Gödel's Lost Letter and P=NP

a personal view of the theory of computation

The Chinese Remainder Theorem With Limits

AUGUST 1, 2009

tags: Algorithms, BDD, chinese remainder theorem, CRT, Factoring, polynomial, polynomial time, time

by rjlipton

Coppersmith's theorem and adding size constraints to the Chinese Remainder Theorem

Sun Tzu is famous for the discovery of the **Chinese Remainder Theorem (http://en.wikipedia.org/wiki/Chinese_remainder_theorem)** (CRT) in China in the third century, way before it was known in the west. His original example was:

> *How many soldiers are there in Han Xing's army? — If you let them parade in rows of 3 soldiers, two soldiers will be left. If you let them parade in rows of 5, 3 will be left, and in rows of 7, 2 will be left.*

For those of us who have a european-centric view of mathematics this should come as a warning. No one has a monopoly on ideas. No one. Since modern crypto-systems are based on number theory, our lack of a monopoly in this area raises a real security issue. If those who do identity theft were able to obtain the right number theory secrets, for example, we would have a potential disaster—I will discuss this another time.

Today I want to talk about two results related to the CRT. One is an old—well not old as in third century old—result due to Don Coppersmith, and the other is an observation about CRT with restrictions. The latter is really a way of raising the problem of integer factoring again. Are you surprised?

The CRT is one of my favorite theorems, but also one of my least favorite theorems. I have used the CRT theorem so many times that once Avi Wigderson said, "it's the only theorem you know." Avi says he never said this, but I recall him saying it. I could be wrong, but I kind of like the statement—so I will stick to my memory.

The CRT is one of my least favorite theorems, Avi's comment notwithstanding, since it makes the task of understanding computation modulo composites hard. One of the open problems of complexity theory is what is the power of computing with gates modulo a composite? The CRT shows that the structure of $\mathbb{Z}_6$, for example, is a direct product $\mathbb{Z}_6 = \mathbb{Z}_2 \times \mathbb{Z}_3$, but that is not very helpful. For example, suppose we need to understand the structure of the solution set to equations like

$$f(x_1, \dots, x_n) \equiv 0 \bmod 6$$

where $x_1, \dots, x_n$ are *boolean* and f is a polynomial. The CRT does not seem to help here. The problem is this: the direct product of two boolean vectors need not be boolean. For instance, a vector can be boolean modulo 2 and boolean modulo 3, but not boolean modulo 6.

I recently posted on a **paper (https://rjlipton.wordpress.com/2009/07/07/linear-equations-over-composite-moduli/)** that will appear at FOCS on a related problem.

Solving an Equation Modulo N

There is a wonderful theorem, due to Coppersmith, that should be in all our tool boxes.

> ***Theorem:*** *Let N be a natural number and $f \in \mathbb{Z}[x]$ a monic polynomial of degree d. Set $X = N^{\frac{1}{d}-\epsilon}$ for some $\epsilon \geq 0$. Then, there is an algorithm to find all $0 < x < X$ satisfying*
>
> $$f(x) \equiv 0 \bmod N.$$
>
> *The running time is polynomial in $\min(\log N, 1/\epsilon)$.*

The power of Coppersmith's theorem is that N can be composite. If N has known factorization, then there are better methods for solving equations modulo N—using CRT. See Dan Boneh's survey on this **theorem (http://130.203.133.121:8080/viewdoc/summary?doi=10.1.1.19.8976)** and related results.

CRT with Limits

Suppose that $Q = (q_1, \dots, q_m)$ is a list of pairwise relatively prime natural numbers. Define $\Pi(Q)$ to be equal to

$$\prod_{i=1}^{m} q_i.$$

The famous CRT theorem shows that for any $A = (a_1, \dots, a_m)$ there is a unique X so that $0 \leq X < \Pi(Q)$ and for all indices i,

$$X \equiv a_i \bmod q_i. \qquad (1)$$

In the original problem, of Sun Tzu, $q_1 = 3, q_2 = 5, q_3 = 7$, which are clearly relatively prime.

