CSE291-14: The Number Field Sieve

https://cseweb.ucsd.edu/classes/wi22/cse291-14

Emmanuel Thomé

February 3, 2022

Part 5a

Collecting relations in NFS

Plan

Plan

The name of the game

The relation collection process as a whole is like this:

Input.

- 2 polynomials f_0 and f_1 such that $N \mid \mathrm{Res}(f_0, f_1)$;
  e.g. $\deg f_0 = 1$ *and* $\deg f_1 > 1$.
- ($f_{0,1}$ output by `polyselect`, or derived from SNFS setting).

Output.

- A set of many, many relations:

  $$a, b : p_1, \ldots, p_k : q_1, \ldots, q_\ell.$$

  with p and q prime numbers below some bounds B_0 and B_1.
- Slight abuse of notations: the integers
  $F_0(a, b) = \mathrm{Res}(f_0, a - bx) = b^{\deg f_0} f_0(a/b) = \prod p_i$ and
  $F_1(a, b) = \mathrm{Res}(f_1, a - bx) = b^{\deg f_1} f_1(a/b) = \prod q_i$
  are often called norms.

What a relation encodes

As we have seen, a relation such as

$$a, b : p_1, \ldots, p_k : q_1, \ldots, q_\ell.$$

encodes a factorization in some algebraic structure, with some info that is only implicit.

Example: non-monic linear polynomial

If we have $f_0(x) = m_1 x - m_0$, the interpretation is:

$$\begin{aligned} \mathrm{Res}(a - bx, f_0(x)) &= m_1 a - m_0 b \\ &= \pm p_1 \times \cdots p_k. \\ a - b(m_0/m_1) &= \pm \frac{1}{m_1} p_1 \times \cdots p_k. \end{aligned}$$

Interpretation (2)

Interpretation on the algebraic side

If $\mathbb{Q}[x]/f_1(x) = \mathbb{Q}(\alpha)$, the interpretation of the right part is as follows.

- Assume for example that only q_1, q_3 are primes modulo which algebraic work is a bit harder.
- All other primes are "easy primes".

$$F_1(a,b) = q_1 \times \cdots q_k.$$

$$\langle a - b\alpha \rangle = J^{-1} \times \qquad \text{(reminder: } J = \langle 1, \alpha \rangle^{-1}\text{)}$$

$$\times \text{ some ideals above } q_1 \text{ and } q_3 \text{ (work needed)}$$

$$\times \text{ some trivially described ideals above other primes.}$$

Coprimality of a, b

Fact. If (a, b) gives a relation but $d = \gcd(a, b)$ is non-trivial, then $(a/d, b/d)$ gives almost the same relation.

Later on in the algorithm, those two relations cannot be non-trivially combined.

Consequence. The output of the algorithm must contain only primitive relations.

Rem. The number of coprime pairs is a constant fraction of the total number of pairs.

Rem. In practice, computing all the GCDs in advance is too costly: do it only on selected locations at the appropriate time.

Plan

Introduction

- The name of the game
- Many ways to reach the same goal
- Terminology

A promise from asymptotic analysis

When we did the analysis, we had:

- A large space $\mathcal{A}$ to choose from ($\#\mathcal{A} = L_N[1/3, \boldsymbol{+}]$)
- a probability π that $\phi(x) = a - bx$ give rise to a doubly-smooth relation. ($\pi = L_N[1/3, \boldsymbol{-}]$).
- a target number of relations, say $\mathbf{B} = L[1/3, \text{something}]$. constraint: $(\#\mathcal{A}) \times \pi \geq \mathbf{B}$.

Then in that case, given the quantities at stake, we claimed that we had a way to find AND output all **B** required relations for a cost of $(\#\mathcal{A})^{1+o(1)} = L_N[1/3, \boldsymbol{+} + o(1)]$.

Our claim: sieving can do that. Sieving = today.

Small excursion: good and bad ways to do otherwise.

Very naive algorithm

1. For a in a certain range:
2. For b in a certain range:
3. For all $p < B_0$
4. check if $p \mid F_0(a, b)$.
5. For all prime ideals $\mathfrak{p}$ of norm $< B_1$
6. check if $\mathfrak{p} \mid J \times \langle a - b\alpha \rangle$.
7. If smooth on both sides, print the relation.

Cost: $(\#\mathcal{A}) \times \mathbf{B} = L_N[1/3, \text{too much}]$.

This naive technique does not work.

