shoup: A Computational Introduction to Number Theory and Algebra. Chapter 15.
- Linear spaces over a field can be found in Chapter 13. http://shoup.net/ntb/