Some Applications of Chinese Remainder Theorem Codes with Error-Correction

Jesse Elliott[1], Éric Schost[1]

[1]University of Waterloo, David R. Cheriton School of Computer Science, Waterloo, Ontario, Canada

Abstract

Modular techniques with rational reconstruction improve complexity when computing over a ground field such as $\mathbb{Q}$ by controlling the growth of intermediate expressions. Working modulo a single prime $p \in \mathbb{N}$, one can solve the problem modulo p and lift the solution to $\mathbb{Z}/p^k\mathbb{Z}$ for sufficiently large p^k using p-adic lifting techniques, when applicable. Alternatively, computations can be done modulo several small primes $p_1, \dots, p_\eta$. One can then obtain a solution modulo their product $p_1 \cdots p_\eta$ using the Chinese remainder theorem. We say primes p are "unlucky" when the procedure modulo p is not well-defined or returns a result that is different from the modulo p reduction of the rational output. Otherwise we say primes are "lucky." For some applications (solving zero-dimensional polynomial systems, for instance), testing if a prime is lucky may be prohibitively expensive. However, it is often possible to bound the number of unlucky primes by proving the existence of a nonzero $U \in \mathbb{Z}$ with all unlucky primes dividing U, and bounding the height of U. Using p-adic lifting requires the initial prime to be lucky, with a high probability of success that is determined by an upper bound on U. On the other hand, the Chinese remainder theorem requires that all primes are lucky, and to guarantee this with high probability usually requires larger primes. We report on work-in-progress that uses error correction techniques with Chinese remaindering that allows us to tolerate a few unlucky primes. Our hope is to then guarantee a high probability of success while using primes of moderate size.

We base our work on independent results from Böhm, Decker, Fieker and Pfister [1, 2] and Pernet [3]. To our knowledge, the consequences we derive, while relatively straightforward, are new. We give explicit sufficient conditions on the number of primes and their size to guarantee an arbitrary probability of success, assuming we can pick primes uniformly at random in a given interval. We also describe a number of applications.

Keywords

Chinese remainder theorem codes, polynomial system solving, modular algorithms

1. Background and previous work

Solving algebraic problems over a ground field such as $\mathbb{Q}$, considering only *algebraic complexity* (the number of base field operations) is hardly a good predictor of practical runtime: a precise analysis should take into account the size of the coefficients in the output, and the number of boolean operations throughout the execution of the algorithm.

One major challenge in such algorithms is the growth of coefficients. In many situations, we can give reasonably sharp *a priori* bounds on the bit size of the coefficients in the output

SCSS 2024: 10th International Symposium on Symbolic Computation in Software Science, August 28–30, 2024, Tokyo, Japan

jakellio@uwaterloo.ca (J. Elliott); eschost@uwaterloo.ca (É. Schost)
https://uwaterloo.ca/scholar/jakellio/home (J. Elliott); https://cs.uwaterloo.ca/~eschost/ (É. Schost)

(a typical recipe being understanding them as determinants built from the input problem). Precisely, we will assume below that we know an upper bound H on the *height* of all rational numbers appearing in the output, where in this note the height $h(c)$ of a rational number c is the maximum of the base-2 logarithms of the absolute values of its minimal numerator and denominator. However, such insight should not be expected for intermediate results of our algorithm. As the algorithm progresses, the size of these coefficients may increase, before possibly collapsing when we reach the end result.

Modular techniques are used to avoid intermediate expression swell. They involve performing computations modulo one or several small primes (ideally, machine primes), thereby avoiding intermediate coefficient growth, the objective being to compute the requested output (typically, a set of polynomials, or a matrix thereof) modulo a certain large integer M. If M is large enough (precisely, if $\log_2(M) \geq H + 1$), rational reconstruction can then be applied coefficient-wise, in order to recover an output with rational coefficients.

On one end of the spectrum, one can consider working modulo a single prime p, solve the problem modulo p and lift the solution to $\mathbb{Z}/M\mathbb{Z}$, for $M = p^k$ large enough, by means of Newton / Hensel techniques, if applicable. The obvious alternative is to compute modulo sufficiently many "small" primes $(p_i)_{1\leq i\leq \eta}$ and use the Chinese remainder theorem to obtain a solution modulo $M = p_1 \cdots p_\eta$.