We are interested in allowing the constraints on X modulo q_i to be more flexible. Rather than a single value a_i, we will allow X to have any value in some set. Let $S = (S_1, \dots, S_m)$ be a list of sets so that each S_i is contained in $\{0, 1, \dots, q_i - 1\}$. We plan to replace (1) with the following,

$$X \bmod q_i \in S_i. \qquad (2)$$

Note, the classic CRT is just the case where each set S_i is a singleton set. This generalization is uninteresting without some additional restriction. Note that the existence easily follows in this case because we can select some $a_i \in S_i$ for each i and there will be an X satisfying (2).

The additional constraint that we will add, first, is the constraint on the size of X. A natural constraint is force X to be in $[0, L]$ where $L < \Pi(Q)$. The obvious algorithm for this problem would be: select $a_1 \in S_1, \dots, a_m \in S_m$ and then find the unique X; then, check whether or not $X < L$. If it is then stop; otherwise, try another selection.

The difficulty with this simple algorithm is that there could be an exponential number of choices, and so the algorithm could take too long. We will show how to do better, provided two things are true: (i) the value of L is not too small compared with $\Pi(Q)$, and (ii) the inequality $X < L$ is replaced by a "softer" constraint. I will explain this shortly.

Define $\Gamma(Q, S, a, b)$ to be the set of X so that

$$a \cdot \Pi(Q) \le X \le b \cdot \Pi(Q)$$

and for each index i,

$$X \bmod q_i \in S_i.$$

Note, $\Gamma(Q, S, a, b)$ may or may not be empty: it depends on the sets and the limits in size on X. Also we will often drop the Q, S and just write $\Gamma(a, b)$ when there can be no confusion. Let $Q_{\max}$ be the maximum of the q_i's.

Theorem: *Let Q and S be as above, and let $\epsilon > 0$ and $0 < \delta_1 < \delta_2 < 1$. Then, there is an algorithm that runs in time $\mathsf{poly}(Q_{\max}, m, \frac{1}{\delta_1}, \frac{1}{\epsilon})$. The algorithm operates as follows:*

1. *If $\Gamma(\delta_1, \delta_2)$ is nonempty, then the algorithm returns an element of this set;*
2. *If $\Gamma(\delta_1 - \epsilon, \delta_2 + \epsilon)$ is empty, then the algorithm returns "none".*

Note, the algorithm essentially finds an X in the given range that satisfies a series of constraints modulo each relatively prime number. The complication in the statement of the theorem is that the inequalities on the solutions are not sharp. It is like a "no-man's land". If there is a solution in the range, the algorithm finds it; if there is no solution even near the range, the algorithm says none. For solutions in between the algorithm is allowed to fail.

Constructive CRT

The algorithm is based on the constructive version of the CRT. Consider a system of equations,

$$X \equiv a_i \bmod q_i \qquad (3)$$

where $i = 1, \dots, m$. For each i there is a $0 < q_i' < q_i$ so that $q_i' \Pi(Q)/q_i \equiv 1 \bmod q_i$. This follows since each $\Pi(Q)/q_i$ is relatively prime to q_i. Define,

$$Y = \sum_{i=1}^{m} a_i q_i' \Pi(Q)/q_i.$$

The claim is that Y is a non-negative solution to all the equations (3). This follows since $Y \bmod q_k$ is equal to

$$\sum_{i \neq k} a_i q_i' \Pi(Q)/q_i + a_k q_k' \Pi(Q)/q_k$$

which is congruent to a_k modulo q_k: the first sum's terms are all 0 modulo q_k.

Note, Y need not be less than $\Pi(Q)$, but for some t,

$$X = Y - t \cdot \Pi(Q)$$

is in $[0, \Pi(Q)]$. The question that we need to understand is: when is this X in the range $[0, \delta \cdot \Pi(Q)]$? The key is the following simple lemma: ($\{z\}$ denotes the fractional part of z)

Lemma: *There is a solution X to the equations (3) in the range $[0, \delta \cdot \Pi(Q)]$ if and only if*

$$\left\{ \sum_{i=1}^{m} a_i q_i'/q_i \right\} \le \delta \qquad (4)$$

Proof: First, assume that (3) has a solution X in the given range. Then, we must show that (4) is true. Clearly, for some non-negative t, $X = Y - t \cdot \Pi(Q)$. This follows by the CRT uniqueness and that $Y \ge 0$. Then, it follows that

$$0 \le \sum_{i=1}^{m} a_i q_i'/q_i - t \le \delta.$$

The sum is clearly equal to $\alpha + k$ where $0 \le \alpha < 1$ and $k \ge 0$ is a natural number. I claim that $k = t$. If this is true (4) will clearly follow, since α is just the fractional part of the sum. We know that

$$0 \le \alpha + k - t \le \delta.$$

Thus, if $k > t$, then the last inequality would be false; and if $k < t$, then the first would be false. Hence, the claim follows.