Same, with ECM

1. For a in a certain range:
2. For b in a certain range:
3. Check smoothness of $F_0(a, b)$ with ECM;
4. Check smoothness of $F_1(a, b)$ with ECM;
5. If doubly-smooth, print the relation.

ECM takes time $L_B[1/2, \sqrt{2} + o(1)]$ to find prime factors below B.

With $B = L_N[1/3, \text{something}]$, that means

Same, with ECM

1. For a in a certain range:
2. For b in a certain range:
3. Check smoothness of $F_0(a, b)$ with ECM;
4. Check smoothness of $F_1(a, b)$ with ECM;
5. If doubly-smooth, print the relation.

ECM takes time $L_B[1/2, \sqrt{2} + o(1)]$ to find prime factors below B.

With $B = L_N[1/3, \text{something}]$, that means $L_N[1/6, \ldots]$.

$(\#\mathcal{A}) \times L_N[1/6, \ldots]$ is $(\#\mathcal{A})^{1+o(1)}$: that works!

- Good news: no memory needed. Infinitely parallelizable. May be relevant for GPU, FPGA, ASIC, at least to a certain extent.
- Bad news: not fast in practice.

Over the top

Another approach:

1. Divide the set of (a, b) into many subsets.
2. For each subset ($=$ many (a, b) pairs):
3. Compute product tree of all $F_0(a, b)$.
4. Compute product tree of all $F_1(a, b)$.
5. Compute remainder trees on both sides.
6. Recover smooth pairs, print relations.

As crazy as it may seem, it works, too!
Important detail: choose subsets so that products are balanced.

The sieving approach (today)

1. Divide the set of (a, b) into many subsets.
2. For each subset ($=$ many (a, b) pairs):
3. For each $p < B_0$:
4. Mark (a, b) such that $p \mid F_0(a, b)$.
5. For each $p < B_1$:
6. Mark (a, b) such that $p \mid F_1(a, b)$.
7. For each (a, b) with a large recorded contribution:
8. Compute and factor $F_0(a, b)$.
9. Compute and factor $F_1(a, b)$.
10. Print relation.

Sieving per se (steps 3-6) costs $(\#\mathcal{A}) \sum_i \log\log B_i$ in total, and steps 8- are only executed for a small fraction π of the input.

Keep in mind: several methods

There is no single way to do relation collection

In practice, the most efficient way involves a blend of:

- Sieving (in order to detect);
- Trial-division;
- Re-sieving (in order to factor);
- ECM;
- Product trees (which can replace sieving entirely).

And in the case of NFS, we have two sides to deal with, with "norms" of very different size to factor. The chain of algorithms need not be the same on both sides.

Plan

Factoring slang

- The name NFS highlights the importance of the sieving process.
  Yet, sieving is not the only way!
- QS variants (large primes, special-q) also come into play here, and lead to the presence of certain types of primes in the relations.

Our preferred terminology

- Large prime bound (for historical reasons): the largest prime that appears in the relations. This is independent of the method that we use for collecting relations.
- Algorithms such as sieving, product trees, etc, may be parameterized in many ways, and this conditions the shape of the relation that they output.

Plan

Introduction

Sieving and special-q

The sieving primes

Outer/inner aspects

Two directions of study:

- Outer aspect: how the work is divided into pieces in general;
- Inner aspect: what we do with each piece.

We start with the outer aspect.

Line sieving

Idea. The (a, b) search area is too large. Cut it into sub-areas that are handled sequentially / independently.

First strategy

Cut the sieving space (rectangle) in lines, according to b.

This is called line sieving. Has been widely used in the past.

- Line sieving alone is no longer competitive, because we can do better.
- However, line sieving is still part of special-q sieving.

Special-q

Better strategy: sieving by special-q

We want to focus on pairs (a, b) so that a specific prime number is forced to appear in the factorization of $\text{Res}(F_0(x), a - bx)$ or $\text{Res}(F_1(x), a - bx)$.

(good to know: this is one prime that we will not have discover in the factorization of the norm!)

Line sieving vs special-q sieving

Line sieving

The work area is divided in lines (constant b).

For each line, we have to loop through many primes to identify for which a we have a contribution to record as $T[a] += \log p$.

Special-q sieving

The work area is divided in sublattices of $\mathbb{Z}^2$.

Each piece of work explores combinations $i \times (a_0, b_0) + j \times (a_1, b_1)$, with (i, j) ranging over a fixed rectangle, e.g. $2^I \times 2^{I-1}$.