For most problems of interest, there exist primes p for which the procedure modulo p is not well-defined, or returns a result that differs from the modulo p reduction of the rational output; we will call these primes *unlucky*, and *lucky* otherwise. In the problems we have in mind, such as solving systems of polynomial equations, testing whether a prime is lucky may be prohibitively expensive. However, it is often possible to bound the number of unlucky primes: this is usually done by proving the existence of a nonzero $U \in \mathbb{Z}$ such that all unlucky primes divide U, and bounding the height of U.

Using p-adic lifting techniques, we need to ensure that the initial prime p is lucky; knowing the upper bound on U, we can determine what interval to pick p from in order to guarantee a high probability of success, say at least $1 - \varepsilon$ for a given tolerance ε. For Chinese remaindering algorithms, though, the direct approach requires all primes being lucky, and guaranteeing that this is the case with probability $1 - \varepsilon$ usually requires us to use larger primes (we discuss this further below). In this short note, we report on work-in-progress that uses error correction techniques (very loosely speaking, analogues of Reed-Solomon decoding, but for rational number reconstruction), where we tolerate that a few primes return wrong results (or no results at all). Our hope is then to be able to guarantee high probability of success, while using primes of moderate size.

We base our work on recent (independent) introductions of this idea, by Böhm, Decker, Fieker and Pfister [1, 2] and Pernet [3], the former in the context of algorithms for algebraic geometry, and the latter mentioning applications to linear algebra. The decoding algorithms and the sufficient conditions for success given in these two families of references are distinct, but similar; both follow an iterative reduction procedure, stated as a variant of Euclid's algorithm in Pernet's work, and as a variant of Gaussian lattice reduction by Böhm *et al.* Our presentation will follows Pernet's.

The core of our discussion concerns the reconstruction of a single rational number. We also point out that in the contexts we are interested in, algorithms usually return several such numbers (typically as coefficients of polynomials), and we can often predict that all these rationals admit a small common denominator. Taking this specificity into account would lead us toward error-tolerant vector rational number reconstruction; ideally, we could hope to reduce the number of primes by up to two, but as of now, this appears to be quite challenging.

2. Our contribution

Let us first review the key result regarding Chinese remaindering for rational reconstruction in the presence of errors. We consider a sequence of prime moduli $p_1, \dots, p_\eta$ and a rational number $r = f/g$, with $g > 0$. The goal is to recover r from a vector $(r_i)_{1 \le i \le \eta}$, where $r_i = r \bmod p_i$ for a certain number of *lucky* primes p_i. We tolerate a number of errors or missing values (e.g., for which p_i divides g), for which we write $r_i = \infty$, for a new symbol ∞; the corresponding primes are *unlucky*. Consider the following integers:

- $N = \sum_{i=1}^{\eta} \log_2(p_i)$
- $L = \sum_{1 \le i \le \eta, p_i \text{ unlucky prime}} \log_2(p_i)$.
- H is a given upper bound on the height of r, that is, $\log_2(|f|), \log_2(g) \le H$

Then, Pernet proved in [3, Lemma 2.5.4] that if $L < (N - 2H + 1)/2$, one can reconstruct r given the r_i's [3, Algorithm 2 p.38]. Böhm *et al.* proved similar results.

Although these statements are well-established, to our knowledge, the following consequences, while relatively straightforward, are new. We provide a quantitative analysis which gives explicit sufficient conditions on the number of primes and their size to guarantee an arbitrary probability of success, in a model where we assume we can pick primes uniformly at random in a given interval.

Stating this result requires us to take all primes into consideration. Thus, r and the upper bound H are as above, and to each prime $p \in \mathbb{N}$ corresponds a value $r(p)$ (possibly ∞); p is called lucky when $r(p) = r \bmod p$ and unlucky otherwise. We assume that there are finitely many unlucky primes and let U be their product. In addition to the output size bound H, we then need a bound C such that $\log_2(U) \le C$. Both H and C are problem-dependent (we discuss a few examples in the next section); once bounds on H and C are available, the following propositions apply.

In what follows, for simplicity, given an interval $\Sigma = \{\sigma, \dots, 2\sigma\}$, we assume that we can sample η primes in Σ uniformly without replacement, as long as this interval is known to contain at least η primes.