Second, assume that (4) is true. Then, we must show that (3) has a solution in the given range. We know that

$$\left\{ \sum_{i=1}^{m} a_i q_i'/q_i \right\} \le \delta$$

which implies that $0 \le \alpha + k \le \delta$ where again $\alpha + k$ is value of the sum. Further as before $0 \le \alpha < 1$ and $k \ge 0$ is a natural number. But, clearly $\alpha \le \delta$ by the assumption. I now claim that

$$X = Y - k \cdot \Pi(Q)$$

is a solution to all the congruences and that X satisfies the required inequalities. The first is easy, so let's turn to the inequalities. Again $X/\Pi(Q)$ is equal to

$$\sum_{i=1}^{m} a_i q_i'/q_i - k.$$

But, this is just α by definition. Thus, $0 \le X/\Pi(Q) = \alpha \le \delta$, and we are done. □

The Algorithm

We can now explain the algorithm. The basic idea is to use a **BDD (https://rjlipton.wordpress.com/2009/06/16/bdds-and-factoring/)** type approach and encode the problem of finding X into a finite state automaton. The input is in the form

$$x_1 x_2 \dots x_m$$

where each x_i is at most $l = \log Q_{\max}$ bits. The automaton reads each input and does two things. First, it checks that x_i is in the current set S_i. Second, it will compute the fractional part of

$$\sum_{i=1}^{m} x_i q_i'/q_i.$$

The first is easy and only uses order $|S_1| + \cdots + |S_m|$ states. The second requires the automaton to compute the sum of rational numbers. This cannot be done exactly, but can be done to some fixed precision. At the end the machine accepts if and only if the first part is correct and also if the fractional part is less than δ.

The details will be worked out in a detailed note that I will post shortly. However, I hope you see the basic idea that is being done. Also it should be clear now why the "soft" inequality is needed. The sum could be very close to the value of δ and this could cause an error. So to avoid this we need a gap.

Open Problems

Can we improve the CRT with limits? Even as stated does it have some interesting applications? Note, one easy extension that might be interesting. We can add another type of constraint on X. Let X correspond to

$$x_1 x_2 \dots x_m.$$

Then, we can also add the constraint that $f(x_1, \dots, x_m) = 1$ for any boolean function that can be computed by a finite state automaton in polynomial number of states. Thus, we could check that there are an even number of 1's in each x_i, or that no $x_i = x_{i+1}$.

from → People, Proofs

6 Comments leave one →

1. **Dave Pritchard PERMALINK**
   August 2, 2009 9:27 pm
   I like this, and it seems like a neat use of BDDs. In my taste the Lemma would benefit from a shorter & more direct proof. Aside from the math, I didn't understand "If those who do identity theft were able to obtain the right number theory secrets, for example, we would have a potential disaster"
   Here's one thing I could imagine you were thinking: that most cryptosystems rely on "generally accepted" unproven facts, and that it would be bad if they were to find a proof that these facts are false?
   Also, I think and hope math is a good example (in principle) of something which is kind of un-dominable by any subset of humanity…

   REPLY
2. **Joker PERMALINK**
   September 2, 2009 6:03 pm
   Super post, Need to mark it on Digg
   Thank you

   REPLY
3. **Juan PERMALINK**
   June 25, 2012 10:31 pm
   This article is very interesting

   Can I use the constructive CRT algorithm for polynomials?

REPLY

4. **John N PERMALINK**
   March 12, 2016 12:11 pm
   I had difficulties resolving the Boneh/Coppersmith hyperlink. Here's a more reliable one: http://www.ams.org/notices/199902/boneh.pdf

   REPLY

Trackbacks

1. What Will Happen When P≠NP Is Proved? « Gödel's Lost Letter and P=NP
2. Twin Primes Are Useful | Gödel's Lost Letter and P=NP