We have to loop through many primes to identify for which (i, j) we have a contribution to record as $T[(i, j)] += \log p$.

Plan

Sieving and special-q

- Special-q and lattice bases
- Forcing divisibility by an ideal
- Special-q: good or bad

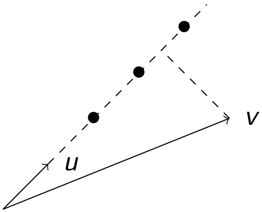
How do we define the q-lattice

Here is a lattice

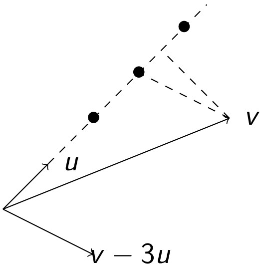
The set of (a, b) such that $q \mid am_1 - bm_0$ is a lattice (say $\mathcal{L}_q$).

Basis (if $\gcd(m_1, q) = 1$): $\begin{cases} (a_0, b_0) = (q, 0) \\ (a_1, b_1) = ((m_0/m_1) \bmod q, 1) \end{cases}$

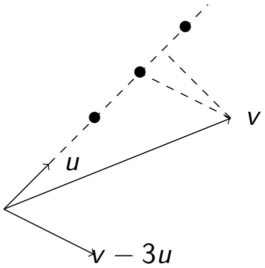
The Gauss reduction algorithm



The Gauss reduction algorithm



The Gauss reduction algorithm




```
while (!done) {
  v ← v − ⌊(v·u)/(u·u)⌉ u
  swap u and v
}
```

Repeating this produces an almost-orthogonal basis for $\mathcal{L}_\mathfrak{q}$.

This is is inherently attached to a scalar product. Here:

$$(a_0, b_0) \cdot (a_1, b_1) = a_0 a_1 + b_0 b_1.$$

The reduced basis

With Gauss reduction, we obtain the reduced basis.
We typically expect $a_0 \approx a_1 \approx b_0 \approx b_1 \approx \sqrt{q}$.

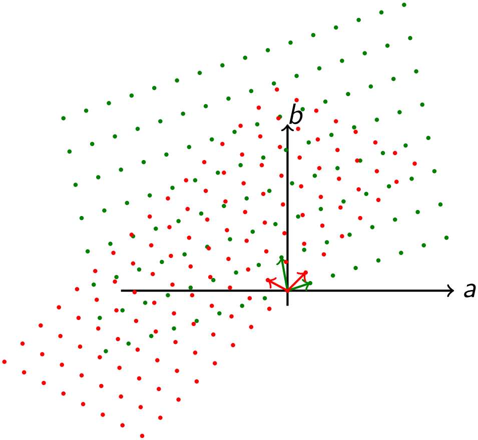
This basis defines the correspondence:

$$(a, b) = i(a_0, b_0) + j(a_1, b_1).$$

We have two 2-dimensional spaces:

- the (i, j) plane: always a fixed-size rectangle.
- and the (a, b) plane. The set of reached points is isotropic.

Special-q



In both cases, the reduced basis defines a change of basis.

q-lattice versus skewness

If our polynomial has skewness S, we have to adapt our lattice reduction in order to reach basis vectors with $a/b \approx S$.

Scalar product for skewed-Gauss

The adaptation to the skewed case simply uses the following alternate scalar product:

$$(a_0, b_0) \cdot (a_1, b_1) = a_0 a_1 + S^2 b_0 b_1.$$

E.g. the vectors $(\sqrt{S}, 1/\sqrt{S})$ and $(-\sqrt{S}, 1/\sqrt{S})$ are orthogonal with respect to this scalar product.

Skewed-Gauss reduction

We expect a pair of vectors (a_0, b_0) and (a_1, b_1) with entries:

$$a_0 \approx a_1 \approx \sqrt{qS}.$$
$$b_0 \approx b_1 \approx \sqrt{q/S}.$$

Note that it doesn't reduce much if $q < S$ (and the orders of magnitude above do not hold), but this case does not occur in practice.

Plan

Sieving and special-q

- Special-q and lattice bases
- Forcing divisibility by an ideal
- Special-q: good or bad

Most ideals are easy ones

Reminder from two weeks ago

An easy ideal $\mathfrak{q}$ represented by $(q, x - r)$ is the prime ideal above q that contains all algebraic integers that are $\mathcal{O}_K$-multiples of $(\alpha - r)$.