Proposition 1. *Let r, H, C be as above. For $\epsilon > 0$, let σ and η be integers such that $\sigma \ge \max(16, \frac{16C}{\epsilon}, 16H)$ and $\eta = \left\lceil \frac{4H}{\log_2(\sigma)} \right\rceil$. Select pairwise distinct primes $p_1, \dots, p_\eta$ independently and uniformly at random from the set $\Sigma = \{\sigma, \dots, 2\sigma\}$. Then, with probability at least $1 - \epsilon$, given $(r(p_1), \dots, r(p_\eta))$, one can reconstruct r.*

Proof. Let $\mathcal{D}$ denote the set $\{p \mid p \in \Sigma \text{ and } U \bmod p = 0\}$, and notice that the product $\prod_{p \in \mathcal{D}} p$ also divides U, so that in particular $\prod_{p \in \mathcal{D}} p \le U$. Each prime in Σ is at least equal to σ, so that

$\#\mathscr{D} \le \frac{C}{\log_2(\sigma)}$. On the other hand, the number of primes in Σ is at least $\frac{\sigma}{2\log_2(\sigma)}$ [4, Ex. 18.18], so that for a prime p chosen at random in Σ, $\mathbb{P}(p|U) \le \frac{(C/\log_2(\sigma))}{(\sigma/(2\log_2(\sigma)))} = \frac{2C}{\sigma}$.

Now, we choose η distinct primes $p_1, \dots, p_\eta$ uniformly at random in Σ, and for $i = 1, \dots, \eta$, we let X_i be the indicator variable defined as

$$X_i = \begin{cases} 1 & \text{if } p_i \mid U \\ 0 & \text{otherwise,} \end{cases}$$

so that $\mathbb{E}[X_i] = \mathbb{P}(X_i = 1) \le 2C/\sigma$. Define further $X = \sum_{i=1}^{\eta} X_i$, so that $\mathbb{E}[X] \le 2\eta C/\sigma$. Now, for any choice of η distinct primes (p_i) in Σ, the quantities N and L defined above satisfy $N = \sum_{i=1}^{\eta} \log_2(p_i) \ge \eta \log_2(\sigma)$ and $L = \sum_{1 \le i \le \eta, p_i \text{ unlucky prime}} \log_2(p_i) \le \log_2(2\sigma)X$. From [3, Lemma 2.5.4] as cited above, we know that the error-tolerant rational reconstruction algorithm succeeds as soon as $L \le (N - 2H + 1)/2$, and in particular as soon as $\log_2(2\sigma)X \le \Delta = (\eta \log_2(\sigma) - 2H + 1)/2$. We will point out below that for our choice of η, Δ is positive.

Then, the probability of failure is at most $\mathbb{P}\left(\log_2(2\sigma)X > \Delta\right) = \mathbb{P}\left(X > \Delta/\log_2(2\sigma)\right)$, which by Markov's inequality is at most $\mathbb{E}[X]/\left(\Delta/\log_2(2\sigma)\right)$. We deduce

$$\mathbb{P}(\text{fail}) \le \left(\frac{2\eta C}{\sigma}\right)\left(\frac{2\log_2(2\sigma)}{\eta\log_2(\sigma) - 2H + 1}\right) \le \frac{8C}{\sigma}\frac{\eta\log_2(\sigma)}{\eta\log_2(\sigma) - 2H} = \frac{8C}{\sigma}\frac{1}{1 - \frac{2H}{\eta\log_2(\sigma)}}.$$

Now, take $\eta = \lceil 4H/\log_2(\sigma) \rceil$, so that in particular $\eta \ge 4H/\log_2(\sigma)$, and thus $2H/(\eta\log_2(\sigma)) \le 1/2$, in which case the right-most factor in the inequality above is at most 2. Besides, this choices ensures $\Delta > 0$. To summarize, in this case, we have $\mathbb{P}(\text{fail}) \le 16C/\sigma$, and this can be made less than ϵ as soon as $\sigma \ge 16C/\epsilon$.

It remains to verify that our interval Σ contains at least η primes. We know that there are at least $\sigma/2\log_2(\sigma)$ such primes, and that η is at most $4H/\log_2(\sigma) + 1$, and one checks that if $\sigma \ge 16$ and $\sigma \ge 16H$, this is indeed less than or equal to $\sigma/2\log_2(\sigma)$. □

Remark 2. *In the context of a modular algorithm, the most important component in the cost analysis is the total time spent solving the problem modulo the primes p_i. In rough approximation, one can assume that each such execution takes T operations modulo p_i, where T is independent of i. It follows that the total boolean cost is softly-linear in $T\sum_{1 \le i \le \eta} \log_2(p_i) \in \Theta(T\eta\log(\sigma))$.*

In our construction, we have $\eta\log_2(\sigma) \le (\frac{4H}{\log_2(\sigma)} + 1)\log_2(\sigma) = 4H + \log_2(\sigma)$. In other words, the boolean cost involves both the output size H, which is as expected, together with $\log_2(\sigma)$, which will increase if we take ϵ close to zero.