This implies that $\mathfrak{q}$ divides $J \times (a - b\alpha)$ if and only if $a/b \equiv r \mod q$.

Basis of the lattice of (a, b) such that $\mathfrak{q} \mid (a - b\alpha) \times J$:

$$\mathcal{L}_\mathfrak{q} : \begin{cases} (q, 0) \\ (r, 1) \end{cases}$$

This can undergo Gauss reduction, and works the same way.

Special-q versus special-$\mathfrak{q}$

Remember: there might be several ideals above the same q.

- On the rational side ($\deg f_0 = 1$), q is enough to refer to a particular set of (a, b) such that $p \mid F_0(a, b)$.
- In contrast, on the algebraic side, we really need the description of an ideal, not just the information of the prime number.
  We may speak of special-$\mathfrak{q}$ sieving in this case (for terminology nerds).

Plan

Sieving and special-q

- Special-q and lattice bases
- Forcing divisibility by an ideal
- Special-q: good or bad

Special-q sieving brings many changes

Special-q sieving is not a way to do the same thing as before.

It is really an important change of the relation collection process.

- The division of the work is not the same as with line sieving.
- The relations that we obtain are different.

What are the advantages and disadvantages of special-q sieving?

Division of the work is different

With special-$\mathfrak{q}$ sieving, we have many $\mathfrak{q}$'s of the same size.

- The yield per $\mathfrak{q}$ is more stable.
- It is easier to make projections of the total yield.

In contrast, line sieving suffers from much more irregular yields, which also drop more quickly.

- The diminishing returns effect is significant.
- Projections are harder to make.

Not the same relations

Cons:

- We miss relations that are very smooth. If the primes involved in the factorization are smaller than all special-$\mathfrak{q}$, the (a, b)-pair belongs to no $\mathfrak{q}$-lattice.
  Such relations are extremely rare anyway.
- Some relations occur several times. If the factorization corresponding to an (a, b)-pair contains two primes of the sizes of the special-$\mathfrak{q}$'s, it belongs to the two $\mathfrak{q}$-lattices.

Pros:

- We know in advance that one norm is divisible by q.
- We avoid considering some positions that are obviously non-smooth. *e.g. when the norm is prime or almost prime.*

Rem. There are more primes and almost primes than very smooth numbers.

Choosing the side of the special-$\mathfrak{q}$

Question. On which side do we put the special-$\mathfrak{q}$?

Consider two numbers; the sum of their sizes is fixed.

Is it more likely for them to be simultaneously smooth if they have the same size or if they are unbalanced?

Choosing the side of the special-$\mathfrak{q}$

Question. On which side do we put the special-$\mathfrak{q}$?

Consider two numbers; the sum of their sizes is fixed.

Is it more likely for them to be simultaneously smooth if they have the same size or if they are unbalanced?

Answer. The concavity of $\log \rho$ tells that it is better to balance the sizes.

Choosing the side of the special-$\mathfrak{q}$

Consequence. Choose q on the side that gives the largest norms.

- In the GNFS case, the algebraic side is the heaviest.
- In the SNFS case, there are low-degree cases where it is better to put the special-$\mathfrak{q}$ on the rational side.

Rem. There might exist cases with no clear answer (esp. SNFS):

- In such cases, alternating the sides of the special-$\mathfrak{q}$ can make sense (recently: HSNFS-1024 DLP).
- Or it is even possible to work with hybrid special-q.
  The set of (a, b) such that $q \mid F_0(a, b)$ and $\mathfrak{q} \mid (a - b\alpha) \times J$ is the intersection of two lattices, and it is in turn a lattice: everything can work pretty much the same way!

Plan

What do we have to do?

We have chosen two basis vectors, and we're going to explore (i, j) in a fixed-size rectangle $[-2^{I-1}, 2^{I-1}] \times [1, J]$. We have:

$$(a, b) = i \cdot (a_0, b_0) + j \cdot (a_1, b_1).$$

On each side, we want to sieve:

- Allocate a big array.
- Initialize each cell with $\log |F(a, b)|$ (for the appropriate F). This is (log-)norm initialization (not today)
- For many primes p, subtract $\log p$ from all array cells when p divides. This is sieving proper (today + Tuesday)

Once this is done, it takes some extra work to list the array cells with the smallest cofactors.

Preferred viewpoint: most general

WLOG, we will assume that we work on the number field side. Everything applies (only simpler, at times) to the rational side as well.

The rational analogue of “the ideal (p, r)” is “the ideal $(p, (m_0/m_1) \bmod p)$”. In other words, r is implicit in the rational case, as it is directly inferred from the polynomial.

The sieving primes

Sieving involves a loop over primes. Which primes?

This is an implementation detail of sieving.

- The more relevant quantity globally is the large prime bound.
- Whether sieving deals with prime ideals of norm within one range or another is only of interest to sieving itself.

Terminology

- The factor base is the set of prime ideals that are considered during sieving.
- The sieving bound or factor base bound is the upper bound on the norms of the ideals in the factor base.

Beware

We had (and we still have) $\mathfrak{q}$, which encodes an ideal that we will force into all relations produced.

Now we also consider $\mathfrak{p}$, which is some ideal in the factor base.

So $\mathfrak{p}$ is not $\mathfrak{q}$(also, we want them coprime).

Challenge

Within the (i, j) rectangle, we want to identify locations where $\mathfrak{p}$ divides $(a - b\alpha) \times J$.

The $\mathfrak{p}$-lattice

Let $\mathfrak{p}$ be an ideal to be sieved.

Fact: The set of positions in the (i, j)-plane where $\mathfrak{p}$ divides $(a - b\alpha) \times J$ is a lattice $\mathcal{L}_{\mathfrak{p}}$.

We already encountered this for the easy primes, but this holds more generally.

- Each (power of a) prime ideal of inertia degree one gives rise to a lattice in the (a, b) plane.
- We'll see how it connects to the (i, j) plane.

Plan

The sieving primes

Sieving primes in the (a, b) plane

From (a, b) to (i, j)

prime ideal $\rightarrow$ lattice in (a, b) plane

Given a prime ideal $\mathfrak{p}$ above p.

- $\mathfrak{p}J^{-1} \cap (\mathbb{Z} + \alpha\mathbb{Z})$ is a lattice. It has a basis.
- Some cases are uninteresting: if $\mathfrak{p}$ has inertia degree > 1, then intersection points only have $p \mid \gcd(a, b)$ (except possibly if $\mathfrak{p} \mid J$).

The description of the basis in the (a, b) plane only involves some non-trivial work for the rare non-easy ideals. Anyway it can be done beforehand.

In effect, we are interested in the description of sets of sieving locations, and we know that each of these is going to be a lattice in the (a, b) plane.

Example of description

Example

Let $f = 3x^4 + x^3 + x^2 + x + 1$. Algebraic number theory tells us that $3\mathcal{O}_K$ splits into three ideals, of norm 3, 3, and 9.

$$3\mathcal{O}_K = \mathfrak{p}_1\mathfrak{p}_2\mathfrak{p}_3.$$

- $\mathfrak{p}_1 \mid (a - b\alpha) \times J$ iff $a \equiv b \mod 3$.
- $\mathfrak{p}_2 \mid (a - b\alpha) \times J$ iff $b \equiv 0 \mod 3$.
- $\mathfrak{p}_3$ never divides $(a - b\alpha) \times J$, unless a, b are both multiples of 3 (and we disregard this case).

Another example of description

Example

Let $f = 9x^4 - x^2 - 5$. Algebraic number theory tells us that $3\mathcal{O}_K$ splits into four ideals, each of norm 3.

$$3\mathcal{O}_K = \mathfrak{p}_1\mathfrak{p}_2\mathfrak{p}_3\mathfrak{p}_4.$$

- $\mathfrak{p}_1 \mid (a - b\alpha) \times J$ iff $a \equiv b \mod 3$.
- $\mathfrak{p}_2 \mid (a - b\alpha) \times J$ iff $a \equiv -b \mod 3$.
- Both $\mathfrak{p}_3$ and $\mathfrak{p}_4$ divide $(a - b\alpha) \times J$ iff $b \equiv 0 \mod 3$.
  - $\mathfrak{p}_3^2 \mid (a - b\alpha) \times J$ iff $3a - b \equiv 0 \mod 9$.
  - $\mathfrak{p}_4^2 \mid (a - b\alpha) \times J$ iff $3a + b \equiv 0 \mod 9$.