Remark 3. *Assume that we do not use error-correction. In this case, in order to be able to reconstruct r, we need* all *primes to be lucky. With notation as in the proposition, we saw that the probability that a single prime is unlucky is at most $2C/\sigma$, so when choosing η primes, the probability that at least one of them is unlucky is at most*

$$1 - \left(1 - \left(\frac{2C}{\sigma}\right)\right)^{\eta} \le \frac{2\eta C}{\sigma}.$$

Assuming we choose η as above, let us derive a bound on σ that ensures $2\eta C/\sigma < \epsilon$. We proceed informally and take $\eta = 4H/\log_2(\sigma)$, so our inequality is satisfied when $8CH/(\sigma\log_2(\sigma)) < \epsilon$. Hence we require that $\sigma\log_2(\sigma) \ge 8HC/\epsilon$; this gives $\sigma \ge R(8HC/\epsilon)$, where R is the reciprocal function of $x \mapsto x\log_2(x)$. This function grows like $x/\ln(x)$, for $x \to \infty$, which gives asymptotically $\sigma \ge 8HC/(\epsilon\ln(8HC/\epsilon))$. As expected, this is inferior to the bound given in the previous proposition.

3. Applications

We end this note with a quick description of possible use cases of this work.

Computing Hermite forms over $\mathbb{Q}[x]$. In [5], Storjohann gives a modular algorithm to compute Hermite forms for matrices with entries in $\mathbb{Q}[x]$, together with bounds H on the output size and C on the unlucky primes. Our work applies directly to this situation. We are not aware of alternative methods that would rely on Newton iteration.

Computing lexicographic Gröbner bases in $\mathbb{Q}[x, y]$. In [6], St-Pierre and Schost provide similar bounds H and C for the computation of bivariate Gröbner bases; again, our work applies directly. In this case, an alternative approach based on Newton iteration exists, but has rather high complexity.

Solving zero-dimensional systems in $\mathbb{Q}[x_1, \dots, x_n]$. The main application we have in mind is the solution of zero-dimensional polynomial systems (by means of a data-structure known as a zero-dimensional parametrization, see [7] for a definition and references). When the complex solutions have multiplicity one, a simple form of Newton iteration is applicable [8], but without this assumption, lifting techniques are complex to analyze. In this case, a bound H on the output size is available by means of the arithmetic Bézout theorem, but the unlucky primes are harder to describe. The reference [9] quantifies primes p for which the number of solutions changes modulo p, but further arguments are needed to control other possible degeneracies.

References

[1] J. Böhm, W. Decker, C. Fieker, G. Pfister, The use of bad primes in rational reconstruction, Math. Comput. 84 (2012) 3013–3027.

[2] J. Böhm, W. Decker, C. Fieker, S. Laplagne, G. Pfister, Bad primes in computational algebraic geometry, in: Mathematical Software – ICMS 2016, Springer, 2016, pp. 93–101.

[3] C. Pernet, High Performance and Reliable Algebraic Computing, HDR, Université Joseph Fourier, Grenoble 1, 2014. URL: https://theses.hal.science/tel-01094212.

[4] J. von zur Gathen, J. Gerhard, Modern Computer Algebra, third ed., Cambridge University Press, 2013.

[5] Storjohann, A., Computation of Hermite and Smith Normal Forms of Matrices, Master's thesis, University of Waterloo, 1994.

[6] E. Schost, C. St-Pierre, p-adic algorithm for bivariate gröbner bases, in: ISSAC'23, ACM, 2023, p. 508–516. doi:10.1145/3597066.3597086.

[7] E. Schost, M. S. E. Din, Bit complexity for multi-homogeneous system solving application to polynomial minimization, Journal of Symbolic Computation 87 (2018) 176–206.

[8] W. Trinks, On improving approximate results of buchberger's algorithm by newton's method, SIGSAM Bull. 18 (1984) 7–11.

[9] C. D'Andrea, A. Ostafe, I. Shparlinski, M. Sombra, Reductions modulo primes of systems of polynomial equations and algebraic dynamical systems, Trans. Amer. Math. Soc. 371 (2019) 1169–1198.