Compact encoding

All congruences we have to deal with in the (a, b) plane (for prime ideals and their powers) are of the form:

$$\lambda a - \mu b \equiv 0 \mod p^k.$$

- If $p \nmid \lambda$, WLOG we can assume $\lambda = 1$. This encodes the most common case of easy ideals ($(p, r) \to a - rb \equiv 0 \mod p$), and this extends to powers with $0 \leq r < p^k$.
- Or $p \mid \lambda$, whence $p \nmid \mu$ and WLOG we can assume $\mu = 1$. We obtain a relation of the form

  $$psa - b \equiv 0 \mod p^k$$

  with p^{k-1} possible choices for s. (Note: s not necessarily coprime to p.)

Compact encoding

Another way to look at this mathematically speaking is to relate this with the space $\mathbb{P}^1(\mathbb{Z}/p^k\mathbb{Z})$ which has $p^k + p^{k-1}$ elements:

- First case: the affine point $(r:1)$ in $\mathbb{P}^1(\mathbb{Z}/p^k\mathbb{Z})$.
- Second case: the point $(1:ps)$ in $\mathbb{P}^1(\mathbb{Z}/p^k\mathbb{Z})$ which is "at infinity". In such cases, the NFS folklore uses the term projective roots (they exist only projectively).

In Cado-NFS, the computation of the factor bases is done prior to sieving, and gives for each prime power lists of one of the two encodings above.

Compact encoding → lattice

Affine case

For a set of sieving locations modulo p^k, described in compact encoding by an affine integer $r < p^k$, the lattice basis is

$$\begin{cases} (a_0, b_0) = (p^k, 0) \\ (a_1, b_1) = (r, 1) \end{cases}$$

Projective case

For a set of sieving locations modulo p^k, described in compact encoding by an integer $s < p^{k-1}$ which denotes the point $(1 : ps)$ on the projective line, the basis is (assuming $\nu_p(ps) = c > 0$):

$$\begin{cases} (a_0, b_0) = (p^{k-c}, 0) \\ (a_1, b_1) = ((ps/p^c)^{-1} \bmod p^{k-c}, p^c) \end{cases}$$

Four points of view

- More mathematical: some power of a prime ideal.
- Also mathematical: a point in $\mathbb{P}^1(\mathbb{Z}/p^k\mathbb{Z})$.
- More down-to-earth: a lattice basis.
- More compact: an integer between 0 and $p^k + p^{k-1}$.
  E.g. by letting $p^k + s$ encode $(1 : ps)$

These are equivalent ways of describing the same thing: a set of (a, b) pairs where we know that some divisibility condition is met.

Plan

The sieving primes

Sieving primes in the (a, b) plane

From (a, b) to (i, j)

Where do we sieve in the (i, j) plane?

$\mathcal{L}_\mathfrak{p}$: locations of interest in the (a, b) plane.
$\mathcal{L}_\mathfrak{p} \cap \mathcal{L}_\mathfrak{q}$ has a basis in the (i, j) plane.

$$(a, b) = i \cdot (a_0, b_0) + j \cdot (a_1, b_1).$$

Example: affine case

$$\begin{aligned} a - rb &\equiv 0 \mod p^k \\ \Leftrightarrow (ia_0 + ja_1) - r(ib_0 + jb_1) &\equiv 0 \mod p^k \\ \Leftrightarrow i(a_0 - rb_0) + j(a_1 - rb_1) &\equiv 0 \mod p^k \\ \Leftrightarrow i - Rj &\equiv 0 \mod p^k \end{aligned}$$

with $R \equiv -\dfrac{a_1 - rb_1}{a_0 - rb_0} \mod p^k$ if the denominator is invertible!

Translation from (a, b) to (i, j)

This is preparatory work that must be done for each special-$\mathfrak{q}$:

Transforming the factor base

For each prime ideal (power) or more down-to-earth representation in the (a, b) plane, compute the down-to-earth representation in the (i, j) plane.

Anything can happen:

- affine in $(a, b) \to$ affine in (i, j).
- affine in $(a, b) \to$ projective in (i, j).
- projective in $(a, b) \to$ affine in (i, j).
- projective in $(a, b) \to$ projective in (i, j).

Ready!

At this point, our gear is packed, we're ready to sieve.

We have:

- a huge array $T[]$, indexed by (i, j).
  $T[]$ can be up to several gigabytes of RAM.

And we also have:

- A (long) list of prime (powers) p^k (not that we must sieve powers, but for sure we can).
- Each comes with a descriptions of the location of hits: places in the (i, j) plane where we want to subtract $\log p$.